ESPOO 2005 VTT PUBLICATIONS 572

Arto Kiviniemi

Requirements management interface to building product models

VTT PUBLICATIONS 572

Requirements management interface to building product models

Arto Kiviniemi
VTT Building and Transport

A dissertation submitted to the Department of Civil and Environmental Engineering and the Committee of Graduate Studies of Stanford University in partial fulfillment of the requirements for the degree of doctor of philosophy

ISBN 951–38–6655–6 (soft back ed.)
ISSN 1235–0621 (soft back ed.)

ISBN 951–38–6656–4 (URL: http://www.vtt.fi/inf/pdf/)
ISSN 1455–0849 (URL: http://www.vtt.fi/inf/pdf/)

JULKAISIJA – UTGIVARE – PUBLISHER

VTT, Vuorimiehentie 5, PL 2000, 02044 VTT
puh. vaihde 020 722 111, faksi 020 722 4374

VTT, Bergsmansvägen 5, PB 2000, 02044 VTT
tel. växel 020 722 111, fax 020 722 4374

VTT Technical Research Centre of Finland,
Vuorimiehentie 5, P.O.Box 2000, FI–02044 VTT, Finland
phone internat. +358 20 722 111, fax + 358 20 722 4374

VTT Rakennus- ja yhdyskuntatekniikka, Lämpömiehenkuja 2, PL 1800, 02044 VTT
puh. vaihde 020 722 111, faksi 020 722 6251

VTT Bygg och transport, Värmemansgränden 2, PB 1801, 02044 VTT
tel. växel 020 722 111, fax 020 722 6251

VTT Building and Transport, Lämpömiehenkuja 2, P.O.Box 1801, FI–02044 VTT, Finland
phone internat. +358 20 722 111, fax +358 20 722 6251

Technical editing Anni Kääriäinen

Otamedia Oy, Espoo 2005

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Martin Fischer

Martin Fischer, Principal Adviser

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Boyd C. Paulson, Jr.

Boyd Paulson

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Vladimir Bazjanac

Vladimir Bazjanac

Approved for the University Committee on Graduate Studies.

Abstract

In current AEC practice client requirements are typically recorded in a building program, which, depending on the building type, covers various aspects from the overall goals, activities and spatial needs to very detailed material and condition requirements. This documentation is used as the starting point of the design process, but as the design progresses, it is usually left aside and design changes are made incrementally based on the previous design solution. As a consequence of several small changes and without any conscious decisions to change the scope, this can lead to a solution that may no longer meet the original requirements.

In addition, design is by nature an iterative process and the proposed solutions often also cause evolution in the client requirements. However, the requirements documentation is usually not updated accordingly. In the worst case the changes are recorded just in the memory of the participants, and in the best case in meeting or personal notes. Finding the latest updates and evolution of the requirements from the documentation is very difficult, if not impossible.

This process can lead to an end result which is significantly different from the documented client requirements. Some important client requirements may not be satisfied, and even if the design process was based on agreed-upon changes in the scope and requirements, differences in the requirements documents and in the completed building can lead to well-justified doubts about the quality of the design and construction process.

My observation is that even a simple active link between the client requirements and design tools can increase the use of requirements documentation throughout the design and construction process and facilitate necessary updates of the client requirements. The key limitation is the lack of a theory to link the requirements to the design systems.

A solution to the above mentioned problems can build on the following five main points of departure: (1) design as an information process, (2) existing client requirements documentation and hierarchies, (3) Lawrence Berkeley National

Laboratory's Design Intent Tool for technical systems, (4) existing IFC specification and its implementation, and (5) Building Lifecycle Interoperable Software (BLIS) implementation views to the IFC specification. My research is also part of CIFE's Virtual Design and Construction (VDC) framework. Objects in the requirements model specification represent desired product form in the Product-Organization-Process (POP) ontology.

I addressed the challenges by formalizing a requirements model specification which can be linked to a building-product-model-based design model of the project. My research consisted of four phases: (1) analysis of client requirements, (2) development of a requirements model specification and its links to the IFC specification, (3) extension of the BLIS view for IFC implementation, and (4) validation of the requirements model specification.

Based on the requirements analysis, the number of possible requirements is high but only a few of them are used on most projects. However, the linkage of direct and indirect requirements to the design model is complicated and cannot be defined on a project by project basis only. Thus, my requirements model specification is based on an inclusive approach; all relevant requirements which were identified in my research are included in the specification, and each requirement object includes the direct and indirect links to the different levels of detail in the design model.

The specification covers 300 requirements in 14 main and 35 sub-categories. It is based on a synthesis of two large, widely used requirements hierarchies, analysis of requirements in five building programs and spatial requirements in the current IFC specifications. These requirements are organized in the specification into 7 main-level and 30 sub-level requirements objects which have direct links to 5 levels of detail and 2 systems in the building product model plus indirect links to 4 levels of detail and 12 systems. The size and complexity of the specification can be managed by a good user-interface design, which is one of the proposed future research topics.

The main scientific contribution of my research is this requirements model specification, based on the following main concepts: (1) division of a project’s data set into four main models; requirements, design, production, and maintenance models, (2) requirements related to the different levels of details in building product models, and (3) direct and indirect requirements. Although the detailed requirements relate mainly to the architectural design, the main concepts of the specification are not domain-specific and apply to a general interface between requirements and building product models. The same link mechanism which is used between objects in the requirements and design models applies also between objects in different design and production models.

My specification defines the structure of the requirements model. Its purpose is to serve as the basis for software development. For AEC professionals it is useful only if implemented into software products. Thus, the main practical implications of my work are that (1) the requirements model specification enables implementation of requirements management applications linked to building product models, and that (2) the use of such applications can improve the management of detailed client requirements in the building process. In addition, I propose some improvements in the current IFC specifications.

One of the goals for my research was to create a basis and a wide framework for future research topics in this area. Thus, the documentation is inclusive rather than exclusive. In general the future research topics can be divided into two categories. (1) Research which expands the requirements model specification, such as the relation between high-level strategic owner requirements and detailed end-user requirements, requirements for other design domains, other parts of the process, and different building types. (2) Research which relates to the use of the requirements model, such as implementation of requirements management applications using model server technology, utilization of requirements history, automated verification of design, and semi-automated design software.

Acknowledgements

"The most beautiful thing we can experience is the mysterious. It is the source of all true art and science. He to whom this emotion is a stranger, who can no longer pause to wonder and stand rapt in awe, is as good as dead: his eyes are closed." – Albert Einstein, What I believe.

The above quote may seem strange – even paradoxical – in the context of my thesis work, which focuses on technical information management issues in a creative process – architectural design. However, this attitude has always been one of the leading themes in my life. I did not learn it from Einstein, but from my parents, **Marja-Terttu** and **Into**, to whom I am grateful for the ability to keep my eyes and mind open to "pause to wonder and stand rapt in awe." Unfortunately my father did not live long enough to see my doctoral thesis.

This openness was the basic reason why I undertook the challenge to complete my Ph.D. at Stanford, but there are many people and organizations I owe for this great possibility:

The first and most important person in this process, and my whole life, is my wife, **Eeva**, who has never complained about my work, although there would often have been reason to do so because of my long working hours and absence from home – sometimes even when I am physically present. She also accepted the move away from our daughters and grandchildren for over two years, which was the price we had to pay for this opportunity. Our daughters, **Katja**, **Tiina** and **Suvi** made this decision easier by their supportive attitude and visits to California.

The key people for my Ph.D. project are Dr. **Ari Ahonen**, who gave me the whole idea of the project and made me believe that it was possible, and Professor **Martin Fischer**, who accepted me into his world-class Ph.D. group and supported me from the very first ideas to the final completion as my Principal Adviser. It has been a great pleasure to have such an adviser and friend in this process. I am also grateful for the support of my other Reading Committee members, Professor **Boyd Paulson** and **Dr. Vladimir Bazjanac**, who were willing to share their limited and valuable time with me.

I have also received invaluable insights and comments in my thesis work from a group of friends, who are also world-class experts in this research area: **Jiri Hietanen**, **Kari Karstila, Patrick Houbaux**, **Richard See, Robin Drogemuller** and **Yoshinobu Adachi**. I want to emphasize especially the role and help of Jiri, Kari and Patrick in the solution for the link between the requirements and design models.

As in any university, the Ph.D. process at Stanford includes many other studies, not just the thesis research. In that process I have got great experiences which also affected my thesis work by teaching me methods and new ways to see and understand research. My most important teacher at Stanford has been Professor **Raymond Levitt**, who taught me a totally new area of interest, organizational research and modeling, in his classes and has been my main teacher outside of my thesis work. Another great teacher for me was Professor **Terry Winograd**, whose class helped me to clarify my thoughts and gave some justification to the "philosophy" part in the Ph.D. title, and who acted as the Chairman in my Oral Examination.

I have also enjoyed working with all my fellow students and other people at Stanford who made me feel part of the group immediately when I arrived. It is not possible to list all the names, but I want to mention **John Chachere**, **John Haymaker**, **Calvin Kam**, **John Taylor** and **Xiaoshan Pan**, with whom I have had many interesting discussions during these two years. Unfortunately our tight schedules did not allow more interaction during my stay in California.

Along the way several other people also influenced my path towards my Ph.D. Again, it is not possible to mention all of them, but I want to mention some key people: **Tapio** and **Tiina Koivu**, who made it easy for us to move to California; **Bo-Christer Björk**, who made me interested in international research; **Matti Hannus**, who got me to join VTT; **Mika Lautanala** and **Reijo Kangas** who managed the Vera Technology Program with me and helped me to create my relations with Stanford; **Jukka Pekkanen** who gave me important support through the Confederation of Finnish Construction Industries RT and helped me to start the project in 2002. And of course, the Steering Committee members of my project: **Ilkka Romo**, Bo-Christer Björk, **Auli Karjalainen**, **Markku Kaskimies**, Tapio Koivu, **Olli Nummelin**, **Riitta Takanen**, **Juha Tammivuori**, **Leo Torvikoski**, and **Eija Virtasalo.**

The research has been funded by Tekes (Technology Agency of Finland), VTT (Technical Research Center of Finland), Confederation of Finnish Construction Industries RT, NCC Oy, Pöyry Oy, Skanska Oy, and YIT Rakennus Oy.

And last, though maybe not least, I want to mention an unusual contributor: the nature of California, which has given my wife and me much joy in spite my busy schedule, and helped us, not only to endure the work, but to really enjoy staying in Stanford and make this process into one of the most memorable experiences of our lives.

History of the "PREMISS" Name

My Ph.D. research project was named "PREMISS" based on a loose abbreviation of its original name (Product model extension for REquirements Management InterfaceS) and this name is used in this document to identify some parts to the project, such as "PREMISS Database" or "PREMISS Specification." PREMISS was selected because it is an old British form of the word "premise" and thus fits well to describe the main topic of my research – client requirements for a building project.

Table of Contents

List of Tables

Table of Figures

1 Introduction

1.1 Problem Description

The problem of *Requirements Management*[i] through the design and construction process is familiar to me from my own 24 years of design experience as an architect. Between 1972–1996 I was Project Manager for 36 major building projects, including several university buildings, cultural centers, municipal halls and other types of buildings, in one of Finland's leading architect offices, Arto Sipinen. The time span of the projects from the first proposal to the delivery of the building varied from two to almost nine years, and the variety of building types demonstrated to me many different aspects of the *Requirements Management* problem in real projects.

A *Building Program* specifying the project's goals and *Requirements* for all *Spaces* is the typical *Client Requirements* documentation in building projects, though there are also several other methods to capture *Client Requirements*. Regardless of the capturing method, the *Requirements*, depending on the project type, consist of more or less detailed information about the required *Properties*: net area, activities, connections to other *Spaces*, security, appropriate or desired materials, and conditions, such as daylight, lighting, temperature, and sound level. Many *Requirements* also "cascade," i.e., create *Indirect Requirements* for building elements bounding the *Space* and systems serving the *Space*. Moreover, an important part of the design process is that some *Requirements* can be in conflict; the *Project Team* must often prioritize and make trade-offs between different *Requirements*, which creates the need to update the *Requirements*, and thus, manage and document the changes to the *Requirements* and the design solution.

---

[i] Definitions of all terms formatted in *Italic* are in Appendix A

In practice several factors make it virtually impossible for all participants to know and remember all relevant *Requirements* and, especially, their relationships to each other and to the design solutions. The main reasons for this argument are:

- The amount and complexity of project information,
- The duration of projects,
- The need for designers to work simultaneously on many projects,
- Changing stakeholders in different project phases, and
- Shifting design focus, e.g., moving from overall problem solving to detailed technical solutions.

1.1.1 AEC Process

The Stanford project guidelines, referred to as "Heartbeat" (Figure 1), represent a typical description of the design and construction process. Though it is basically correct, it easily creates an often used, but false, image of a sequential process, where the *Requirements* are set in the programming phase, and design simply solves the needs documented in the programming phase (Figure 2).

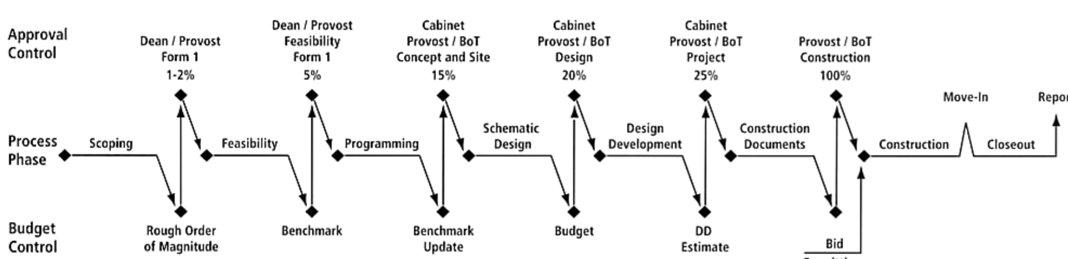

Figure 1: "Heartbeat," the project delivery process at Stanford [*Stanford 2001* [1]]

However, this is not the case. As Daniel Fällman wrote: *"The building design is a deeply iterative process – constant dialog between ideas, analysis, synthesis, and evaluation. It is indeed as much problem setting as problem solving"* [*Fällman, 2003* [2]]. The provided design solutions also affect *Client* expectations, thus causing evolution of the *Requirements*. The iterative nature of the design process is clear to any experienced person working in the AEC industry, but the current *Requirements Management* methods reflect the simplified sequential representation (Figure 2); the *Requirements* are not updated in the process.

Figure 2: Often used, but false sequential process illustration

Though the intensity of *Requirements* definition and design activities, and the character of the changes, are different in different stages of the process, I argue that the process should be described as partly parallel activities, including *Requirements Management* through the whole process and several stages where local authorities check if the design and construction meet the regulations. Inside this parallel process the progress on the detailed level is a “spiral of iterations”: almost like a barbed wire entanglement (Figure 3).

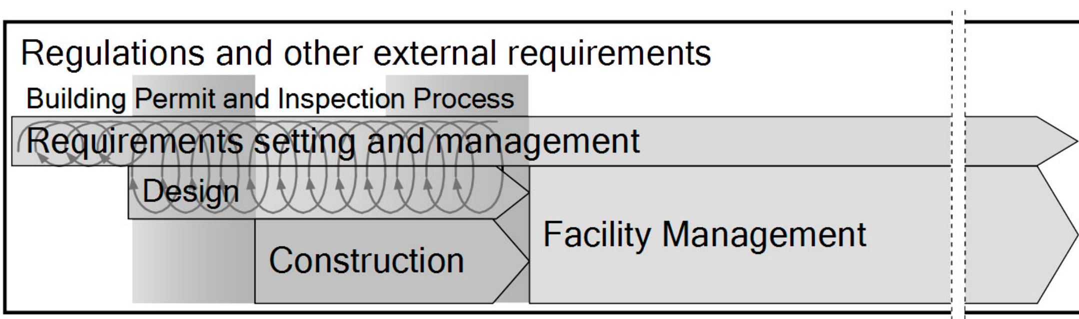

Figure 3: Parallel process view

The iterative nature of the design process and the usually large number of changes during the process increase the complexity of the problem. The *Project Team* has to make rapid decisions on how to solve a specific issue, and it is often difficult to notice all interdependencies. Thus, a solution which meets one *Requirement* can have a significant negative effect on another crucial *Requirement*. One trivial example of this is accessibility vs. access control; optimizing the accessibility to the various *Spaces* in a building is in contradiction with access control, which demands as few access points and alternative routes as possible. My observation is that the current process could improve significantly if:

- The *Project Team* could manage and update evolving *Requirements*, and
- The designers could easily find the *Requirements* related to their on-going task.

A logical solution is a data interface, a link between the *Requirements* and the design solutions, which more effectively connects the *Requirements* to the design process. A link between *Requirements* and *Design Objects* can help designers to understand the interaction between *Requirements* and design solutions better. It also helps the project managers and *Clients* to manage the *Requirements* and to evaluate the design solutions compared to *Requirements* (Figure 4).

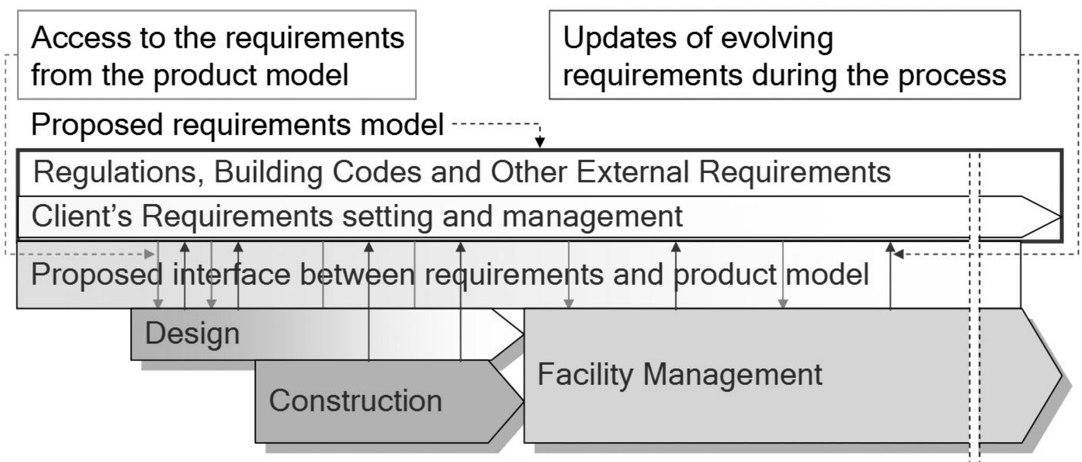

Figure 4: Interface supporting interaction between *Requirements* and design solutions using linkage between *Requirements Model* and existing *Building-Product-Model*-based design, construction and facility management software.

1.1.2 Shifting Focus

After conceptual design, *Requirements Documentation* is usually not used actively in the current process (Section 1.2.2.2, A2), and often the evolving *Requirements* are not even communicated to the whole *Project Team* [*Kagioglou et al., 1998* [3]]. Thus, the changes are compared to, and decisions are made based on, the previous design solutions. Current design tools do not support recording of *Client Requirements* or designers' intent in the documents. Thus, the people deciding on the changes do not always even know the original intent, and the solution can "shift away" from the original goal (Figure 5) without actual decisions to change the goal or an understanding of the contradiction between the proposed design and project goals.

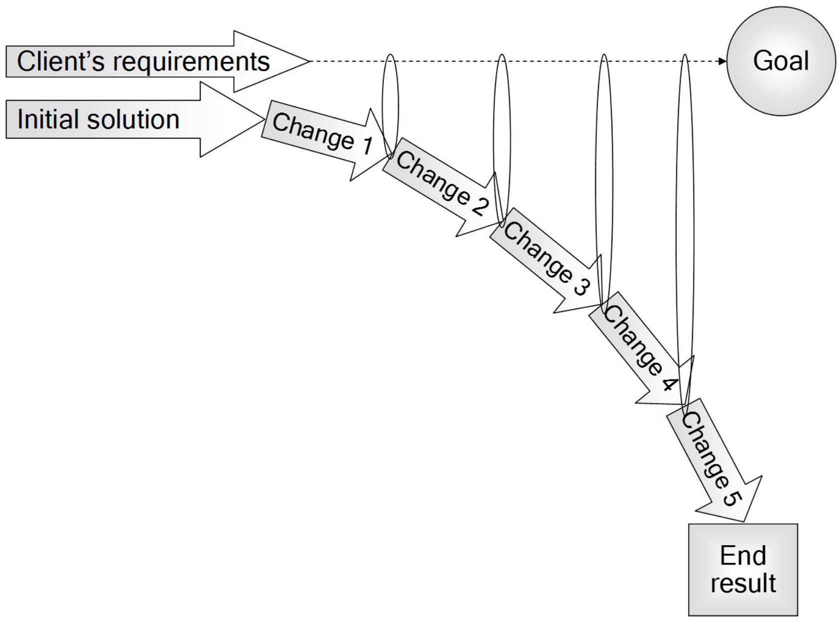

Figure 5: Shifting away from the goal

My observation, supported by interviews and discussions with many industry experts [*Discussion and interviews 2002–2003* [4]], is that to some extent this happens on most projects. This does not mean that most buildings are badly designed or that they do not meet their overall purpose. However, I argue (1) that they often miss some *Properties* which the end-users might have preferred and (2) that the changes of *Requirements* are not well documented. This happens because the design tools do not support such documentation, and the design process includes many trade-offs between different *Requirements*. Therefore, I suggest that the changes should be based on conscious decisions to adjust (1) solutions (Figure 6), (2) *Requirements* (Figure 7) or (3) in many cases both, and that (4) the approved updates in the *Requirements* should be recorded so that they can be checked and compared with the final building afterwards.

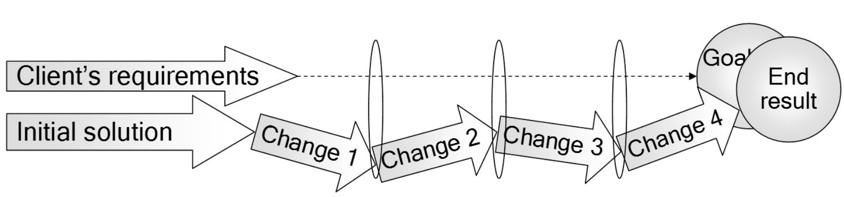

Figure 6: Adjusting design solutions

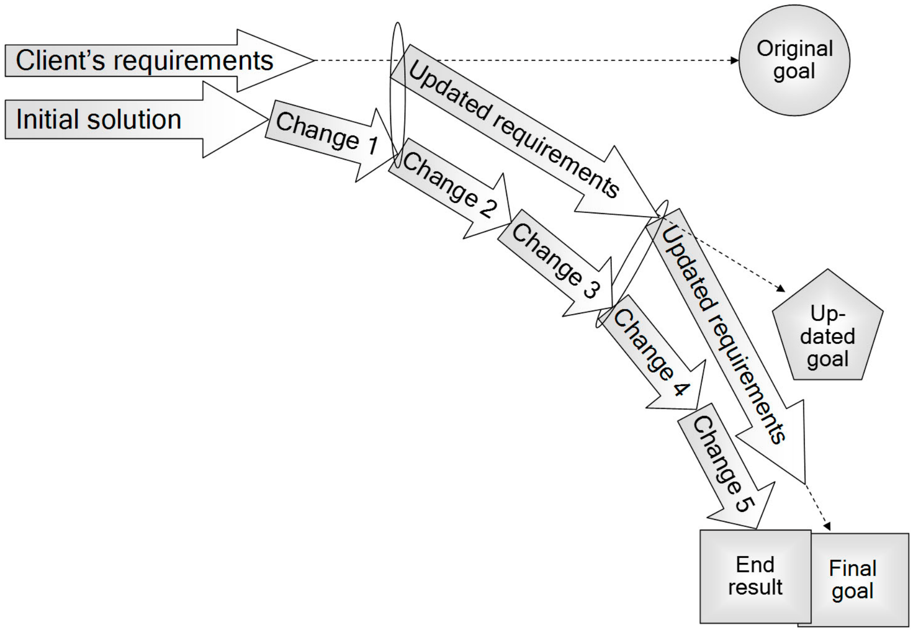

Figure 7: Adjusting *Requirements*

1.1.3 Main Problems

The main problems I have identified are:

1.1.3.1 No connection between Requirements and design documents

The current design tools do not support documentation of the reasons behind the design solutions. As described earlier, *Requirements Documentation* is often used actively only in the early design stages (Section 1.2.2.2). Later in the process the changes are made based on the previous solution. This leads to the two main problems described above: The design can shift away from the original goal, and the evolving *Requirements* are not updated in the *Requirements Documentation*.

1.1.3.2 The impact of project personnel changes and project duration

In the current process *Requirements Changes* are not updated coherently and in an easily accessible format. In the best case, they are stored in the meeting minutes, but in actuality they are often stored only in the minds of the *Project Team* as tacit and implicit knowledge (Section 1.2.2). Even if the changes are documented in the minutes, they are scattered and difficult to find, especially for people who do not know exactly what to look for and where to find it (Section 1.2.2.1). This situation leads to significant loss of *Requirements Knowledge* if some key persons leave the *Project Team* (Section 1.2.2.2). Long project duration has a similar impact because of personnel changes and human difficulty in remembering details.

1.1.3.3 Impact of "middle-men" in the process

The actual end-users are not always closely involved in the design and construction process. Thus, they may lack the means to follow and control what happens to their demands in the process (Section 7.1.2.1). This emphasizes the need to have *Requirements* actively linked to the process, because it would help (1) the designers find the relevant *Requirements* more easily themselves and (2) end-users compare their *Requirements* to the design. In addition, because of described inadequate documentation of the *Requirements Changes*, it is difficult to find the approved *Requirements Changes*, and the end-users may compare the building to the original, outdated *Requirements*.

1.1.3.4 Direct and Indirect Requirements

Most *Client Requirements* are related to *Spaces* and in current practice these are recorded in the *Space Program*. However, these *Direct Requirements* often lead to *Indirect Requirements* for the *Bounding Elements* and technical systems. *Bounding Elements*, e.g., walls, windows, doors and slabs, can have *Requirements*, such as sound or thermal insulation, security, and load bearing *Requirements*, which come from the *Space Requirements*. Likewise, technical systems, such as mechanical, electrical and plumbing (MEP) systems or information and communication networks, are affected by the *Space Requirements*. These *Indirect Requirements* can be difficult to notice or remember, because the detailed design related to them usually occurs late in the process and is often done by people who were not involved in the early stages when these *Requirements* were defined, and the design documentation does not include *Requirements Documentation* or these *Requirements* relations.

1.1.4 Building Product Models as an Enabling Technology

My observation (Section 1.2) is that the effect of these factors could decrease if the *Requirements* would be **easily available** and **actively linked** to the design solutions. Another important part of a good solution is the **appropriate level of detail**, i.e., finding only the relevant information for the on-going design task from the project data. This need also creates the demand to **link the *Direct and Indirect Requirements***, so that, for example, the wall *Requirements* caused by the related *Space Requirements* can be easily found.

One prerequisite for this is a meaningful semantic content in the design documents. Traditional documents based on drawings cannot provide sufficient structure for a connection between *Requirements* and design solutions (Section 3.1). However, emerging *Building-Product-Model*-based design software has changed the situation, and together with the existing, structured *Requirements Documentation* in the beginning of the process, provides a potentially usable point of departure. **The key limitation is the lack of a theory to link the *Requirements* to the *Building-Product-Model*-based design systems.** The key elements missing are:

- Lack of a formal *Specification* of the link between *Requirements* and *Building Product Model*, and
- Lack of a formal *Specification* to derive the *Indirect Requirements* for *Bounding Elements* and technical systems from the *Direct Requirements*.

1.1.5 POP, FFB and VDC Framework

The Center for Integrated Facility Engineering (CIFE) at Stanford University has introduced the concepts of Product-Organization-Process (POP), and is using Form-Function-Behavior (FFB) modeling for Virtual Design and Construction (VDC). This framework enables integration of different *Models*, which are often seen as separate entities. Each of the POP elements consists of all three FFB elements, which are divided into three sub-elements: Desired, Predicted and Observed (Figure 8). This structure provides a conceptual framework for a project ontology connecting the different views to the information. My *Requirements Model* represents the Desired Product Form connected to the Predicted Product Form (*Design Model*) in the POP ontology.

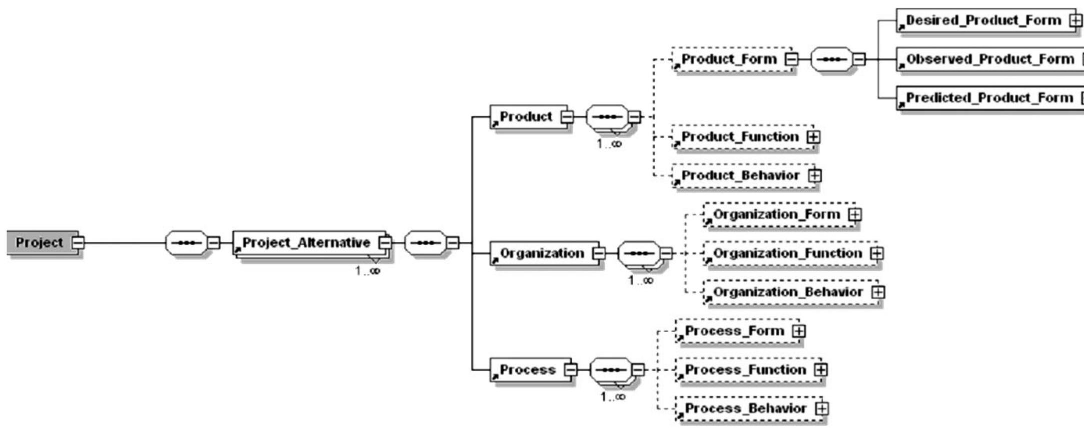

Figure 8: POP Ontology [*Garcia et al., 2003* [5]]

1.2 Motivating Case Examples

To test the existing problems and possible solutions I studied the *Building Programs* of two real world projects, implemented some test databases in MS Access and entered the project information into the database. The two projects are the ICL Headquarters project in Helsinki, built in 1994–1996, and the Lucas Center Expansion at Stanford University, which was under construction when the study was made in summer 2003. These two projects were selected to test the generality of the problem and possible solution, because their characteristics are very different. The ICL Headquarters is a large office building consisting mainly of standard office *Rooms*, but also including some special *Spaces* and *Requirements*. The Lucas Center Expansion is a small special laboratory consisting mainly of unique *Spaces* with very little repetition.

In the test cases my research concentrated only on *Client Requirements* related to the *Spaces*. *External Requirements* or *Requirements* related to other issues, such as project or building, were not in the scope at this stage.

1.2.1 ICL Headquarters, Helsinki

The case which originally suggested the idea for the potential solution is the ICL Headquarters project designed and built in Helsinki, Finland from April 1994 to June 1996. The project is a large office building for approximately 1,000 employees, including *Space* for an extensive computer service and delivery center. The net area in the *Building Program* is around 20,000 m^2, consisting of about 800 *Spaces*.

The PM defined the project's *Building Program* entirely in MS Excel based on a simple *Space Type* classification. In the design phase, I linked the MS Excel data to AutoCAD, where my application automatically created *Spaces* using simple objects consisting of polylines and extended data to link the *Spaces* in the drawings and the area *Requirements* in the MS Excel spreadsheet (Figure 9). During the entire design process, I exported the actual areas from drawings into MS Excel and the PM and *Client* compared *Target Values* to the design solutions almost in real-time, at least once a week. However, we did not link and observe *Requirements* other than area using this method.

The ICL Headquarters' *Building Program* was one document. The *Project Team* constantly compared the required areas to the actual design solutions and updated the *Requirements* file during the design process. The *Requirements Documentation* with respect to required *Space* areas was coherent. The only identified problem related to the structure used in the document: The PM entered all classification codes and *Requirements* manually in each cell in the MS Excel spreadsheet, which created the possibility for incoherent content and made updates more laborious. Use of references to one data source, i.e., a simple *Cascading* structure, would have prevented this problem.

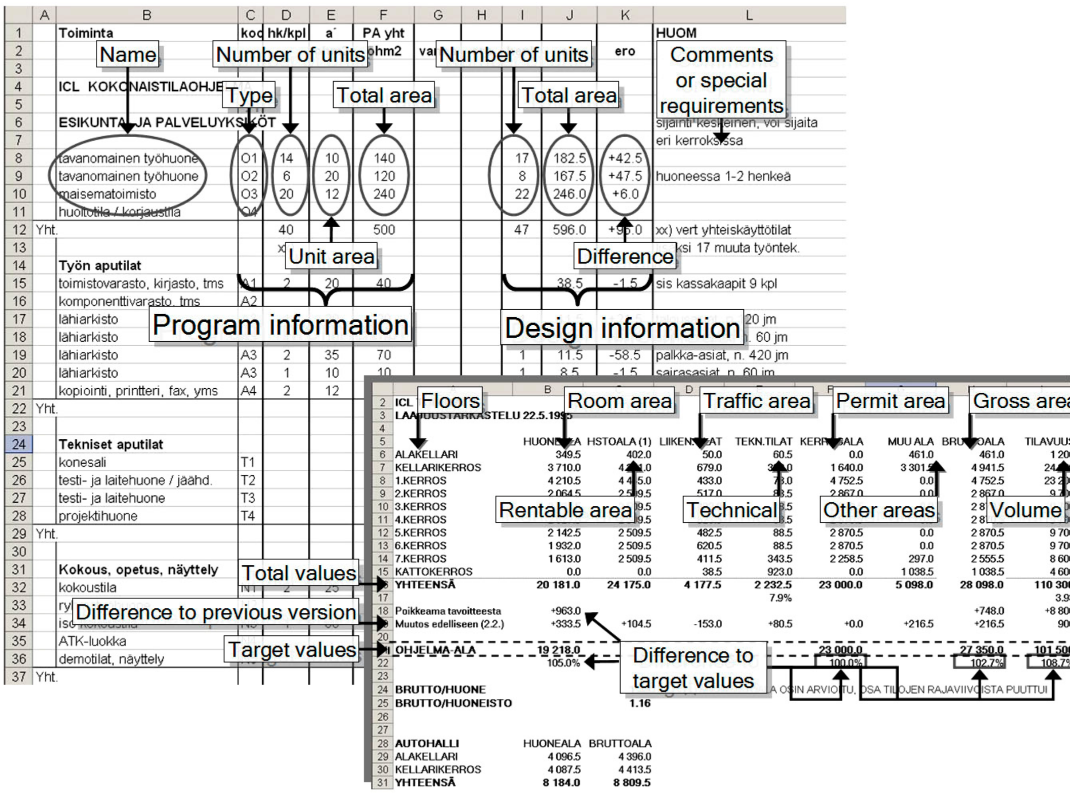

Figure 9: Examples of ICL Headquarters spreadsheets

1.2.2 Lucas Center Expansion, Stanford University

The structure and size of the *Building Program* of the Lucas Center Expansion project is very different from the ICL Headquarters. The Lucas Center Expansion (LCE) is a small special laboratory for the Cyclotron and 7T magnetron laboratories for Stanford University. The net area is 480 m^2, including 23 *Spaces* in the first *Space Program* (February 1st, 2002), and 1,300 m^2, including 43 *Spaces*, in the latest available documents (November 26th, 2002). The available project documentation consists of a set of design sketches, drawings and MS Excel spreadsheets of different project stages, the architect's *Requirements* database in Claris Filemaker, meeting minutes, and technical specifications. The project was in the early construction stages when I did the study (November 2003), and although the final project documentation was available, it was not relevant for my research, because it contained only design solutions and no updates of the *Requirements*.

LCE's Project Manager for Stanford University (PM) and MBT Architecture's Project Architect (PA) provided some insight into the project. The basic conclusion based on these interviews is that Stanford's projects are generally well-managed and have clearly defined processes for different stages. However, as is typical in the AEC industry, the *Requirements Capturing* process is somewhat fuzzy and based strongly on meetings, where end-users and the *Project Team* interact to find solutions to specific problems. The *Project Team* records decisions in the meeting minutes, and the architect and PM document the *Space* areas of each design stage in MS Excel spreadsheets. The reasoning behind the changes and proposed solution becomes tacit knowledge and is "stored" only in the minds of the participants.

1.2.2.1 Interview with the Project Manager

On this specific project, the Project Manager recalled two major issues where the necessary *Requirements Information* was not available, causing problems to the design process:

- In the first sketches the cyclotron and 7T laboratories were co-located. The reasoning for the design solution was to combine the heavy MEP systems and their spatial needs and separate them from the less demanding office and laboratory *Spaces*. The whole *Project Team* was satisfied with the solution until the equipments' technical information from the manufacturer showed that the *Spaces* must be as far apart from each other as possible, because of the electric and magnetic interference. This led to a completely new design starting essentially from scratch. This could have been avoided if the necessary information had been available in the *Space Requirements*.
- The other major issue was the number of fume hoods in one laboratory. The original demand for fumes was 6, then 8 and finally 12. However, at that stage 12 fume hoods were not possible because of the increasing spatial need for ducts. After some lengthy discussion, the problem was solved by having eight fume hoods of the original type and four additional bio-safe fume hoods, which circulate the air instead of exhausting it.

Both cases illustrate (1) the need for detailed *Requirements Information* in the early design stages and (2) the connection between *Requirements*, spatial solutions and technical systems. The first example illustrates "inverse adjacency," i.e., the need to know which *Spaces* must be far apart. The second example illustrates evolving *Client Requirements*. However, these examples are just anecdotal information and based entirely on the PM's memory. The design history or the actual *Client Requirements* were not recorded in the documents in a way which would enable tracking of changes.

When asked if the PM could find the *Requirements* or design criteria to a specific *Space* or building element, the PM's answer was a direct and emphatic *"No."* He said that it would be a very laborious task to go through the meeting minutes trying to find the *Requirements* for any specific *Space* or building element. An excellent practical example of this problem is a quote from the PM's secretary's email: "*I am attaching samples of programming documents per your request, but I am having a hard time finding MBT's design criteria.*" This illustrates excellently that not only detailed *Requirements*, but even the high-level documents are hard to find in the current process. This is of course a problem which can be solved partly just by using existing document management systems, but document-based systems cannot provide formal linking mechanisms to the information content with the necessary structure, as a *Building-Product-Model*-based environment can do (Section 3.1).

The PM's opinions about the identified problem were:

- "*The problem of Requirements Management is real. We have no mechanisms to record, manage and track changes in Requirements and especially the reasons behind them.*"
- "*Lots of information is totally 'human dependent.' Thus, keeping the same people in the process is crucial, and for Stanford University the preferred method is to work with the same people in several successive projects.*"
- *"QFD* (Quality Function Deployment) *is an interesting method."* (The PM had just read an article on QFD in the Journal of Construction Engineering and Management [6]) The PM felt that "*the main reason that it is not widely*

used in the construction industry is its separate software environment; there are too many software tools in the process already. If the Requirements Management solution would be integrated on the same platform which is already used in the process, the usability and benefits would be much higher."

1.2.2.2 Interview with the Project Architect

In an interview, the Project Architect gave the following answers:

Q1a: Could you find the answers and how much time would it take if you would have to trace back any specific *Requirements*, such as: "What did the *Client* exactly require for this laboratory? Who set the *Requirements* and when?"

A1a: "*We back up the design documents of every phase. It would take several hours for me to restore the backups, but then we could trace back how the design solutions developed. However, we do not record the actual Requirements Changes. The only documents where this could be found are the meeting minutes, but they do not cover all issues."*

Q1b: Could anyone else find them and how much time would it take?

A1b: *"Even in the best case it would take much more time than for me. In the worst case they could never find the right answers."*

Q2: Can you recall a concrete situation where you spent much time searching for relevant *Requirements* or where you worked with the wrong assumptions?

A2: *"Not on this project, because we have been involved from the beginning. But it happens often if the project personnel change, because a large amount of the information is just in the head of the Project Architect."*

Q3: Do you use any other methods to communicate the *Client Requirements* to the other participants than telling what you know?

A3: *"Only in the programming phase, where we use our database tool to record some Requirements, but even then not all the details. In the design-development phase there is too much work and information. We don't update our requirements* *database after the programming phase. The information in later phases is our design recorded in the drawings and other design documents."*

My review of the architect's *Requirements* database supports the statement that the architect did not record all the details (Section 1.2.2.3).

1.2.2.3 Detailed Findings and Problems for the LCE project

Based on my own experience plus interviews and discussions with industry experts [*Discussion and interviews 2002–2003* [7]], information management problems increase when the project size and complexity increases. The Lucas Center Expansion is a small project, and the amount of information in the *Requirements Documentation* was also relatively small, but in spite of this the information was incomplete and contained several inconsistencies, which demonstrates that these problems occur on both small and large projects.

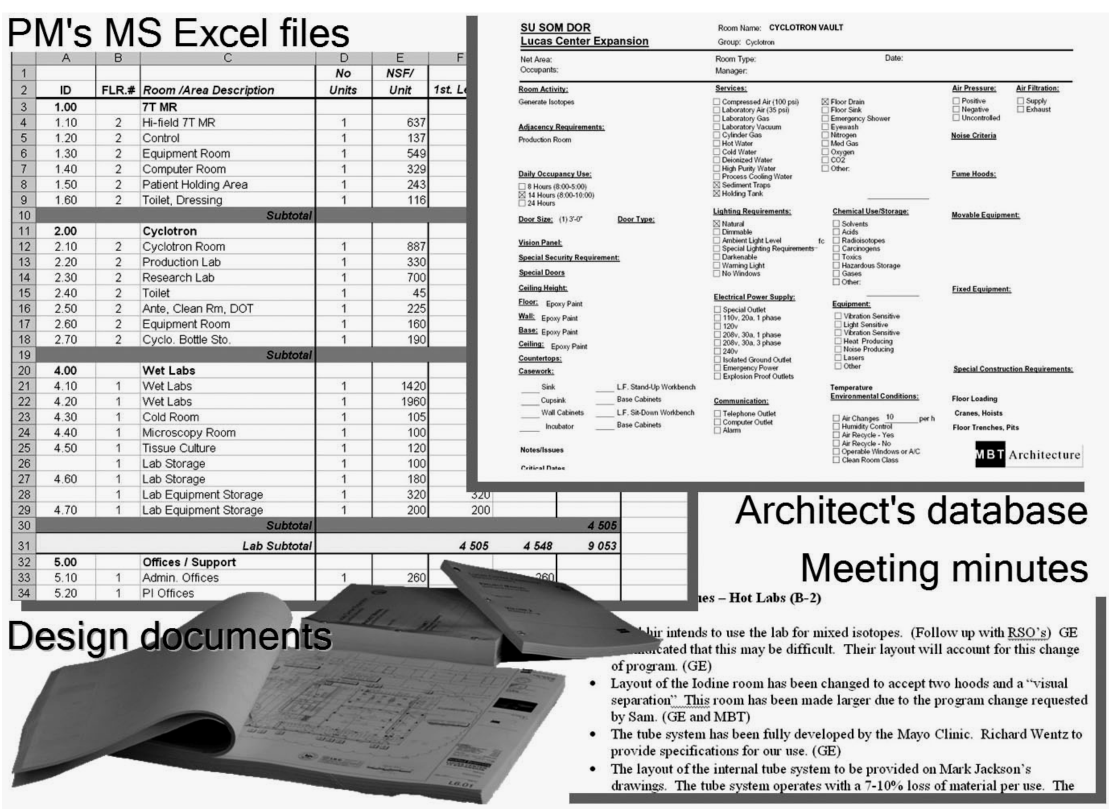

Figure 10: Examples of LCE *Requirements* documents

The main problems were related to the use of two different sources of information, the PM's MS Excel spreadsheets and the PA's *Requirements* database, and their different and partly inconsistent content (Figure 10). In addition, the MS Excel sheets for different stages were in separate files and the development history or reasons were not recorded even for large changes. For example, the changes in the net area in different stages were very large (242%–1076%, Table 1). In fact, only the first version (February 1st, 2002) presents the actual *Client Requirements*; later versions instead summarize the design status in different stages.

More complex technical specifications, such as MEP descriptions, have no relation to the PM's or PA's *Requirements Documentation*. "MEP Utility Planning and System Description VI, March 05, 2002" document specifies clearly the *Requirements* for the two main *Spaces*, 7T MR and Cyclotron, but the required *Properties* for the other *Spaces* are not easy to interpret. However, because the actual MEP systems are out of the scope of my research (Section 1.3), this was not studied in detail. It indicates, though, that the *Requirements Management* problem is also related to other design areas.

Table 1: Changes of the *Building Program* summary of Lucas Center Expansion

	Feb 01 2002	*Apr 17 2002*	*Change*	*Sep 11 2002*	*Change*	*Oct 18 2002*	*Change*	*Nov 26 2002*	*Change*	*To the original*
7T MR	2 380	1 680	*71%*	1 736	*103%*	1 802	*104%*	2 011	*112%*	*84%*
Cyclotron	1 050	1 034	*98%*	997	*96%*	1 005	*101%*	2 536	*252%*	*242%*
Hot Lab	1 020	690	*68%*	1 288	*187%*	1 120	*87%*		*0%*	*0%*
Wet Labs		3 550	*1076%*	3 252	*92%*	4 326	*133%*	4 505	*104%*	*442%*
Lab subtotal	**4 450**	**6 954**	*156%*	**7 273**	*105%*	**8 253**	*113%*	**9 052**	*110%*	*203%*
Admin&Support	750	750	*100%*	750	*100%*	2 856	*381%*	4 926	*172%*	*657%*
Total	**5 200**	**7 704**	*148%*	**8 023**	*104%*	**11 109**	*138%*	**13 978**	*126%*	*269%*
Technical spaces		771		1 150	*149%*	1 162	*101%*	1 196	*103%*	*155%*
Unassigned		5 195		5 234	*101%*	5 895	*113%*	5 895	*100%*	*113%*
Gross area	**10 400**	**13 670**	*131%*	**14 407**	*105%*	**18 166**	*126%*	**21 069**	*116%*	*203%*
Efficiency, real	*50%*	*56%*		*56%*		*61%*		*66%*		
To the original	***100%***	***131%***		***139%***		***175%***		***203%***		

Detailed list of discovered problems and contradictions:

PM’s MS Excel spreadsheet:

- The information content is just ID, name, area and *Required Location* (floor) ⇨ the file covers only area *Requirements;* all other *Requirements* are found only in the architect’s database. In fact, as mentioned before, even the area information reflects the design status rather than the *Client Requirements*.
- The same ID (5.10) is used for two different *Spaces* -> Identification based on Information and Communication Technology (ICT) is impossible.
- Three *Spaces* do not have IDs at all ⇨ ICT-based identification is impossible.
- In some cases a manual summary of *Spaces* per floor exists ⇨ summary and individual areas do not match.
- The original area *Requirements* are not stored ⇨ changes are difficult to follow.

PA’s *Requirements* database:

- Only 1/3 of the *Spaces* are in the database (13 of the 43 *Rooms* in the PM’s MS Excel spreadsheet).
- Area *Requirements* are not included in the database.
- The IDs are often different or missing, and the *Space* names are often different from the names in the PM’s MS Excel spreadsheet ⇨ ICT-based identification is impossible, and in some cases identification of *Spaces* is impossible for people who do not have the tacit project knowledge:
    - There are two different wet labs, but they do not have IDs ⇨ it is impossible to know which is which in the other documents.
    - Hot labs are missing from the MS Excel file.
    - In some cases there are adjacency references to *Spaces* which do not exist in the documentation.

- There are several, slightly different ways to document the same issues:
    - *Space Types*
    - Activities
    - Materials
    - Casework and equipment
- There are some obvious mistakes in the *Requirements*:
    - The natural light *Requirement* is sometimes in unnecessary (storage rooms) or absolutely impossible places (cyclotron room). The natural light *Requirement* appears to be a default value in the database, and as a consequence, it is listed for these *Spaces* as well.
    - A 1' door in the Hot Lab/Research room.
    - A maximum noise level *Requirement* for a storage room.

1.2.3 Conclusions from Both Test Cases

Based on my own experience and several interviews and discussions with industry experts [*Discussion and interviews 2002–2003* [8]], LCE project's *Requirements Documentation* and process are typical examples of practices on current construction projects. Different parts of the *Requirements* are recorded in several documents, and there is no comprehensive document containing all needed information. In addition, the names and IDs for the *Spaces* are often ambiguous, and similar *Requirements* are formulated in different ways. This makes it difficult to connect *Requirements* to the correct *Space* even manually, and any use of ICT to manage the relations between the *Requirements* and design solutions is impossible.

The main problem categories in the *Requirements Documentation* for the LCE project were:

- Lacking or different identifications of the *Spaces*,
- Contradictions in the content of different documents,
- Incoherent way to describe the same *Requirements*,
- Wrong or missing information,

- Instead of actual spatial *Requirements*, the documents recorded the areas of the *Spaces* in the design solution, and
- Documents specifying detailed technical *Requirements* had no relation to the *Space*-related *Requirements Documentation*.

Though many of the mistakes in the LCE project were small, and probably caused few, if any, real problems to the people who have been involved in the project all the time, they are a clear indication of the general *Requirements Management* problem in the current process. To anyone who joins the project later, it is very difficult and time-consuming and sometimes impossible to find out which *Requirements* are correct and still relevant. Furthermore, someone who wonders about the growth in the size of the project will have great difficulty finding an answer in the project documents.

Though I linked only the required area information with the design solution in the ICL Headquarters project, the link provided significant benefits in the project management (Section 7.1.1). In addition, despite the simple approach taken in the ICL project to link only the *Requirements* and the design information to compare required and designed areas, the coherent *Requirements Information* suggests that a link between *Requirements* and design tools and the constant use of *Requirements Information* in the process could improve *Requirements Management*.

1.3 Definition of the Research Scope

My research focused on the *Requirements Model* and its connection to the architectural *Design Model*. The *Requirements* structure is based on traditional *Building Programs*. The *Direct Requirements* are limited to architectural design. The derivation of *Indirect Requirements* to the *Bounding Elements*, e.g., walls, windows and doors, from these *Direct Requirements* is within the scope of my research. Cost *Requirements* on the project level are in the scope, but the detailed cost on the item level are not.

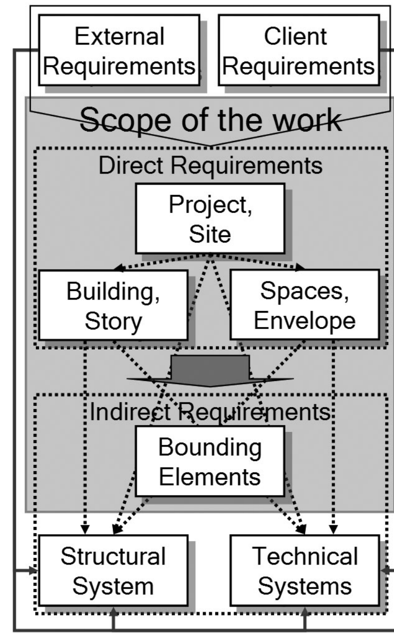

Figure 11: Scope of the work

Project types in my research were limited to office and laboratory buildings. Other building types were not in the scope.

Detailed *Requirements* for other design areas, such as MEP and structural engineering, are not in the scope of the research, but the connection from architectural design to these design areas is addressed. However, only the need for such a connection from the architectural design was analyzed and shown; the detailed content of these *Requirements* was not in the scope of the research.

Because many *Client Requirements* are based on descriptions, not on technical values, automated comparison of the *Requirements and Design Models* was out of the scope, though I can assume that the proposed system would enable automated checking of how well a design solution meets the *Requirements*, at least to some extent.

2 Research Questions and Methods

My work addresses the following research questions:

Question 1 (RQ1): How can a *Requirements Model Specification* for *Client Requirements* in a building project be formalized?

The method to answer RQ1 consists of three steps:

- Detailed analysis of *Client Requirements* and principal solutions for *External Requirements* (Section 2.1),
- Development of a *Requirements Model Specification* for these *Requirements* (Section 2.2),
- Validation of the *Requirements Model Specification* (Section 2.4).

Question 2 (RQ2): How can the relation between this *Requirements Model* and *Design Model* be formalized?

The method to answer RQ2 consists of three steps:

- Development of an interface between the proposed *Requirements Model Specification* and current *IFC Specification* (Section 2.2),
- Definition of an expanded view for implementation of the proposed *Requirements Model Specification* and *IFC Specification* (Section 2.3), and
- Validation of the interface and *Implementation View* (Section 2.4).

2.1 Detailed Analysis of Client Requirements

The first stage in answering RQ1 was the analysis of five *Building Programs* to test:

- The generality of the *Client Requirements*, i.e., on which level are the *Client Requirements* the same in different projects?
- Relevant *External Requirements*, i.e., on which level should the *External Requirements* be linked and managed in a project-specific *Requirements Model*?

- Useful level of detail, i.e., which *Requirements* should be in the *Requirements Model Specification*, and which should be project-specific additions?

A detailed analysis of these issues is in Chapter 4, and it is one of the scientific contributions of my research (Chapter 8).

2.2 Development of the Requirements Model Specification

Development of the final *Requirements Model Specification* was based on the analysis described in Section 2.1. In this stage I relate the answer to RQ1 to the generality of the types of projects, i.e., is the *Requirements Model Specification* reasonably useful in the projects which are within the scope of my research?

Chapter 6 documents the developed *Requirements Model Specification*, and it is the main scientific contribution of my research (Chapter 8).

2.3 Expanded View for the Implementation of the IFC Specification

The Building Life Cycle Interoperable Software (BLIS) group has developed the concept of *Implementation Views* to support IFC-based information exchange (Section 3.5), and my research expanded the existing "Client Brief / Space Layout -> Architectural Design" view. I base the content of the expanded view on the *Requirements Model Specification* described in Section 6.3. The expanded view will be the basis for the implementation to link the *Requirements Model* and the *Building-Product-Model*-based software, and it is one of the scientific and practical contributions of my research (Chapter 8).

2.4 Validation of the Requirements Model Specification and Interface to the Building Product Model

The validation criteria for the *Requirements Model Specification* are:

1. **Usefulness**: Does the *Requirements Model Specification* address relevant factors of the identified problem within my research scope and could its implementation into a tool improve the current process?

2. **Generality**: Does the *Requirements Model Specification* cover a reasonable part of the identified problem?
3. **Implementability**: Is the *Requirements Model Specification* possible to implement?

There is no objective method to measure or validate the usefulness or generality of a *Conceptual Model*, such as the *Requirements Model Specification*. Thus the validation must be based on:

- Comparison of the potential *Model* features and problems in real projects.
- Comparison of the *Specification* content and the *Requirements* in real projects.
- Implementability of other *Specifications* based on similar methods.

Chapter 7 documents how the *Requirements Model Specification* and its interface to the *Building Product Model* meet the above three validation criteria.

3 Point of Departure

As stated in Section 1.1.3, there is no theory which would provide the basis to link *Requirements* to a *Building Product Model* representing a design solution. A solution to the above-mentioned problems can build on the following five starting points:

- Design as an information process,
- Existing *Client Requirements Capturing* methods and *Requirements Hierarchies*,
- Lawrence Berkeley National Laboratory's (LBNL) Building Life-Cycle Information Support System (BLISS) and Design Intent tools
- Existing *IFC Specifications* and their implementation, and
- Building Lifecycle Interoperable Software (BLIS) *Implementation Views*.

Design as an information process justifies why the *Requirements* and their management should be linked to the design process. Existing *Client Requirements* provide the basic content for the *Requirements Model*, i.e., what should be linked. LBNL's Design Intent and BLISS tools are a reference for *Requirements Management* in the MEP area. The existing *IFC Specifications* describe what is available in the *Building Product Models*; to which *Requirements* can be linked, and the existing implementations and BLIS *Implementation Views* provide the technical platform; how to establish the link. The existing elements are *Requirements Documentation* and *Building-Product-Model*-based design software; the main limitation is the lack of a method to link these together and handle the relation between *Direct and Indirect Requirements* (Section 1.1.4 and Figure 25).

3.1 Information Processing and Management in Design

The design process contains many elements, and we can analyze it from several viewpoints, such as, art, creativity, problem solving or information processing and management. My research concentrates on the field of information management: how to maintain and use evolving *Client Requirements* in the design process.

This does not mean that I would consider the artistic and creative parts of the architectural design less important; on the contrary, they are the essence of architecture. Information technology can also be an important element in the creative process; for example, Frank Gehry has said, "Much of what I have done in the last decade has been made feasible by our use of CATIA" [*Gehry, 2003* [9]]. However, this part of the process is not within my research scope.

The information processing and management possibilities in design changed dramatically when computer aided design (CAD) replaced traditional hand-drafting. In the paper-based environment, each piece of information was represented in one or several documents and the only possible “links” between these documents were written references. Information technology enables actual links between documents and also between objects which can contain significantly more explicit information about the building elements than their traditional graphical representations on drawings.

I am not trying to formulate a design theory, such as Christopher Alexander’s “Pattern Language” [*Alexander et al., 1977* [10]]. It would be possible to make an application of my *Requirements Model Specification* which would include Pattern Language to describe architectural *Client Requirements*. However, this would require that the designer teach the *Client* to understand this language; based on my experience it is unusual that *Clients* would use such a language to describe their *Requirements*.

My approach more closely resembles Horst Rittel’s “Argumentative Process” [*Rittel and Kunz, 1970* [11]], where the initially unstructured problem area or topic develops through documented arguments to a formal decision. In my opinion the Argumentative Process describes the development of final *Requirements* in the design process well, and my *Requirements Model* helps to connect the *Requirements* with the design topics in a somewhat similar way as Rittel’s Issue-Based Information System (IBIS).

Froese (2002 [12]) describes another valid approach, the design process as an information processing activity: *"All design and management tasks are primarily* *information-based activities; they take certain information as inputs, create new information about the project, and produce some type of information as a result."* In the beginning of the process, the inputs are (1) *Client Requirements*, (2) *External Requirements*, such as site *Requirements*, building codes, and other regulations, and (3) the *Project Team's* knowledge [*Kamara et al., 2003* [13]]. Later in the process the previous design solutions – modified information – are increasingly used as inputs, while the use of *Client Requirements* – original information – decreases (Section 1.2.2.2). As described in Section 1.1.2, incremental changes based on previous solutions without comparison to the original *Requirements* can gradually lead to an end-result which differs significantly from the *Requirements* without conscious decisions to change the scope of the project. This is the key observation behind the *Requirements Management* problem, and the basis for this research. However, there is little research on this problem related to *Building Programs* in the building design process.

Efficient and appropriate information management is crucial for the success of projects [*Best and De Valence, 1999* [14]*, Kamara et al 2003* [15]]. The information processing needs in complex building projects are very high and the increasing demands to fast-track the process make the information management an even more crucial issue [*Eastham, 2002* [16]*, RT 2002* [17]]. The development of ICT has provided significant insight into many of the problems related to information management.

However, many design tools were developed to automate drafting, instead of serving information management. The drawing-based documents are human, not computer interpretable, which sets serious limitations to the reuse and linkage of the information represented in the documents [*Froese, 2002* [18]]. Froese identifies two basically different approaches to the information management problem: Internet-based collaboration, and *Model*-based approaches. Internet-based collaboration is mainly based on electronic versions of the traditional human-readable documents, while the *Model*-based approach is based on a different abstraction of a real building having a defined semantic content which is also computer interpretable.

Another approach to the design as an information process is the Active Design Documents (ADD) concept [*Garcia et al., 1993* [19]], which demonstrated an automated approach to record the design intent in preliminary routine design. The main focus in the ADD research was on designers' decisions, while my research is focusing on the management of *Client Requirements* and the connection between the *Requirements* and design solutions.

My research is based on these two observations:

- The need to manage *Requirements Information* during the design process.
- The possibility of linking *Requirements* to the objects in the *Design Model*.

Because the semantic content of *Building Product Models* enables a meaningful connection between *Requirements* and project, site, building, *Spaces*, building elements and systems, my research builds on the *Model*-based approach; specifically on existing *Building Product Models* (Sections 3.4 and 3.5).

3.2 Requirements Documentation and Hierarchies

3.2.1 Requirements Capturing and Documentation

Requirements Capturing is a wide area, starting from high-level strategic views of real estate portfolios and ending with detailed technical specifications. My research scope covers only a small subset of this area; *Requirements* related to architectural design. These *Requirements* are in practice captured mostly by interviewing *Clients*, owners and end-users of the future building, and they are documented in a *Building Program*.

Some *Requirements* are common to practically all buildings, such as required area, activities in the *Space*, and connections to other *Spaces*. Some *Requirements* are specific only to some building types, such as exact limits for minimum and maximum temperatures and moisture, which are common for laboratories, museums, demanding technical facilities, etc., but not defined for most buildings. I argue that we cannot fully standardize these different types of *Requirements*. Thus, the goal of my research is to identify a reasonable set of *Requirements* within the defined scope and create a framework, *Requirements Model Specification*, which also enables the addition of project-specific *Requirements* in the project's *Requirements Model* (Section 2.1).

Furthermore, the source of *Requirements* is varying, in many cases the original *Client Requirements* are fuzzy, and designers turn them into more accurate *Requirement Descriptions* or *Requirement Attributes* [*Whelton and Ballard, 2003* [20]]. These varying needs make it difficult, if not impossible, to define a perfect method to capture *Requirements* or define their content, i.e., a comprehensive set of *Requirements*. However, *Requirements Capturing* is not in the scope of my research. My starting point is the assumption that somebody has defined *Client Requirements* using some method and in some structured form which has a relation to the project, site, building and *Spaces*.

There are several structured *Requirements Capturing and Documentation* methods; including Quality Function Deployment (QFD), Client Requirements Processing Model (CRPM), Total Quality Management (TQM), and Failure Mode and Effects Analysis (FMEA) [*Kamara et al., 2003* [21]].

QFD includes many *Requirements Management* features, and it is widely used in other industries. However, it is not commonly used for building projects. One of the reasons might be the different process compared to the manufacturing industries, for example: the AEC industry produces mainly unique buildings, which makes the design process different compared to the design of the mass-products of the manufacturing industries. The AEC industry makes no prototypes – or every building is a prototype – and usually the objectives are not clearly defined in the beginning of the process. In many cases, there are no defined metrics for most objectives even at the end of the process. Calvin Kam studied the decision making process in his Ph.D. research. In his four case studies, 77 decision topics had only 7 defined criteria [Figure 12, *Kam, 2005* [22]]. One of the reasons is that it is nearly impossible to define clear metrics for some *Requirements* (Section 8.3.1.8), but often *Requirements* are not explicitly defined even when it would be possible.

The design and decision making process in the AEC industry is not as well defined as in the manufacturing industries. As described in Section 1.1, the *Requirements* on AEC projects change throughout the process; the decision points are less clear than in the manufacturing industry and the product specification is not fully “frozen” even when construction starts.

To / From	Decision Topic	Criterion	Option	Alternative
Decision Topic **78**	**77**	**7**	**37**	**4**
Criterion **7**				
Option **96**	**7**		**64**	**1**
Alternative **11**	**1**		**33**	**2**

Figure 12: Decision topics and criteria, 4 case studies [*© Kam, 2005*]

Earlier research has identified several additional reasons why QFD has not been adopted by the AEC industry. It has been observed that the effectiveness of QFD diminishes downstream, e.g., in actual design and planning stages, phase 3 and 4 in Figure 13, which are the stages of building design and construction activities [*Evbuomwan, 1994* [23]]. Prasad argues that this makes QFD less likely to deal with complex products and conflicting *Requirements*, such as buildings [*Prasad, 1996* [24]]. Furthermore, the latest AEC-related QFD research [*Syed et al., 2003* [25]] finds the method potentially useful for defining strategic goals, but not for detailed *Requirements*. An interesting observation about QFD was the LCE’s PM’s comment of the need to integrate the tools into today’s practice, instead of trying to bring a new software platform to the process (Section 1.2.2.1). This supports the basic idea of my research: linking *Requirements* and existing design software.

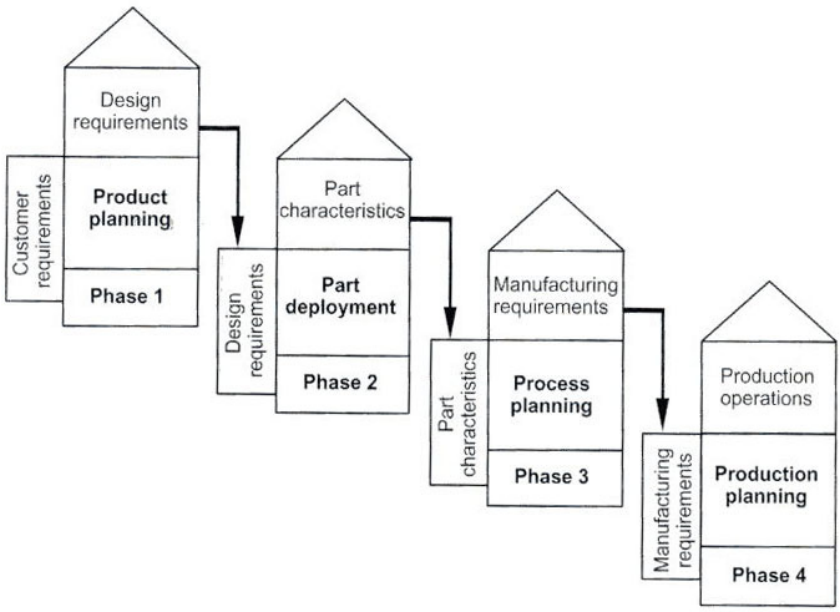

Figure 13: The four stages of the QFD process [*Kamara et al., 2003* [26]]

Kamara, Anumba and Evbuomwan developed the CRPM system to improve the *Client Requirements Capturing* process [*Kamara et al., 2003* [27]]. Its main focus is in high-level *Requirements*, but its most detailed level could be a useful source for *Space*-related *Client Requirements*. However, the method is new and not widely tested or used. In addition, from the viewpoint of the problems I have identified in this research, the division to primary, secondary and tertiary *Requirements* in CRPM system is somewhat arbitrary [*Kamara et al., 2003* [28]]. It is difficult to discern the lower-level *Requirements* from the higher-level *Requirements* and even more difficult to evaluate how the changes in lower-level *Requirements* affect the higher levels. Another problem in the system is that the proposed weighting system in CRPM [*Kamara et al., 2003* [29]] is applicable for a small number of *Requirements* only. However, in reality the choices are seldom done based on individual *Requirements* but combinations of *Requirements*. In large projects the number of such combinations becomes so high that the usability of the weighting system is questionable (Section 8.3.1.9). Thus, I have not included such a method in my *Requirements Model Specification*.

More important from my research viewpoint is that traditional *Building Programs* provide at least the same information related to the *Spaces* than the CRPM. However, CRPM is an interesting effort to structure and manage *Requirements*. A direct data link from CRPM, or some other existing *Requirements Capturing* tool, to my *Requirements Model* is a possible future research topic (Section 8.3.2.5), but it is not in the scope of my research.

As stated above, the most common method to document *Client Requirements* is the traditional *Building Program*, and I have chosen it as the starting point for my *Requirements Model Specification*. In addition, my argument is that as long as the information content is the same, my method can help *Project Team* to manage *Client Requirements* regardless of the capturing method; the purpose of required area, minimum temperature, access control, etc., is the same if they are defined with QFD, CRPM or any other method. The important issue is the relevant content, and though it cannot be fully standardized, as described above, my main contribution is to define a concept and method to link different types of *Requirements* to the *Building Product Model*.

The focus of my research – detailed *Direct and Indirect Requirements* for architectural design, and their connection to the *Design, Production, and Maintenance Models* – is specific to the construction industry. My argument is that because of the specific product structure and different processes the existing *Requirements Management* methods used in other industries, such as software engineering, do not directly apply to the identified problem on the practical level, although many of the principles apply on a generic level, such as the iterative design process, documentation, and traceability of *Requirements* [Figure 14, *Oinas-Kukkonen, 1997* [30]*; Karatmaa, 2000* [31]*; Swebok 2004* [32]]. In addition, I have not found any examples of links between *Requirements* and *Design Objects* similar to my solution documented in Chapter 6.

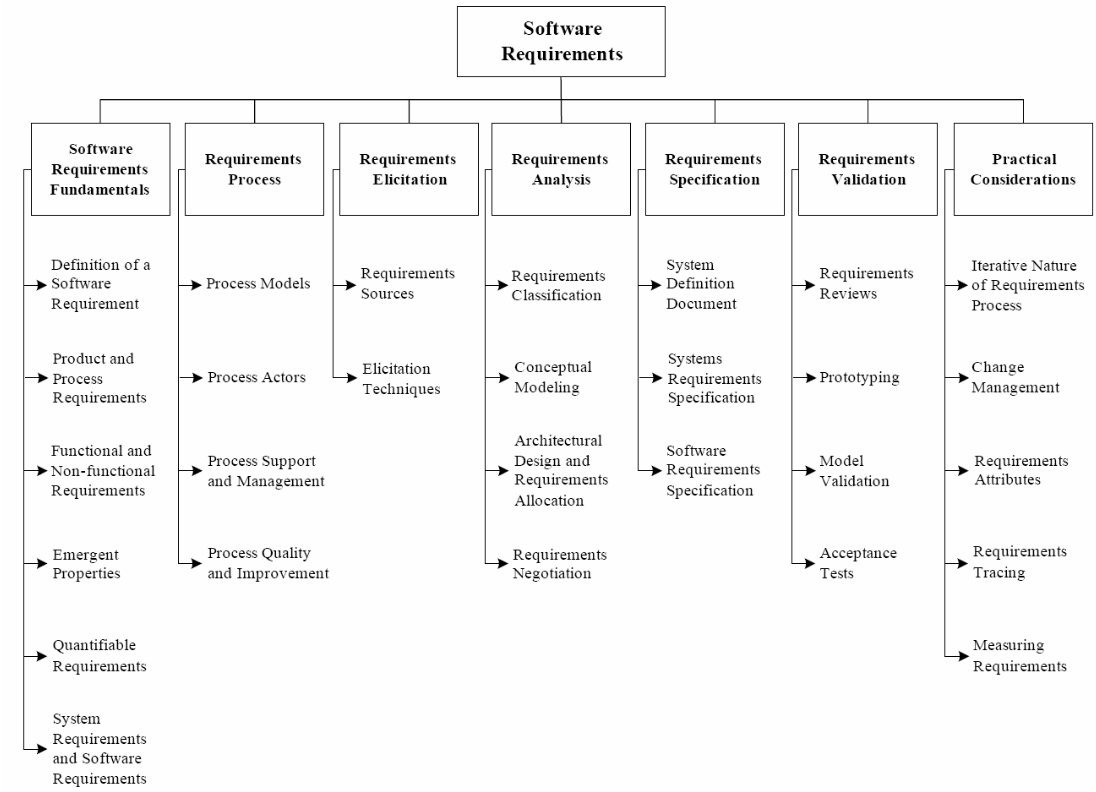

Figure 14: Breakdown of topics for software requirements [*Swebok 2004* [33]]

3.2.2 Existing Requirements Hierarchies

As the starting point for the development of my *Requirements Model Specification*, I selected two existing *Requirements Hierarchies* which are relevant to address some of the problems identified in Section 1.1.3, consist of a large number of different *Requirements* and have been used widely in the industry.

3.2.2.1 Serviceability Tools of International Centre for Facilities

The International Centre for Facilities (ICF) has published several volumes documenting their standards for Whole Building Functionality and Serviceability (WBFS) since 1992 [*ICF 1993* [34], *ICF 2000* [35]]. The purpose of these standards is to help organizations to define their functional *Requirements* for the buildings. The methods and scales in the standards are applicable for the evaluation of existing buildings and on some level also for definition of *Requirements* for a new building project. However, the focus of the WBFS system is in its use as a check-list for gathering data and evaluating existing buildings from the portfolio management or tenant viewpoint.

The *Requirements Hierarchy* of WBFS is very detailed and includes a set of scales which can be used in defining the minimum required level and the importance of each scale. The system covers several building types and different activities in the buildings. A detailed list of the items in the WBFS *Requirements Hierarchy* is in Appendix B5, Table 17 and Table 18. Some *Requirements* in the ICF system, especially in Table 18, describe operation and maintenance services or the condition of an existing building, and are therefore not relevant for my research.

The main value of the WBFS system is its systematic approach to defining evaluation scales for *Requirements*. Each of the 90 main topics usually include 5 descriptions of different *Requirement* levels which are rated on a scale from 9 to 1 (9, 7, 5, 3, and 1), and the end-user can define the importance (Exceptionally important, Important, Minor Importance, Not applicable, Not relevant) as well as the minimum threshold level for each topic. This helps both the owner and tenant compare several existing buildings. However, the system is not as useful in defining *Requirements* for design and especially linking purposes, because many descriptions combine several *Requirements* elements or describe the activity rather than its explicit *Requirements* for the design.

The WBFS system's viewpoint differs from my research scope. The WBFS system provides a high-level, strategic view for evaluation of buildings, while my research concentrates on detailed information needed for design solutions. The *Requirements Hierarchy* in the WBFS system is based on high-level functions, and in many cases the descriptions do not provide information in a usable format for linking purposes. However, there is of course a connection between these issues. The WBFS descriptions could serve as the *Requirements* intent in the PREMISS system (Section 6.3.3), and the *Project Team* could elaborate them into more detailed *Requirements*. The detailed *Requirements* must match with the strategic *Requirements*. The CRPM system (Section 3.2.1) is trying to build this connection, and an alternative, less structured, but potentially usable, approach is briefly discussed in Section 8.2.3.1. This is not in the scope of my research and in Section 8.3.1.7 I propose this issue, the relationship between strategic and detailed *Requirements*, as one of the topics for future research.

3.2.2.2 EcoProp by VTT

VTT (Technical Research Centre of Finland) has developed a software application called EcoProp [*EcoProp*[36]], which is intended to help building owners to define the sustainability *Requirements* for their building projects. The definition of sustainability in the tool is very broad and it covers not only ecological and energy *Requirements*, but a wide variety of *Requirements* related to the functional *Properties* and quality aspects of the building. It is clear that some parts of the EcoProp hierarchy are based on the principles of the WBFS system, although there are some differences.

The latest version of EcoProp, version 4.1.0 (Figure 15), can export the *Requirements* in the IFC format. All *Requirements* are exported on the building level. However, many *Requirements* defined in the system should in fact relate to *Spaces*. This link to the wrong level in the *Model* makes the current IFC implementation and the use of information for the design process less useful than it could be. It is difficult, if not impossible, for designers to know to which *Spaces* the *Requirements* relate if it is not specified.

There are also some small logical errors in the system. In some parts, the system includes very detailed content on the level which should rather be a categorization. An example of this is Category A3, ‘Services’, which is a long list of uncategorized services; in total 30 items. In my opinion the system should define different categories of services for which the user could then define the content. For example, rather than trying to include all possible restaurant types and food shops in the system, there should be a category “Food Services” for which the end-user could define the required services.

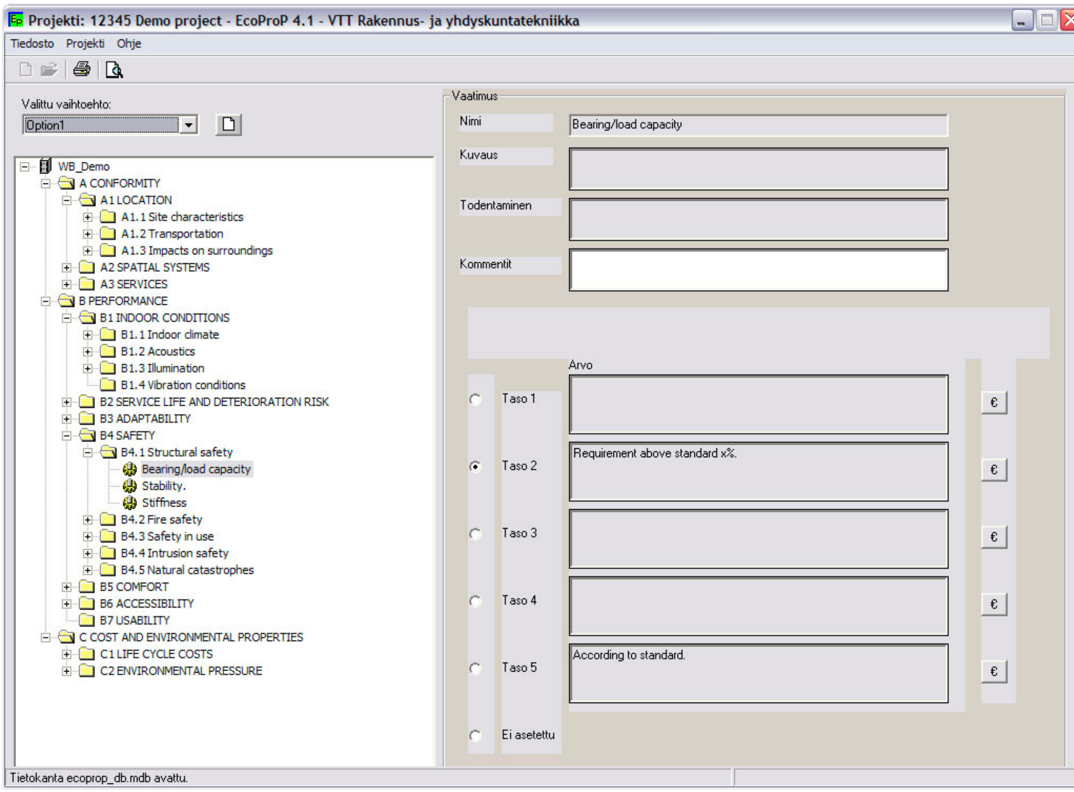

Figure 15: EcoProp user-interface

A similar issue is related to EcoProp's Category B4.5, 'Natural Catastrophes,' which tries to list all possible accident risks and catastrophes. Again, making a full list of all possible risks and catastrophes is practically impossible, and many natural catastrophes are related to just some areas in the world, which makes most of the items on a long list irrelevant for most places. For example, snow-storms are hardly an issue in San Francisco or Sydney, while earthquakes and bush fires are not relevant risks in Helsinki. I bring up this issue, because having many irrelevant issues on the lists can cause difficulties in the use of the system; the relevant items disappear in the "noise." This issue is important also to my *Requirements Model Specification* (Sections 4.3, 6.4.3, and 8.2.2.2).

Examples of different types of logical flaws in the system are, for example, that the cost and environmental *Requirements* are in the same main category, which is not a very logical grouping. I have divided them into two different main catego-ries. Similarly, in my opinion 'Radiation accident' and 'Toxic substance leak' are not ‘Natural Catastrophes’ as they are categorized in EcoProp, but ‘Accident Risks’. In my opinion, this issue is not just a “word play”; natural catastrophes relate to location and their occurrences cannot be influenced, but accidents result from human activities and the project can cause these risks as well. Thus, I separated these two issues, but at the same time reduced the list to only a few risk related categories in my *Requirements Model Specification* (Section 6.3.6.9).

The third type of development which I have done in my *Requirements Hierarchy* compared to EcoProp is the differentiation of ‘Site Selection Requirements’ on the project level from the ‘Site Design Requirements’ on the site level. ‘Site Selection Requirements’ relate to the selection of the site for the project; for example, available infrastructure and transportation systems. ‘Site Design Requirements’ relate to the design of the project; for example, the number of parking spaces, permitted building footprint, and number of floors.

In addition, EcoProp contains some redundancies; the same *Requirements* are repeated in slightly different format. In some cases, the same *Requirements* are defined based on the *Space* or building type, which is not consistent with the overall structure of the system. The detailed analysis, comparison to the PREMISS system, and rejected EcoProp *Requirements* are in Appendix B, Table 15 and Table 16.

However, this critique in some details does not mean that the EcoProp system would be unusable. On the contrary, it formed an excellent point of departure for my *Requirements* analysis (Chapter 4 and Appendix B, Table 13). With minor modifications it formed also the framework of PREMISS *Requirements Hierarchy* (Chapter 6 and Appendix B, Table 14, and Table 15). It could also easily be developed to use my *Requirements Model Specification* to connect it to the *DPM Models* (Section 8.2.3.1).

3.2.2.3 Building Codes and Other Requirements Set by the Community

Building codes and other *Requirements* defined by the community are both important *Requirements* for building projects. However, their nature from the *Requirements Management* viewpoint is very different.

Building codes are a legally binding, relatively static set of *Requirements* and they affect all building projects. The need to include specific information about the building codes into a *Requirements Management* system depends on the project type. A justified assumption is that the designers must know the local regulations without extensive project-specific documentation. However, if there are many unusual codes related to a project, it might be useful to include at least some links to the code in its *Requirements Model*. In a "standard" project the linkage would be feasible only if the building codes would be in a format which could be automatically linked. To my knowledge, such a system is not available today. However, Singapore is in the process of developing such a system in the Corenet project [*Corenet 2004* [37]]. The analysis of building codes is not in the scope of my project, but I have included a possibility of refering to the codes in most *Requirements Objects* in my *Requirements Model Specification* (Section 6.3).

The next level of community *Requirements* for a project are the site-specific *Requirements*, such as zoning codes: allowed *Location* and height of the building, noise, glare, shading limitations, etc. These are necessary information for the design, and because they vary from project to project, they should be included in the *Requirements Model*. The *Requirements Model Specification* contains several attributes in this category (Section 6.3).

The third type of community *Requirements* is the various comments, expectations and limitations set by the neighbors and different other interest groups. In many cases these *Requirements* can affect a building project strongly, and in some cases even prevent the whole project. Thus, recording and managing these *Requirements* can be crucial for the project, and I have included a possibility of including these *Requirements* in my *Requirements Model Specification*. However, these *Requirements* are difficult to predefine and thus the *Specification* contains two generic elements for this purpose, CommunityReference and CommunityRequirements. The first can refer to any types of documents and the second can contain free textual description of these *Requirements* (Section 6.3.7.2).

3.2.2.4 Conclusions from WBFS and EcoProp Hierarchies

Both WBFS and EcoProp are useful tools for *Requirements Capturing*, and each contains a well structured *Requirements Hierarchy* for their intended purpose. However, as described above there are two main limitations related to the problems addressed in Section 1.1.3:

- The WBFS system has no connection between *Requirements* and design tools, and EcoProp links all *Requirements* on the building level only.
- Neither system formally identified *Indirect Requirements* resulting from the *Direct Requirements* (Section 1.1.3.4 and 6.1.6)

However, both systems provided an excellent point of departure for the PREMISS *Requirements Hierarchy* documented in Chapter 4 and Appendix B.

3.3 LBNL's Requirements Management Research

In the technical systems area the research to capture and manage the *Requirements* has been more active than in the research related to *Requirements* for architectural design. Lawrence Berkeley National Laboratory (LBNL) has carried out several projects in which the main focus was in building performance and especially in energy efficiency [*LBNL 1995–2003* [38]]. Two main efforts have been the Building Life-Cycle Information Support System (BLISS) and the Design Intent Tool. As described in Sections 3.3.1 and 3.3.2, these projects do not provide a direct basis for my research, but the work at LBNL in this area is related to my research. Thus, collaboration with LBNL's development has been an important part of my research, and Dr. Vladimir Bazjanac from LBNL's Environmental Energy Technologies Division is one of the Advising Committee members of this doctoral dissertation.

3.3.1 Building Life-Cycle Information Support System, BLISS

The BLISS development aimed to be consistent with the *IFC Specifications*, and according to the BLISS web site the project goals partly overlapped my research [*LBNL BLISS, 1997* [39]]: "*The goal of the BLISS effort is to create a software infra-*

structure that can be used for information sharing across disciplines and can be used to link interoperable software tools throughout the building life cycle. The project has three major elements: (1) to specify the distributed software architecture, (2) to develop a life-cycle building model database schema, and (3) to develop a mechanism to capture and update "design intent" throughout the life cycle. The distributed systems architecture describes how various software components communicate, and the building model schema specifies the structure and semantics of the database."

However, LBNL has not published the results, and the project has finished without the intended link between the design intent and design software. Another quote from the BLISS web site: "*An initial version of the BLISS infrastructure will be built as an extension of the Building Design Advisor data model. Intended extensions to this model include data for documenting design intent, in the form of performance metrics, and time-series data for documenting actual building performance over time. An initial implementation of BLISS is expected to be developed during 1997.*" The website is still accessible (January 2005), but the link to software tools points to a non-existing page.

3.3.2 Design Intent Tool, DIT

The Design Intent Tool is publicly available software, including some parts of the earlier BLISS development mentioned above. Quote from LBNL's website [*LBNL DIT, 2003* [40]]: *"This database tool provides a structured approach to recording design decisions that impact a facility's performance in areas such as energy efficiency. Using the tool, owners and designers alike can plan, monitor and verify that a facility's design intent is being met during each stage of the design process. Additionally, the Tool gives commissioning agents, facility operators and future owners and renovators an understanding of how the building and its subsystems are intended to operate, and thus track and benchmark performance."*

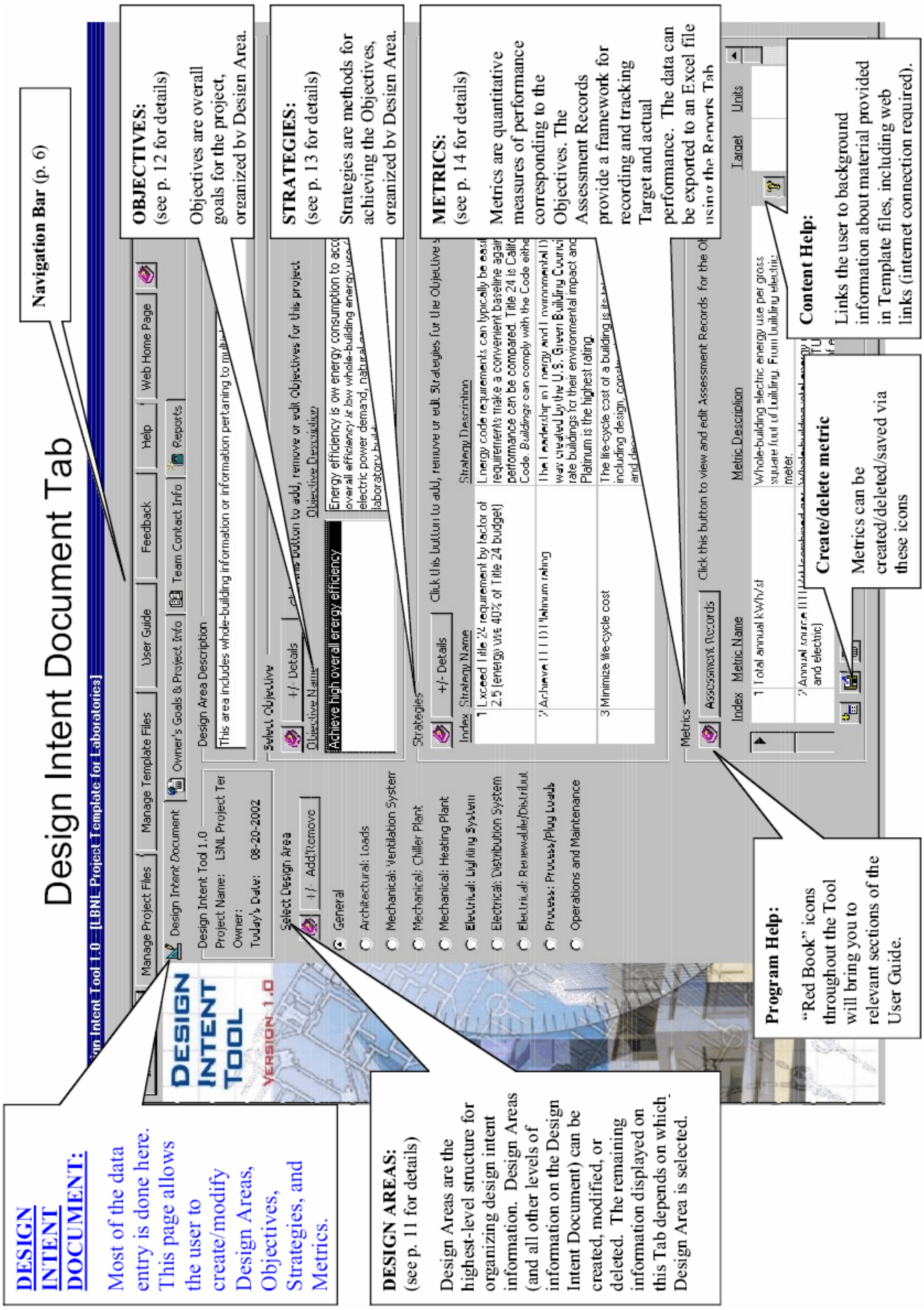

Figure 16: Design Intent Tool's user-interface, © LBNL 2002 [41]

As described, the DIT implementation focuses on energy efficiency, but the overall goal, managing the *Requirements* through the design, construction and maintenance processes, is the same as in my research, though the application area is different. The tool consists of a database solution enabling flexible documentation of objectives, strategies, metrics, and responsible agent for the MEP systems (Figure 16). All these elements are useful for my *Requirements Model Specification*. In addition, DIT provides a usable example for the user-interface design, which I propose as one of the future research topics (Section 8.3.2.3). However, the tool concentrates on *Requirements Documentation* only, and does not have a link to the design solution, which is the core element for my research.

3.4 Current Status of Building Product Models

The key element in my research is a link between *Requirements* and design solutions. As described in Section 3.1, the link cannot be based on traditional, human-readable documents. Its prerequisite is a semantic *Building Product Model* which consists of objects such as Building, *Space*, wall, door, window, and system.

Many current architectural design software products are based on such a *Model*: for example, ArchiCAD by Graphisoft [*ArchiCAD* [42]], Architectural Desktop [*ADT* [43]] and Revit by Autodesk [*Revit* [44]], MicroStation by Bentley [*MicroStation* [45]], even some low-cost software, such as Visio by Microsoft [*MS Visio* [46]]. All these products have their own internal *Model*, and they could serve as a basis for the described link. However, development of a link between a *Requirements Model* and these proprietary *Design Models* is complicated by three main problems: (1) the structure of a proprietary *Model* can change without any notice, (2) each product needs a different link, and (3) the documentation of the internal structure of a proprietary *Model* might not be publicly available. Thus, a publicly available documented *Design Model* is a better basis for research purposes. However, the same principles apply to links between *Requirements* and proprietary *Design Models*.

The International Alliance for Interoperability (IAI) has developed *Building Product Model Specifications* for the AEC/FM industry. IAI has produced several versions of these *Specifications* called Industry Foundation Classes (IFC). The IFC2x Platform *Specification* was officially accepted as a Publicly Accessible Specification ISP/PAS 16793 by ISO (International Organization for Standardization) in October 2004 [*ISO 2004* [47]]. This gave an official standard status to the *IFC Specifications*. In addition, in January 2004 the US General Service Administration (GSA) published an Internal Directive stating that the GSA will start demanding IFC compliant *Design Models* to support concept reviews for projects receiving design funding in FY2006 [*IAI NA 2003a* [48]]. This strengthened significantly the status of *IFC Specifications* also as a de-facto standard for *Building Product Models*. Thus, *IFC Specifications* are the logical basis for the *Building-Product-Model*-related part of my research. The *IFC Specifications* and their implementation provide sufficient information content for the objects related to the *Requirements* relevant for my research.

William Behrman strongly criticizes top-down data exchange standardization efforts, such as IFC [*Behrman, 2002* [49]]. Many of his arguments are valid, such as the difficulty and slow speed of the development and complexity of the implementation of the standard. I agree with Behrman that the lack of high-level commitment of a critical mass of key players is a fundamental problem in data standardization efforts in the AEC industry.

However, the bottom-up development – independent minimalist standardization based on each use-case, which Behrman recommends – has not been more successful in the AEC industry or replaced IFC development since the publication of Behrman's report. On the contrary, aecXML, which tried to use the bottom-up approach, has not progressed since 2002, while IAI has published two new versions of the *IFC Specifications*. landXML and gbXML are still the only aecXML schemas; both discussed in Behrman's report and still in draft stage almost three years later [*aecXML 2005* [50]]. Although the development and implementation of the *IFC Specification* has been slow, it has progressed during that time, and as mentioned strengthened its position as a de-facto standard since 2002.

In addition, Behrman's report does not include the latest technologies in IFC implementation: IFC *Model Servers* and standardization of their APIs (Appendix C, C2). The development of *Model Servers* started in 2001 and as of January 2005 three products exist [*IMSvr 2002*[51], *WebSTEP 2002*[52], *and EPM 2003*[53]]. This development would not have been possible without a comprehensive *Model Specification*, such as the IFCs. The *Model Servers* and their standardized APIs hide the complexity of the underlying *Model* and enable the use of standard protocols in data exchange, such as XML and SOAP in the software implementation (Appendix C, C2), which is one of Behrman's main critiques of the *IFC Specifications*.

For me, the most important reason to use the *IFC Specifications* as the basis for my *Requirements Model Specification* is that the *IFC Specifications* are the only existing open and documented standard for *Building Product Models*. Thus *IFC Specifications* are the only non-proprietary basis for a link between *Requirements and Design Models*.

However, the same principles of how to link *Requirements* with *Design Objects* apply to any semantically meaningful representation of building *Models*, although the detailed modeling language would be different. Thus, the usefulness of my *Requirements Model Specification* is not dependent on the success of the *IFC Specifications;* an existing open standard simply provides an easier platform for the implementation.

One of the limitations in the current *IFC Specifications* is related to *Requirements Management*. The main focus in IFC development has been on the design view; i.e., the *Specification* includes extensive building geometry representation and many other design *Properties* for building objects, but it does not support other phases of the process as well. When I started my research, the *IFC Specification* version 2x contained only limited support for *Space*-related *Requirements* (Figure 17).

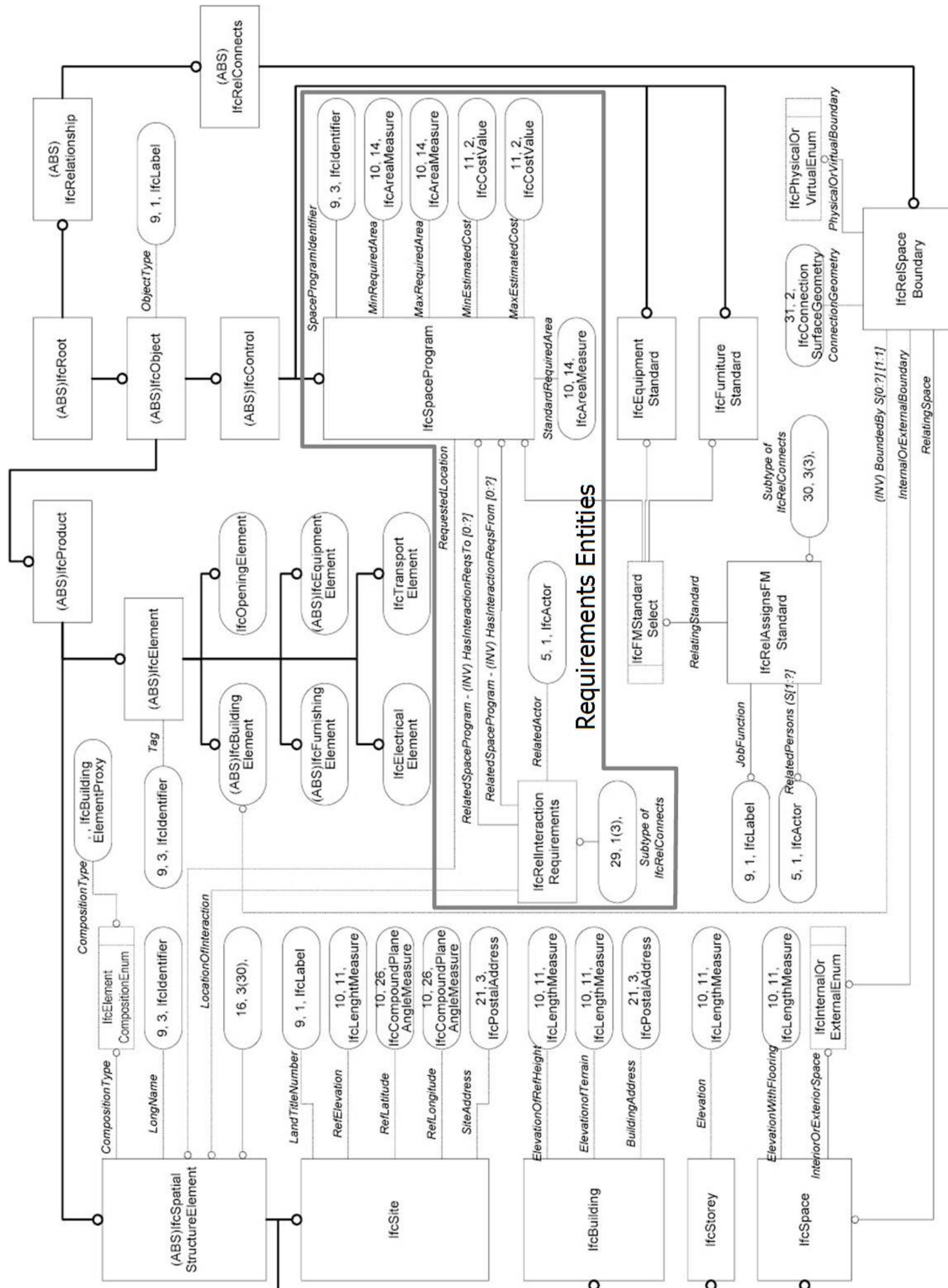

Figure 17: IFC 2x *Space*-related *Requirements* elements and their relations

As far as I know, this *Space Program* part of the *IFC Specification* has been implemented only in one commercial software, Alberti, which was developed by acadGraph [*acadGraph* [54]]. This software does not support *Requirements* other than minimum and maximum areas and physical connections between *Spaces*, and based on the experiences in a Finnish project which tested the software in 2000–2002, it is not a suitable tool for large projects [*SPADEX 2002* [55]]. The main reason for this is the complexity of defining the connections, and the software's attempt to automate the creation of *Space* layout, which is extremely complicated if the number of the *Spaces* is large.

In addition, the *IFC Specifications* include a generic *Requirements* object, IfcConstraint, and several *Property Sets* for *Requirements*. These are analyzed in detail in Sections 6.2.1 and 6.2.2.

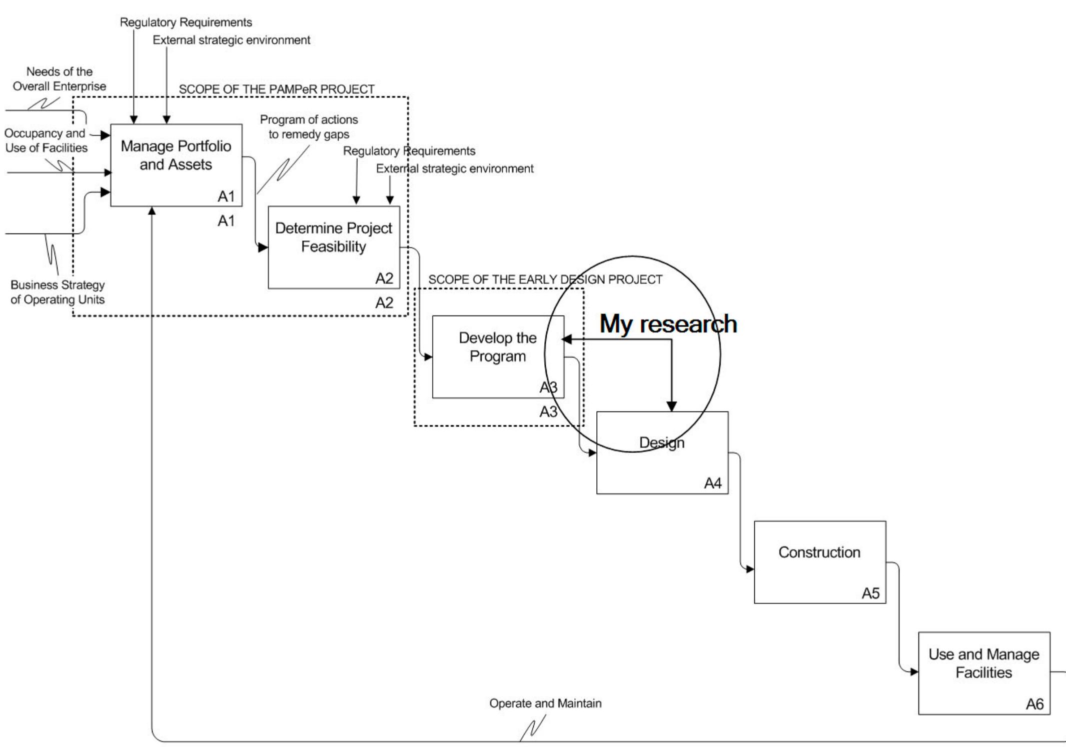

Figure 18: PAMPeR/ED project scope and relation to my research [*IAI NA 2003b* [56]]

There are two on-going projects expanding the *IFC Specifications* to the *Requirements* level [*IAI NA 2003b* [57]]. These projects are:

- Portfolio and Asset Management: Performance Requirements (PAMPeR, IAI FM-9)
- Early Design (ED, IAI AR-5).

The focus in these two projects is in capturing and documenting the *Requirements*, not in linking the *Requirements* to *Design Models*, which is the focus of my research (Figure 18). In addition, the *Requirements* are stored in *Property Sets*, which have certain limitations (Section 6.1.4)

However, during my research *IFC Specifications* have also developed. Two new versions, IFC 2x2 and IFC 2x2 Addendum 1, have been published since I started my work. They include some development, which is relevant for my work, but the basic problem has remained. The *IFC Specifications* have no coherent definitions for *Requirements*, they are scattered in several *Property Sets*, and a large portion of the *Requirements* is attached to the *Space* entities (Section 6.2.2). The objects related to my *Requirements Model Specification* in the latest version of *IFC Specifications*, IFC 2x2 Addendum 1, are documented and analyzed in detail in Section 6.2.

My argument is that including the *Requirements* in the *Design Objects*, such as *Spaces*, on the *Instance* level in the *Building Product Model* is not a feasible solution to *Requirements Management* (Section 5.1.1 and 6.1.4). In addition, IFC implementation is already very demanding and this has created the need to develop easier methods to use the *IFC Specifications* [*SABLE 2003* [58] and Appendix C, C2]. I argue also that on the *Instance* level the *Requirements Model* and *Design Model* are two different levels of abstraction (Section 5.1.1). Thus, combining them into the same objects on the *Instance* level would make both the implementation of the *IFC Specifications* and the project's information management more complicated.

My solution is a link between *Requirements Objects* in the *Requirements Model* and objects in *Design, Production, and Maintenance Models*, separating the

Requirements and solutions at the concept level to individual objects (Figure 24). This approach also matches research on representing form, function and behavior (FFB) (Section 1.1.5). However, this creates a new challenge for the *IFC Specifications*, because the current *Specifications* do not include a mechanism to link objects in two different *Models*. I am proposing such an addition to the *IFC Specifications* (Section 6.3.2). It is one of the scientific and practical contributions of my research (Sections 8.1.3 and 8.2.4.2). In addition, this link between two *Models* includes aspects for proposed future research (Section 8.3.1.6).

As of February 2005, to my knowledge the only *Requirements Capturing* tool supporting IFCs is EcoProp (Section 3.2.2.2). However, its content is limited mainly to sustainability issues, although it covers some common project objectives as well. All *Requirements* in the EcoProp system are connected to the building level. In addition, Anders Ekholm and his research group executed an object-based briefing study recently [*Ekholm and Lehtonen, 2002* [59]], and there is also a prototype software linking area *Requirements* to the *Design Model*, Space Layout Editor (SLE) [*BLIS 2004* [60]].

The semantic structure of the *IFC specifications* and its current implementations provide the basis to link the *Requirements Model* and *Design, Production, and Maintenance Models* as described in Section 5.1.1. The needed elements from the *Design, Production, and Maintenance Models* are identifiable objects which can be linked to the *Requirements Objects*, and the identification of related objects which can be affected by the *Indirect Requirements*, such as *Bounding Elements* of the *Space* and technical systems serving the *Space*.

For practical software applications, the preferred solution to implement the interface between the *Requirements Model* and the existing *Building Product Model* applications is to use *Model Server* technology and some standardized API, such as the SABLE interface described in Appendix C, C2. Using a standardized API would make the implementation easier and provide a connection to several design software. However, this is not in the scope of my research; the connection between a *Requirements Model* and *DPM Models* can be implemented on three levels (Figure 25):

- Using the SABLE application interface [*SABLE 2002* [61]]
- Using one of the IFC compliant *Model Servers* [*IMSvr 2002* [62], *WebSTEP 2002* [63], *EPM 2003* [64]]
- Directly with some *Building-Product-Model*-based software [for example, *ArchiCAD* [65]*, ADT* [66]*, MS Visio* [67]], by creating a link between the design software and the *Requirements Management* application.

The IFC file exchange is naturally the fourth method to import *Requirements* to a Design Software; for example, EcoProp and the SLE prototype use this method. However, the file exchange does not create a real connection between the *Models;* it can only export and import information. This means that when the project evolves the information in either *Model* can get outdated easily.

3.5 BLIS Views

Because of the complexity of the *IFC Specifications* and the ambiguity of the EXPRESS language, it is possible to implement a *Specification* in several ways, and any individual software product supports only some parts of the *Specification.* However, the information exchange must be based on the same content and interpretation of the *Specification*. Thus, the software developers need additional guidelines and agreements on how to implement the *Specification*. In IAI, the software vendors have made these agreements in Implementation Support Group (ISG) meetings, but the process has not been systematic; the implemented part is defined mainly based on the information structure of the existing software products. This means that those features in the *IFC Specifications* which would add new functionalities into software are easily ignored. In addition, the agreements are not documented and published adequately. The information of the existing agreements is not easily available and this makes the implementation difficult for new developers.

To correct this situation, the Building Life Cycle Interoperable Software (BLIS) group developed the concept of *Implementation Views* to support IFC-based information exchange [*BLIS 2004* [68]]. The views are based on a thorough analysis: what is the necessary information content for a certain task, and how should the software products present that information, such as geometry and properties. These views are then documented in detail and published on the BLIS web site. The current BLIS *Implementation Views* are:

- Architectural Design -> Quantities Take Off / Cost Estimating
- HVAC System Design -> Quantities Take Off / Cost Estimating
- Architectural Design -> Thermal Load Calculations / HVAC System Design
- Client Brief / Space Layout -> Architectural Design
- CAD View

The relevant *Implementation View* for my research is "Client Brief / Space Layout -> Architectural Design" (CB/SL-AD). The following descriptions are quotes from the BLIS website [*Hietanen, 2003* [69]]:

"The view for 'Client Brief / Space Layout -> Architectural Design' defines the subset of the IFC model that is used for transferring spatial data from the client brief to architectural design applications.

The Client Brief application can be anything from a simple spreadsheet to a real application, the important thing is that it can output the requirements captured in the client brief into IFC format. Architectural design applications can import the resulting IFC file and start the actual design process designing the Spaces, walls, doors, windows, etc. There can also be a special space layout program between the Client brief and the architectural design application.

The first level of functionality is to be able to generate a 'space skeleton' that matches the requirements set in the client brief, e.g., the right number of Spaces of the right types and areas. The second level is to actually store the requirements in the design application and to be able to give feedback about how the design meets the."

As described in Section 1.1.4, I base my solution for the second level of functionality on separation of the *Requirements Model* from the *Design Model* instead of storing the *Requirements* in the design application. I discussed this approach with the BLIS technical team and accepted as the preferred basis to expand the current CB/SL-AD view. This approach also enables the use of a *Model Server* database as the repository for both the *Requirements and Design Model* information (Figure 93, Option #2).

3.5.1 List of supported concepts in the current CB/SL-AD view

The BLIS *Implementation Views* consist of 'concepts'; functional units isolated from the *IFC Specifications*. The *Implementation Views* are built by combining the relevant 'concepts' using them as "building blocks." The following lists of BLIS 'concepts' in the CB/SL-AD view for IFC 2.0 are grouped based on their relevance for the *Requirements Model Specification* and the link between it and the *IFC Specifications*. A short explanation is in the brackets after the 'concept' name:

BLIS 'Concepts' which are part of the *Requirements Model Specification*:

- *Actor role* (Part of IfcActorSelect, and thus can be part of the new Requirement Element object, Section 6.3.4.)
- *Building* (One of the direct link levels between the *Requirements and Design Models*.)
- *Building story* (Can be a relevant link level for *Requirements,* although not often used.)
- *Containment* (For example, *Space* can be a container for its furniture and equipment.)
- *Dynamic property assignment* (The mechanism to assign property objects or *Property Sets* to objects dynamically, i.e., without changing the IFC schema. This can be used to add new *Requirements* to the *Model*. However, there must be some agreement on the additional attributes, because otherwise other applications cannot handle them.)
- *Organization* (Part of IfcActorSelect, and thus can be part of the new Requirement Element object, Section 6.3.4.)
- *Owner history* (Defines the ownership of the objects in the *Models*.)

- *Person* (Part of IfcActorSelect, and thus can be part of the new Requirement Element object, Section 6.3.4.)
- *Project* (One of the direct link levels between the *Requirements and Design Models*.)
- *Property Set system* (Software developers can use this method to attach new properties to IFC objects; e.g., implement attributes, which are not defined in the *IFC Specifications*. However, this method causes problems, which are discussed in Section 6.1.4.)
- *SimpleProperty* (This defines a simple property for a *Property Set*. The 'SimpleProperty' has a name and a value.)
- *Site* (One of the direct link levels between the *Requirements and Design Models*.)
- *Space* (Central element for the *Requirements Management Specification*. *Space Program Instance* objects in the *Requirements Model* link to the *Space Objects* in the *DPM Models*.)
- *Space program properties* (Central element in the *Requirements Model* and the link to the *Design Model*. The *Requirements Model Specification* defines two new elements to replace the current IfcSpaceProgram object; NewSpaceProgramInstance and NewSpaceProgramType. Sections 6.2.2, 6.3.1.5 and 6.3.10 document this issue is in detail.)
- *Space Type* (Central element for the *Requirements Model*. The current use of *Space Type* in the BLIS view is based on the use of the Description attribute to store a value of the *Space Type*, and there is a proposed list of types. The NewSpaceProgramType replaces this practice in my *Requirements Model Specification;* see Section 6.3.10.)
- *Unit assignment* (IfcUnitAssignment defines whether the units are metric or imperial.)
- *Units [metric]* (Defines the metric units used in the project.)

BLIS ‘Concepts’ which are not used in the current *Requirements Model Specification* but might be useful in the future:

- *Address* (A *Project Attribute*, not a *Requirement*. Depending on the implementation, this information can be stored in the *Requirements Model*, *Design Model*, or both.)
- Site address (A *Project Attribute*, c.f. Address.)

BLIS ‘Concepts’ which are not used:

- 2D placement (*Geometrical Locations* are not *Requirements*.)
- 3D placement (*Geometrical Location*.)
- Absolute placement (*Geometrical Location*.)
- *Bounding box geometry* (Geometrical representations are not *Requirements*.)
- *Extruded solid: arbitrary* (Geometrical representation.)
- *Geometric representation* (Geometrical representation.)
- *Polyline* (Geometrical representation.)
- *Profile: arbitrary* (Geometrical representation.)
- Relative placement (*Geometrical Location*.)

My *Requirements Model Specification* expands this *Implementation View* with several new *Requirements Objects* which also include the *Direct Requirement* links to the *DPM Models* (Section 6.3). It is possible to expand these *Requirements Objects* further using *Property Sets* for additional, project-specific *Requirements*. I discuss this issue and related problems in detail in Section 6.1.4. The expanded view is one of the scientific and practical contributions of my research and can serve as a basis for an official extension of the *IFC Specifications*.

In the current implementations of *IFC Specifications*, the identification of objects is based mainly on the Globally Unique ID (GUID), which can be problematic for several reasons discussed in detail in Section 6.2.3.3. Because of these problems the rapid prototyping (Chapter 5) was based on the idea of using the Description attribute in the SpaceCommon *Property Set* to store the RoomID as the link between the *Space Program Instance* (*SPI*) and *Space* objects in the

Design Model. However, the same concept can be implemented in several ways. Section 6.3.2 documents the solution used in my final *Requirements Model Specification*.

4 Requirements Analysis

4.1 Requirements Defined in Different Projects

The analysis of *Requirements* in *Building Programs* is based on the *Requirements Documentation* of five building projects [*Programs 2003* [70]]:

- ICL Headquarters, Helsinki 1994–1996, total gross area 27,350 m^2
- Aurora II, Joensuu University 2003, total gross area 7,120 m^2
- CSLI-Media X / EPGY Annex Building, Stanford University 2003, total gross area 1022 m^2
- Kavli Institute, Stanford University 2003, total gross area 2,330 m^2
- Lucas Center Expansion, Stanford University 2003, total gross area 1,960 m^2

The items in this analysis consist of the *Project Attributes* — such as purpose, ID and name of a *Space* — and *Requirements* — such as minimum area, number of *Spaces*, illumination, and maximum air velocity. For clarity reasons I call them in this analysis part *"Requirement Components*." The *Requirements Hierarchy* used as the basis in the analysis phase was EcoProp's attribute list (Section 3.2.2.2, Table 15 and Table 16), and all *Requirement Components* which are defined in at least one of the projects, but not in EcoProp, were added to the list. The full list of attributes is in Appendix B1, Table 13. The following Table 2 includes only the *Requirement Components* which are defined in at least one project. The "Defined" column indicates how many projects have defined each specific *Requirement Component*, e.g., number "5" indicates that all five analyzed projects use that information. The identifier column (ID) refers to the main categories of the *Requirements Hierarchy* documented in Appendix B4, Table 16, and Figure 19 documents the main types of the *Requirement Components*.

Table 2: Defined *Requirement Components:* number of projects

ID	Requirement Component	Defined
A3.1	Department	5
A3.1	Name	5
A3.2	Minimum area	5
A3.2	Number of the rooms	5
A3.2	Adjacency requirements (connections to other rooms)	4
A3.2	Maximum number of occupants	4
A3.1	Main purpose of the room (description)	3
A3.3	Activities	3
A3.3	Equipment	3
A3.3	Furniture	3
B4.2	Fire alarm and sprinkler systems	3
B4.2	Fire-resistance rating	3
B4.2	Fire-resistance time	3
B4.2	Surface layer fire-propagation rating	3
B4.2	Surface layer inflammability rating	3
B5.2	Aesthetic appearance of the building	3
B6.1	Emergency vehicle access	3
B6.2	Building is accessible for disable/handicapped	3
C2.3	External doors, U-value	3
C2.3	External walls, U-value	3
C2.3	Roof, U-value	3
C2.3	Windows, U-value	3
A3.1	Identifier	2
A3.1	Room type	2
A3.3	Ceiling finishes	2
A3.3	Doors	2
A3.3	Floor finishes	2
A3.3	Wall finishes	2
B1.1	Indoor climate, descriptive text	2
B1.1	Maximum room temperature	2
B1.2	Acoustics, descriptive text	2
B1.2	Sound insulation between rooms	2
B1.3	Daylight	2
B1.3	Illumination, descriptive text	2
B4.1	Bearing/load capacity	2
B4.1	Stability	2
B4.1	Stiffness	2
B4.2	Fire-resistance rating of functional elements and accessories	2
B5.1	Functionality and comfort of the spaces	2
B5.1	Interior design and furniture	2
B5.1	Way finding	2
B5.1	Outdoor area comfort and usability	2
B5.1	Site amenities	2
B5.2	General design objectives for the building	2
C2.1	Existing vegetation which must be preserved	2
C2.1	Existing vegetation; quantity, condition, and extent	2
C2.2	Energy consumption, lighting	2

ID	Requirement Component	Defined
C2.2	Total electrical energy consumption	2
C2.3	Base floor, U-value	2
A1.1	Gas supply infrastructure	1
A1.1	Sewage infrastructure	1
A1.1	Size and suitability requirements for the site	1
A1.1	Soil type requirements; excavation and foundation	1
A1.1	Waste service infrastructure	1
A1.1	Water supply infrastructure	1
A1.2	Accessibility for bicyclists	1
A1.2	Accessibility for pedestrians	1
A1.2	Bike parking	1
A1.2	Parking spaces	1
A1.2	Vehicular access to site	1
A1.3	Existing buildings which have related activities	1
A1.3	Existing buildings which must be preserved	1
A1.3	Noise level on the site (traffic, airplanes, neighbor buildings, etc.)	1
A1.4	Allowed building footprint size	1
A1.4	Allowed building location	1
A1.4	Allowed number of floors	1
A1.4	Wind effects	1
A3.3	Access floor	1
A3.3	Ceiling height	1
A3.3	Windows	1
B1.1	Ammonia and amines (NH_3)	1
B1.1	Carbon dioxide (CO_2)	1
B1.1	Carbon monoxide (CO)	1
B1.1	Formaldehyde (H_2CO)	1
B1.1	Individual control of room temperature (maximum ± difference)	1
B1.1	Maximum air velocity	1
B1.1	Maximum floor temperature	1
B1.1	Maximum vertical temperature difference	1
B1.1	Minimum floor temperature	1
B1.1	Minimum relative humidity	1
B1.1	Minimum room temperature	1
B1.1	Odor intensity (intensity scale)	1
B1.1	Radon	1
B1.1	Temporary deviation from set values	1
B1.1	Volatile organic compounds (TVOC)	1
B1.2	Maximum traffic noise level on the site	1
B1.3	Adjustability	1
B1.3	Brightness/shine/luster reflection	1
B1.3	Color rendering, Ra	1
B1.3	Contrast repetition/reproduction CRF	1
B1.3	Glare (IES-IND)	1
B1.3	Luminance distribution	1
B1.3	Maximum color temperature	1
B1.3	Maximum luminance at the task area	1
B1.3	Minimum color temperature	1

ID	Requirement Component	Defined
B1.3	Minimum luminance at the task area	1
B1.3	Shadow formation	1
B1.4	Vibration, descriptive text	1
B2.1	Expected building service life	1
B2.1	Expected service life of components which are difficult to replace	1
B2.1	Expected service life of load bearing structures	1
B2.1	Expected service life of major internal elements (e.g.. partition walls)	1
B2.1	Expected service life of major, replaceable external elements	1
B2.1	Expected service life of other internal elements (surface materials, doors)	1
B2.2	Easily replaceable piping (visible)	1
B2.2	Heat yield machinery (heat transfer casing/boilers, accumulators, tanks)	1
B2.2	HVAC equipment/machine heat transfer-element/installment	1
B2.2	HVAC pumps, fans	1
B2.2	HVAC-EL automation cabling	1
B2.2	HVAC-EL-automation systems (control room devices, regulation/control)	1
B2.2	Inconveniently replaceable piping (inside or behind structures)	1
B2.2	MEP-metering, safety and control devices	1
B2.2	Sewer system plumbing and components.	1
B2.2	Terminal, control and other devices in ventilation/air conditioning ducts	1
B2.2	Ventilation/air conditioning ducts	1
B2.2	Water and sewer fittings (wash basins, WC-seat, bath)	1
B2.2	Water circulation heat distribution machinery (steel pipes and battery)	1
B2.2	Water plumbing system components (sealing and control valve, mixers)	1
B3.1	Alternative furnishing of spaces	1
B3.1	Alternative use of spaces	1
B3.1	Division and combination of spaces	1
B3.1	Expandability of the building	1
B3.1	Flexibility of the building envelope	1
B3.1	Flexibility of the floor structures	1
B3.1	Flexibility of the frame structure	1
B3.1	Flexibility of the horizontal installations	1
B3.1	Flexibility of the partition walls	1
B3.1	Flexibility of the vertical shafts	1
B3.1	Initial users' possibility of making individual choices	1
B3.1	Possibilities to make changes in the use of the building	1
B3.1	Users' possibilities to make changes later	1
B3.2	Flexibility of the building automation systems	1
B3.2	Flexibility of the electrical systems on space level	1
B3.2	Flexibility of the fire alarm system	1
B3.2	Flexibility of the heating system	1
B3.2	Flexibility of the illumination system	1
B3.2	Flexibility of the main electrical distribution system	1
B3.2	Flexibility of the security and access control system	1
B3.2	Flexibility of the sprinkler system	1
B3.2	Flexibility of the telecommunications and IT networks	1
B3.2	Flexibility of the ventilation and cooling system	1
B3.2	Flexibility of the waste disposal system	1
B4.3	Electricity backup systems	1

ID	Requirement Component	Defined
B4.3	Security of information systems	1
B4.4	Space	1
B4.5	Earthquake	1
B5.1	Visual contact/privacy externally	1
B5.1	Visual contact/privacy internally	1
B6.1	Vehicular access	1
B6.2	Building is accessible for hearing impaired people	1
B6.2	Building is accessible for sight disabled people	1
C2.1	Possible effects to the fauna	1
C2.2	Energy consumption, AC	1
C2.2	Energy consumption, fans	1
C2.2	Energy consumption, HVAC system in total	1
C2.2	Energy consumption, office equipment	1
C2.2	Energy consumption, other HVAC equipment	1
C2.2	Heating/cooling energy consumption	1
C2.2	Site heating system	1
C2.2	Use of solar protection/screens	1
C2.2	Water consumption	1
C2.3	Windows, shading coefficient	1
C2.4	CO_2eq	1
Code	Building	1
Code	Egress	1
Code	Envelope	1
Code	Fire systems	1
Code	Materials	1
Code	Others	1
Code	Site	1
Code	Structural systems	1

The total number of *Requirement Components* in the list (Appendix B1, Table 13) is 277, and 171 (62%) of these are defined in at least one of the projects (Table 2). However, only 49 of the *Requirement Components* (18%) are defined in more than one project and 22 (8%) in at least 3 projects. Only 4 *Requirement Components* (1%) are defined in all five analyzed projects. This confirms also the preliminary analysis results from the two *Building Programs* before the rapid prototyping [*Kiviniemi et al., 2004* [71]]:

- There are only very few *Requirements* (1%) which are defined in all projects; most *Requirements* are project-specific.
- Most of the pre-defined *Requirements* in a typical *Requirements Capturing* system are not used for most projects.

4.2 Most Frequently Defined Requirements

4.2.1 Requirements Categories

Because the number of projects analyzed in my research is relatively small, only five, the details are not statistically significant, such as the occurrences of a specific *Requirement Component*. In spite of this, the results indicate some clear trends when observing different categories. These categories are based mainly on the EcoProp system (Appendix B, Table 16), and the category IDs in Figure 19 – Figure 21 refer to the first two characters in the individual *Requirement* IDs in Table 2. Because the goal of my research is not to specify all possible *Requirements* for building projects nor to make statistical analysis of the use of different *Requirements*, but to define relevant categories and a reasonable set of *Requirements* in those categories, the accuracy of the results is sufficient.

Requirement Components which appear in 5 or 4 projects are all in the "Spatial Systems" category, all other types of *Requirement Components* occur only in 1–3 projects (Figure 19 and Table 3). For example, 5 *Requirement Component* types used in 3 projects are in the "B4, Safety" category which is equal to 31% of the total 16 *Requirement Component* types in that group.

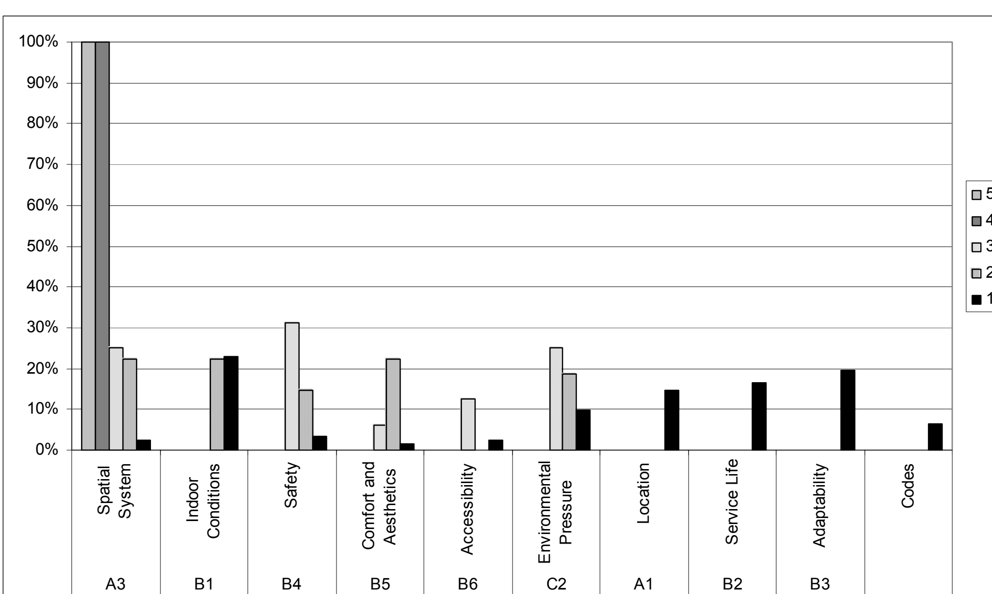

Figure 19: *Requirement Component* types used in the projects

Table 3: *Requirement Component* types used in the projects

			Number of projects using different RC types					Used	Defined	All	5-2
			5	**4**	**3**	**2**	**1**	in total	in total	projects	projects
A3	Spatial System	Number	4	2	4	6	3	19	19	55	52
		%	100%	100%	25%	22%	2%	11%	8%	22%	40%
B1	Indoor Conditions	Number				6	28	34	57	40	12
		%				22%	23%	20%	24%	16%	9%
B4	Safety	Number			5	4	4	13	25	27	23
		%			31%	15%	3%	8%	11%	11%	18%
B5	Comfort and Aesthetics	Number			1	6	2	9	9	17	15
		%			6%	22%	2%	5%	4%	7%	12%
B6	Accessibility	Number			2		3	5	5	9	6
		%			13%		2%	3%	2%	4%	5%
C2	Environmental Pressure	Number			4	5	12	21	31	34	22
		%			25%	19%	10%	12%	13%	13%	17%
A1	Location	Number					18	18	36	18	
		%					15%	11%	15%	7%	
B2	Service Life	Number					20	20	21	20	
		%					16%	12%	9%	8%	
B3	Adaptability	Number					24	24	25	24	
		%					20%	14%	11%	10%	
	Codes	Number					8	8	10	8	
		%					7%	5%	4%	3%	
	In total	**Number**	**4**	**2**	**16**	**27**	**122**	**171**	**238**	**252**	**130**
		%	**100%**	**100%**	**100%**	**100%**	**100%**	**100%**	**100%**	**100%**	**100%**

Another way to analyze the importance of different *Requirements* is to look at the total numbers of used *Requirement Component* types and their distribution to different categories. This total number is the number of used *Requirement Component* types in each category multiplied by the number of projects which have used that specific type. For example, all five projects used ‘Space Name’, ‘Department’, ‘Minimum area’ and ‘Number of the rooms’. This totals 20 defined *Requirement Component* types to the spatial system category; i.e., the total number is not the number of individual *Instances* of the type in the *Building Programs*. Counted with this method the total number of *Requirement Component* types used in the analyzed *Building Programs* is 252 (Table 3, “All projects” column).

Figure 20 presents the distribution of total numbers of *Requirement Component* types into the different categories. Again the *Requirement Component* types in the “Spatial Systems” category clearly dominate (22%), but also “Comfort and Aesthetics” and “Safety” and “Indoor Conditions” categories have over a 10% share of the total number of *Requirement Component* types.

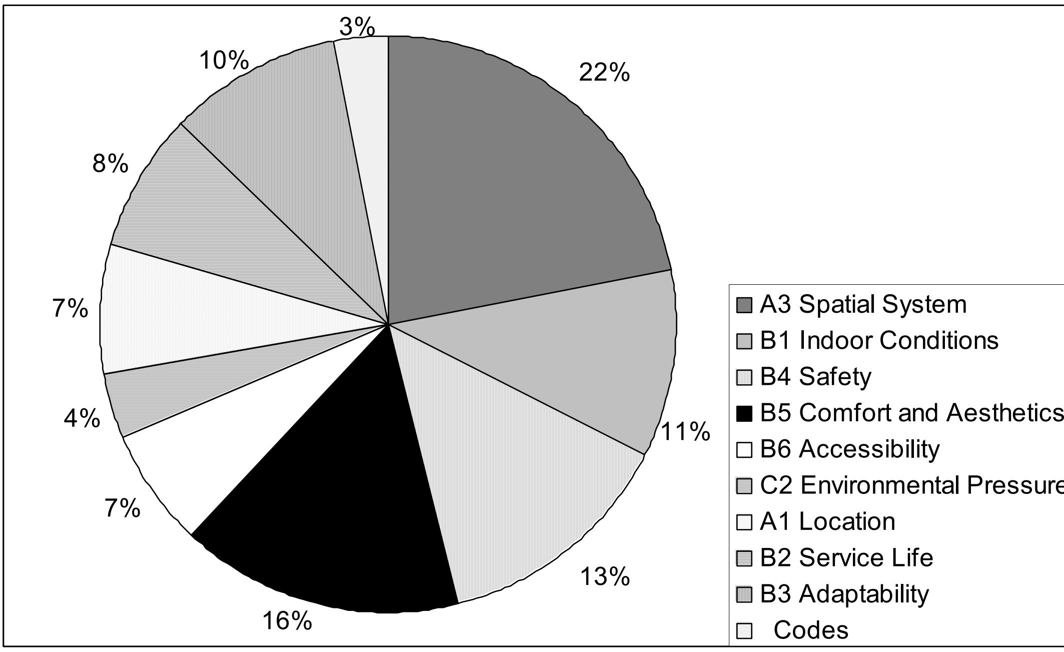

Figure 20: Distribution of all *Requirement Component* types

Counting the *Instances* of *Requirement Components*, the amount of spatial information would be overwhelming compared to all other *Requirement Components.* For example, only in the 186 *Space Program Instances (SPI)* in the ICL project requirements database multiplied by the four above-described types would produce nearly 750 *Requirement Component Instances* into the Spatial System category (nearly 800 *Space Instances* in the *Design Model*, Section 5.3). Although this comparison method is not quite relevant, it emphasizes the importance and amount of *Space Requirements* information in a project's *Space Program*.

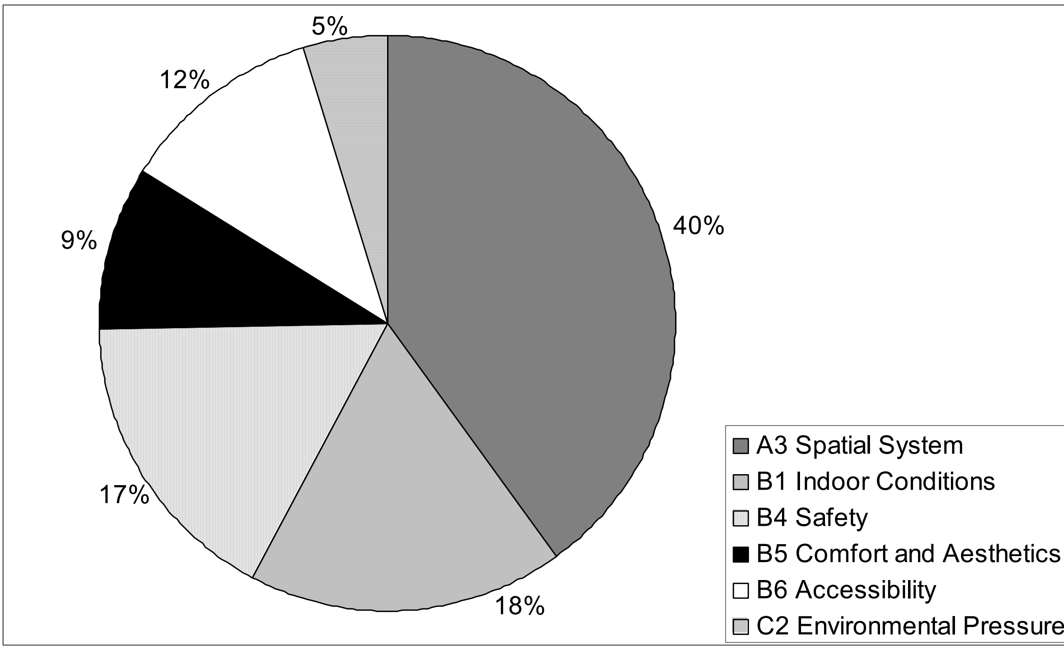

Figure 21: *Requirement Component* types defined in at least two projects

Figure 20 represents the number of all *Requirement Component* types used in the projects, but the types used in one project only can be considered as project specific. If only the *Requirement Component* types defined in at least two projects are taken into account, the distribution is different (Table 3, "5-2 projects" column and Figure 21). In this case, the total number of defined *Requirement Component* types is 130. The "Spatial Systems" category is clearly dominating (40%), and the importance of "Indoor Conditions" (18%) and "Safety" (17%) categories increases compared to the previous results. The number of *Requirements* in the "Comfort and Aesthetics" is clearly lower (9%), because one of the projects had in this category many detailed *Requirements* which were not used in other projects.

4.2.2 Requirements Data Types

Another observation of the analysis is the distribution of different data types for *Requirement Components*. In the analyzed projects the *Requirement Components* consist of five different data types: binary (yes/no), numeric (real or integer), enumerations, text, and links (hyperlink or external documents). Several *Requirements* include more than one data type. Thus, the total number of *Requirement Component* types in this view is 344. Textual descriptions are slightly dominant (33%), but also numeric *Requirements* are often used (30%). The amount of binary *Requirements* is very small (3%).

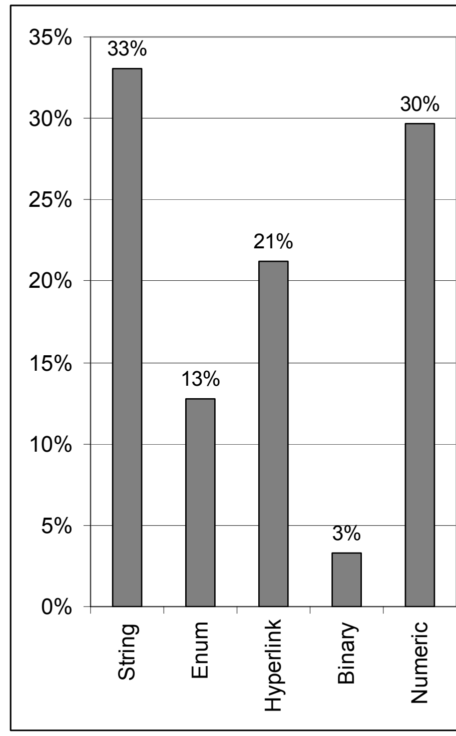

Figure 22: *Requirements* data types

The wide distribution of different data types means that the *Requirements Model* must support different *Requirements* data types to be usable.

4.3 Conclusions from the Requirements Analyses

The small portion of commonly used *Requirement Components* of the defined types (Section 4.1) raises interesting questions about the information content of the *Requirements Model Specification*, and also about the development of user-interfaces for *Requirements Management* software:

- Should the *Requirements Objects* be totally generic? In this case, users would define all *Requirements* based on the project's needs, including the direct and indirect links to the *Design Model*.
- Should the number of pre-defined *Requirements* in the *Specification* be very limited? In this case, there would be a small basic set of *Requirements* and users would add new *Requirements* based on the project's needs, which

would require them to also define the direct and indirect links to the *Design Model* for these *Requirements* explicitly.

- Or, should the *Requirements Model Specification* have a large number of different *Requirements*, which are seldom used? In this case, the resulting complexity of the underlying *Requirements Model Specification* could be addressed by a hierarchical user-interface, for example.

Section 6.1 analyzes the advantages and disadvantages of these approaches in more detail. My intuition is that an optimal solution is somewhere between these extremes. However, in this research I have selected an inclusive approach, and my *Requirements Model Specification* consists of a large number of *Requirements*, based on the reasons documented in Sections 6.1.5 and 6.4.3.

Regardless of the analysis method, it is clear that *Requirements* in the "Spatial Systems" category are the most often defined *Requirement Components* in the sampling of this research. In addition, most of the other frequently defined *Requirements* also relate to the *Spaces:* most "Indoor Conditions" and some "Safety" *Requirements*.

Based on my own professional observations, this is the case in most building projects. The *Spaces* are the core element of the end-user activities in the buildings. Thus, defining the *Requirements* related to the *Spaces* is a quite natural approach, and has a long tradition in the AEC industry. It is justified to claim that the *Spaces* are in many ways the reason for the buildings; they provide a controllable environment for the human activities.

Based on the described observations and preliminary analyses [*Kiviniemi et al., 2004* [72]] the next phase of my research, rapid prototyping (Chapter 5), concentrated on the *Space*-related *Client Requirements*.

5 Rapid Prototyping

The main conclusions from the two motivating case examples described in Section 1.2 and the *Requirements* analysis described in Chapter 4 are:

- The *Requirements* are not well documented and managed during the design process, and
- An active link between the *Requirements* and design tools could improve the process.
- The *Requirements* related to *Spaces* are the most commonly defined *Requirements* for building projects.
- Other types of *Requirements* vary strongly from project to project.

On the detail level the *Requirements* for different projects cannot be fully standardized (Section 3.2), but the framework, i.e., the *Requirements Model Specification*, must be project independent. However, the *Requirements Model* for a project can have project-specific *Requirements* (Section 3.5.1).

As defined in Section 1.3, the scope of my research is limited to the *Requirements Model* and its connection to the architectural *Design Model*. The derivation of *Indirect Requirements* to the systems and *Bounding Elements*, e.g., walls, windows and doors, from the *Direct Requirements* is within the scope of my research. Project types in the research are limited to office and laboratory buildings. Other building types are not in the scope.

An example which illustrates the *Direct and Indirect Requirements* is a *Room* which has *Requirements* for area, temperature and sound insulation. All these *Requirements* are linked to the *Room* (*Direct Requirements*). However, only the area *Requirement* affects the *Room* object itself directly, the other *Requirements* affect the conditions in the *Room* indirectly. The sound insulation *Requirement* affects the *Bounding Elements,* such as the walls and doors. The temperature *Requirement* affects primarily the HVAC system, but, depending on the design solution, it can also affect the *Bounding Elements*.

All 171 defined *Requirements* in the analyzed projects (Chapter 4) have direct links to the *Building Product Model*, e.g., all are *Direct Requirements*. 107 of these *Requirements* (63 %) have one or several indirect links, e.g., they cause *Indirect Requirements*. In total the *Requirements* defined in the analyzed projects include 127 indirect links.

Another aspect affecting the *Requirements Database* are the *Single-Value (SVR)* and *Multi-Value Requirements (MVR)*. *SVRs* can have only one value or reference for each *Space*, such as *Requirements* for noise level, maximum number of occupants, and maximum temperature. *MVRs* can have a number of different values or references in each *Space*, such as *Requirements* for activities, equipment, and adjacent *Spaces*. Table 4 documents the distribution of *SVRs* and *MVRs* in the analyzed projects. Table 10 documents the distribution of different *Requirement* types in the final *Specification*.

Table 4: Distribution of *SVR* and *MVR* types in the analyzed projects

	SVR	MVR
Requirement Attributes	74	
Requirement Descriptions	73	24
In total	**147**	**24**

Based on the analyses documented in Chapter 4, I limited the rapid prototyping to *Client Requirements* related to the *Spaces*. The purpose was to test the general idea to link *Requirements* to the objects in the *Design Model*. The points of departure for a technical solution to address these issues in the rapid prototyping were:

- The *Space*-related *Client Requirements* are defined and documented in the beginning of the process,
- The existing *IFC Specifications* contain the necessary elements to link *Space*-related *Client Requirements* to the *Building Product Model*,
- The existing *IFC Specifications* provide a connection between the *Spaces* and *Bounding Elements*,
- The existing IFC implementations provide a platform which can be used as a technical basis for the rapid prototyping to test the solution.

To explore the possible solutions to manage *Client Requirements*, I used rapid prototyping and implemented some different database structures to find a usable solution to store the *Space*-related *Client Requirements* in a structure which:

- Provides solutions to the problems identified in the LCE project (Sections 1.2.2.3 and 1.2.3),
- Supports *Cascading Requirements* from the *Space Program Type* to individual *Space Program Instances* (Figure 24),
- Enables a link between the *Requirements Model* and the existing *Building Product Model* (Figure 24).

As described in Chapter 1, the goal of this research is to improve the design process by providing a method to update and manage *Client Requirements* coherently, and give direct access from design software to the *Client Requirements* related to the on-going design task.

After the rapid prototyping phase, in the development of the final *Requirements Model Specification*, I discovered a solution for *Cascading Requirements* which simplifies the database structure significantly. This solution, based on the *Virtual Space Program Type*, is documented in Section 5.5.

5.1.1 Conceptual Model Structure

My solution to address these limitations is a concept that divides the instantiated *Model* of a project, i.e., project's data set, into four separate *Models* (Figure 23):

- *Requirements Model*
- *Design Model(s)*
- *Production Model(s)*
- *Maintenance Model*

This does not mean that the information structure, *Model Specification*, would have to be four separate *Models;* it can be one *Specification*. My *Requirements Model Specification* is using definitions from the current *IFC Specifications*. Thus, my *Requirements Model Specification* can be integrated with the *IFC Specifications*. However, the instantiated *Model*, i.e., project's data set, should be divided into several *Models*. In fact, the information content in the different design and contractor domains is so different that there is a need for several *Design and Production Models*, but this topic is not in the scope of my research. It is one of the proposed topics for future research (Sections 8.3.1.2 and 8.3.1.3).

The "PM4D Final Report" [*Kam and Fischer 2002* [73]] addressed the problem of one integrated *Model* in data exchange by pointing out the different content and structure of different design domains, although the report did not propose a solution for the problem. Also, John Haymaker recognizes the need for several *Models* in his Ph.D. research [*Haymaker et al., 2003* [74]]. However, to my knowledge, a similar division of a project's data set into these four main *Models* has not been published earlier and it is one of the main scientific contributions of my research (Section 8.1.3).

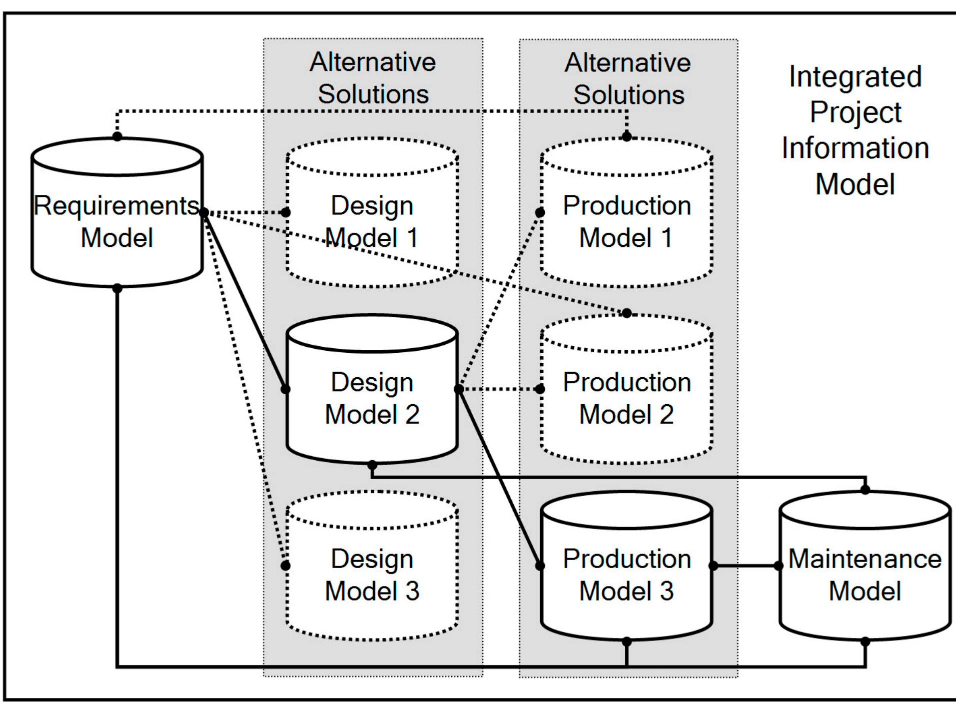

Figure 23: *Integrated Project Information Model*: *Model Hierarchy* and connections

There are several reasons for this separation of *Instantiated Models*:

- The data content and structure of these *Models* are different. For example, one *Space Program Instance* (Figure 24) can relate to a number of separate *Instances* with identical *Requirements* in the *Design, Production, and*

Maintenance Models. Similarly, for example, one slab or wall in the architectural *Design Model* can be several objects in the *Production Model*, or separate objects in the *Design and Production Models* can be one object in the *Maintenance Model*. However, my research scope covers the *Requirements* for architectural design only. The content of and links with other *Models* are topics for proposed future research (Sections 8.3.1.2 and 8.3.1.3).

- Although the *IFC Specifications* allow shared *Property Sets*, to my knowledge all IFC implementations are using instance-specific attribute sets, because the internal structures of design software do not support shared attributes. In practice it means that if the *Requirements* are stored in the *Design Model* the same *Requirements* are multiplied in all *Instances*, which can cause serious problems in the *Requirements Management* when the *Requirements* evolve and must be updated (Section 6.1.4).
- Typically, the *Project Team* produces several alternative design proposals which all should meet the defined *Requirements*. Thus, having one *Requirements Model* linked to the alternative *Design Models* is a logical structure instead of multiplying the same *Requirements* to different design alternatives, which would easily lead to *Requirements Management* problems. Similarly, there can be several alternative *Production Models* and finally a separate *Maintenance Model*. All four of these *Models* should be connected into one *Integrated Project Information Model* so that it is possible to access the content of the different *Models* and compare the alternatives at any stage of the process (Figure 23). My research focuses on the *Requirements Model* and its connection to the architectural *Design Model*.
- The flexibility of the *Requirements Model Specification* is greater if the *Models* are separated and connected with a "thin" link, e.g., there is only one identifier in both *Models* connecting the *Requirements* and *Design Objects* (Section 6.3.2). Adding or removing *Requirements* in the *Requirements Model Specification* does not change the design applications. In the prototype, the only element needed for the link of *Space Requirements* is an

ID in the *Space* object, which is supported by almost any design software. For *Indirect Requirements*, the functional demand is to recognize the connection between *Bounding Elements* and *Spaces*, which is supported by some commercially available *Building-Product-Model*-based software.

- Another reason for the separation is to make the distinction between *Requirements* and *Properties* clear; for example sound insulation is a *Requirement* for a *Space* in the *Requirements Model* and a *Property* of the *Bounding Elements* in the *Design Model*.
- Separation of *Requirements* and *Design Models* allows access control to *Requirements*; it is possible to show the information to designers but not allow them to modify *Requirements* if such control is wanted, for example, for project management or quality system purposes.
- *Requirements* are not attributes of *Design Objects* but independent entities, i.e., if the design changes so that a *Design Object*, such as a *Space*, is removed, its *Requirements* should remain unless the need for the *Space* has changed too. Otherwise reliable comparison of the design solutions against the *Requirements* is impossible.

A further important observation is that a *Space Program Instance* (*SPI*) in the *Requirements Model* has no *Geometrical Locations*, i.e., the *Requirements* for *Bounding Elements* can relate to one *Space* only. In the *Design, Production, and Maintenance Models* the *Bounding Elements* are always between two *Spaces*; either between two *Rooms* or as a part of the building envelope. This means that the *Requirements* for the *Bounding Elements* must be aggregated from the *Requirements* of the related *Spaces*. They cannot be defined directly for the building elements in the same manner as the *Space Requirements* relate to the *Spaces* (Figure 24).

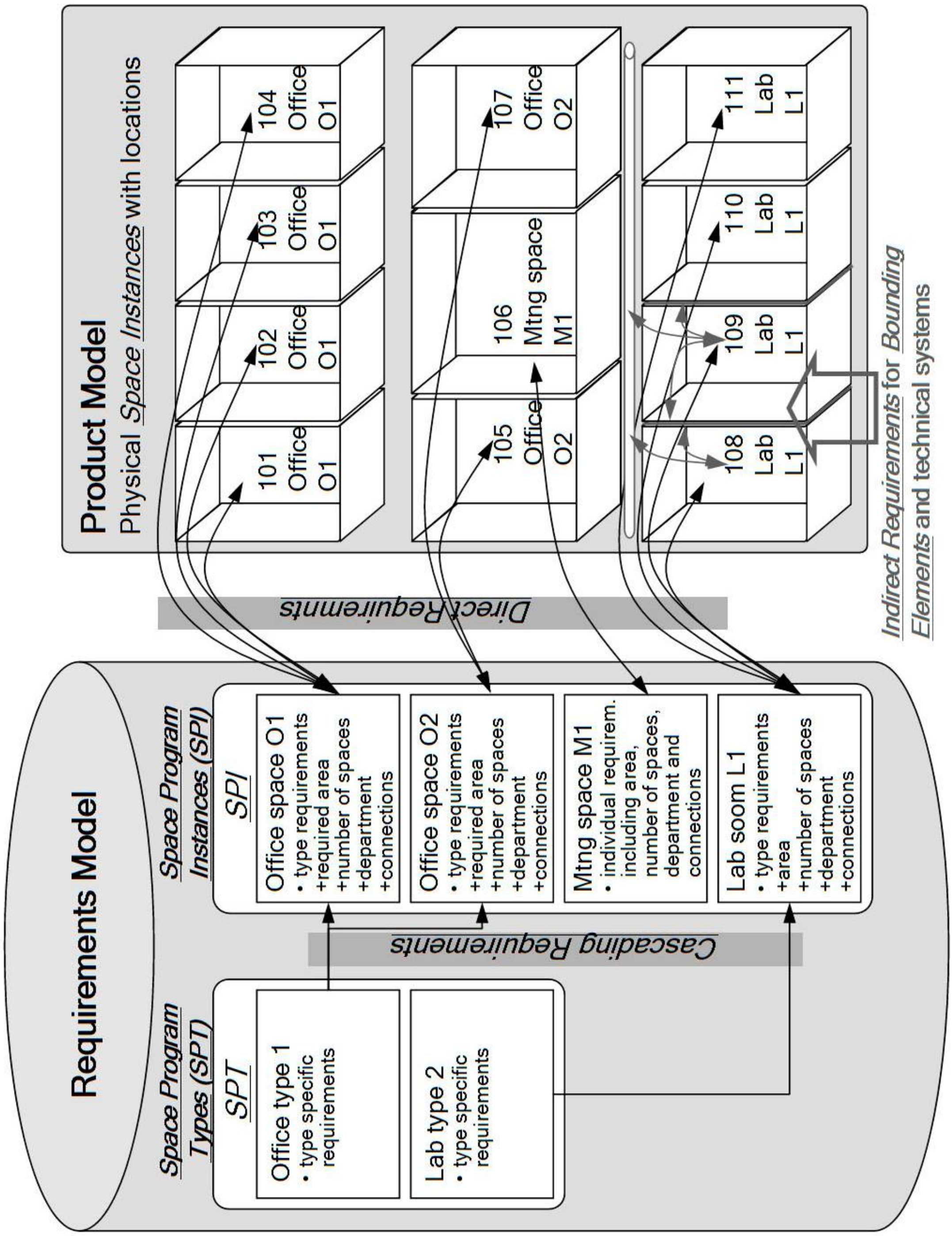

Figure 24: Concept used in the rapid prototyping to link *Requirements* to a *Design Model:* Relations between *Space Program Types (SPT)*, *Space Program Instance (SPI)*, physical *Space Instances* and *Indirect Requirements*.

5.2 Requirements Database Tests with LCE Project Data

The user-interface and database structure of the first prototype were based mainly on the *Building Program* documents of Stanford's Lucas Center Expansion. The prototype implementation was made in a MS Access 2002 database. The main criteria for the database structure were to provide a solution to the identified problems:

- Unique IDs for the *Spaces;* i.e., *Space Program Instance* (*SPI*) and all the *Space Instances* in the *Design, Production, and Maintenance Models* (*DPM Models*) referencing it must share the same ID ⇨ unambiguous identification.
- Use of *Space Program Type* (*SPT*) and *Cascading Requirements* ⇨ efficient and easy maintenance and updating of repetitive *Requirements*.
- Use of user-definable enumeration (list of values) instead of free text ⇨ coherent content.
- No default values which might inadvertently set wrong *Requirements*.
- Functionality to compare area *Requirements* with areas in design documents.
- Functionality to link external documents to the *Requirements Database*, e.g., to include also complex *Requirement Descriptions*, not only short text and numerical *Requirements*.

I tested several database structures in the development of the first prototype, mainly to find possible solutions for a structure and user-interface which could support *Cascading Requirements* from *Space Program Types* (*SPT*) to *Space Program Instance* (*SPI*) and *Multi-Value Requirements* (*MVR*). Figure 26 presents the final prototype structure for the first test case, Lucas Center Expansion, and also illustrates the terms "*Multi-Value Requirement*" *(MVR)* and *"Single-Value Requirement" (SVR).*

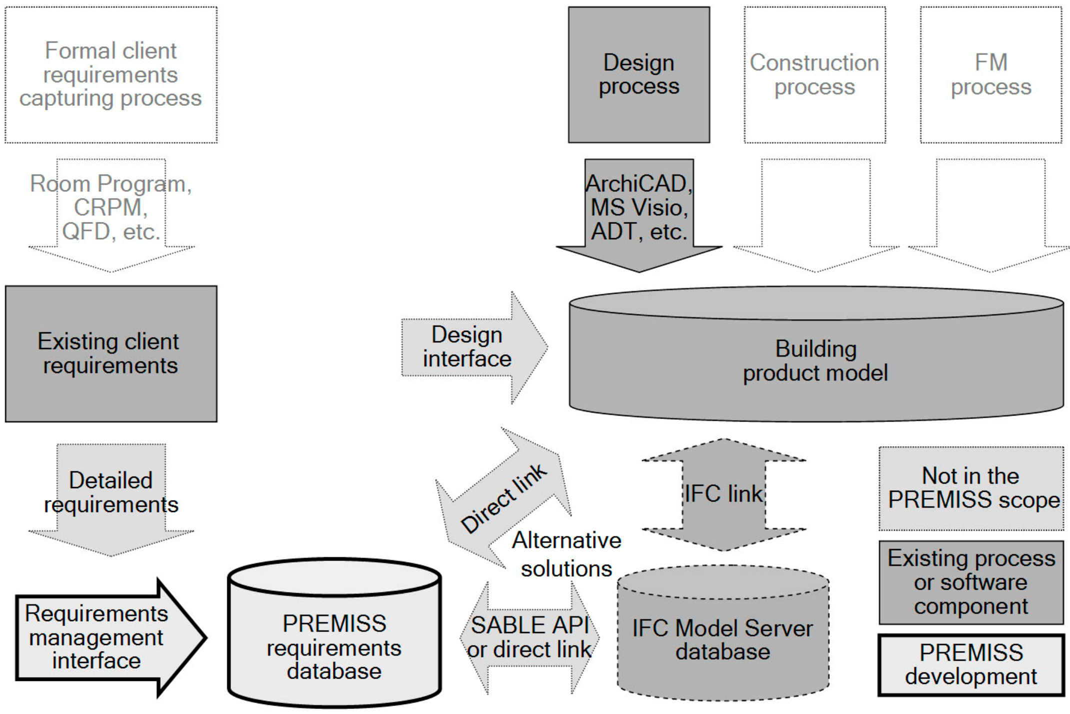

Figure 25: Rapid prototyping and its relations to existing solutions

As introduced in Figure 24, the key idea is the use of two main tables: *Space Program Type* (*SPT*) and *Space Program Instance* (*SPI*). In the prototype both have the same fields and references (*Shared Properties*, *ShP*) with the following exceptions:

- *SPI* can reference a *SPT* to "inherit" its *Cascading Requirements*, but the opposite relation is not possible,
- *SPI* can have a relation to department and other *SPIs*, but *SPT* cannot have these relations (*Instance-Specific Properties, ISP*)
- The *SPI* table contains a "NumberOfInstances" and "RoomName" fields, which are not in the *SPT* table (*ISP*)
- Only *SPT* has "RoomTypeDescription" and "RoomTypeDoc" fields, (*Type-Specific Properties*, *TSP*)

The *Requirements* used in the implementation are only one example of possible *Requirements*, and do not cover all possible building types or use cases. However, they can be categorized in two main groups:

- *Single-Value Requirements* (*SVR*) which can have only one value or reference for each *Space*, such as *Requirements* for noise level, maximum number of occupants, and maximum temperature.
- *Multi-Value Requirements* (*MVR*) which can have a number of different values or references in each *Space*, such as *Requirements* for activities, equipment, and windows.

For the following reasons this separation of *SVR* and *MVR* types is an important issue, and it defines the basic structure of the *Requirements Database*:

1) If all *Requirements* would be defined and implemented as *SVR* types, the database structure would not allow use of an unlimited number of *Requirements* for each *Space*, which is necessary for some *Requirement* types as described above.

2) If all *Requirements* would be defined and implemented as *MVR* types, the possibility of giving multiple values for all *Properties* could cause contradicting *Requirements*, such as several different maximum temperatures. In addition, the database structure would be more complicated, which could create performance problems, and the user-interface to the data would be more difficult to understand and slower to use, if all values were in sub-tables.

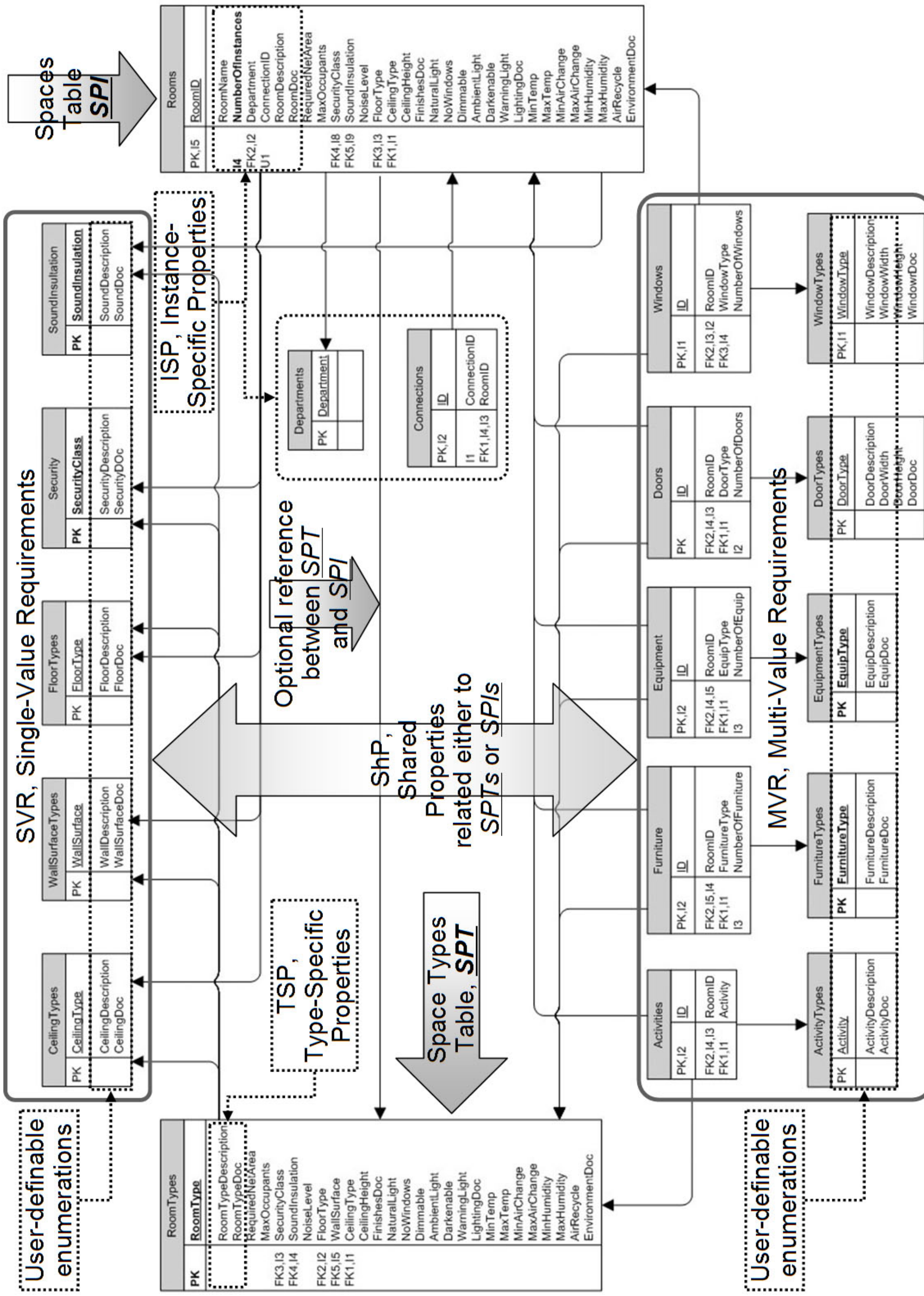

Figure 26: Database structure for the LCE project

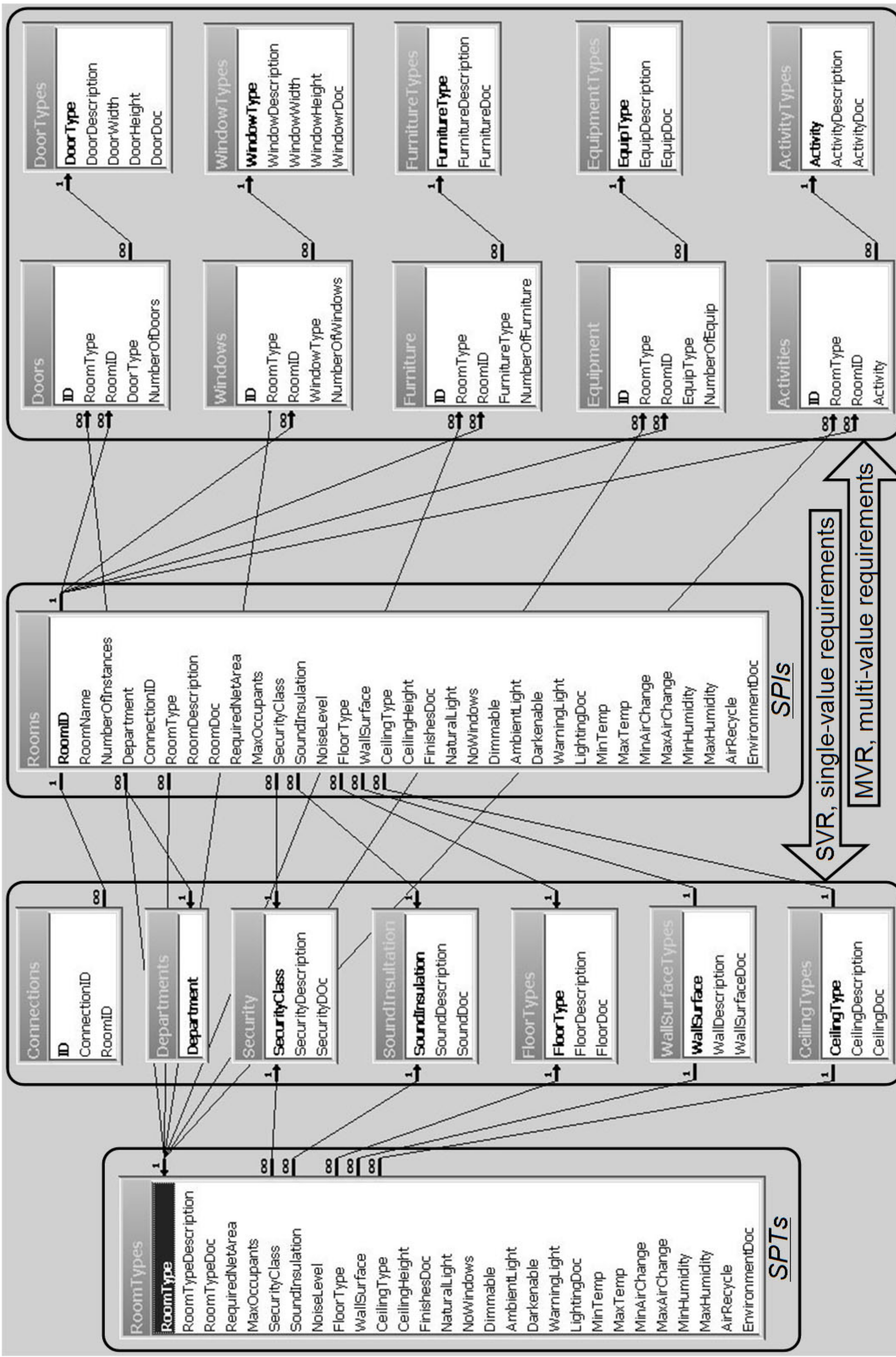

Figure 27: Relations in the LCE database

Figure 27 shows the one-to-one and one-to-many relations in the first prototype. “RoomType” and “RoomID” are the key links between different tables.

The structure forces the user to define unique IDs for each *Space Program Instance*, and I have defined all possibly repeated “free text” *Requirements*, such as departments, adjacent *Spaces*, equipment, activities, etc., as enumerations (user-definable lists) which prevents slightly different descriptions of the *Requirements* or references to non-existing *Spaces;* all problems identified in the LCE project data. I did not use the *Space Program Types (SPT)* in the LCE project database, because the LCE *Building Program* does not include any repeating types; I defined all *Space Requirements* in the LCE project database as separate *Instances (SPI).*

5.3 Test and Results with ICL Requirements Data

When starting to populate the database with the ICL project data, one observation came up almost immediately; “RequiredNetArea” and “MaxOccupants,” which were located in both the “RoomTypes” and “Rooms” tables in the LCE test, would have demanded extensive duplication of similar type definitions with different area and occupant values. Thus, I changed the database structure so that these *Requirements* were removed from the “RoomTypes” table and changed to *Instance-Specific Properties* in the “Rooms” table (Figure 28).

Otherwise the same database structure which was used in the LCE project test also worked for the ICL Headquarters project and enabled recording of the *Requirements* in a usable format; *Requirements* for 782 physical *Space Instances* are stored in 186 *SPIs* based on 51 *SPTs*. The maximum number of type references is 16, the average 3.8 and the median 2. The population of the database took about 3 hours, which is a reasonable effort.

My conclusion from the rapid prototyping phase is that the final database structure is sufficient proof of concept.

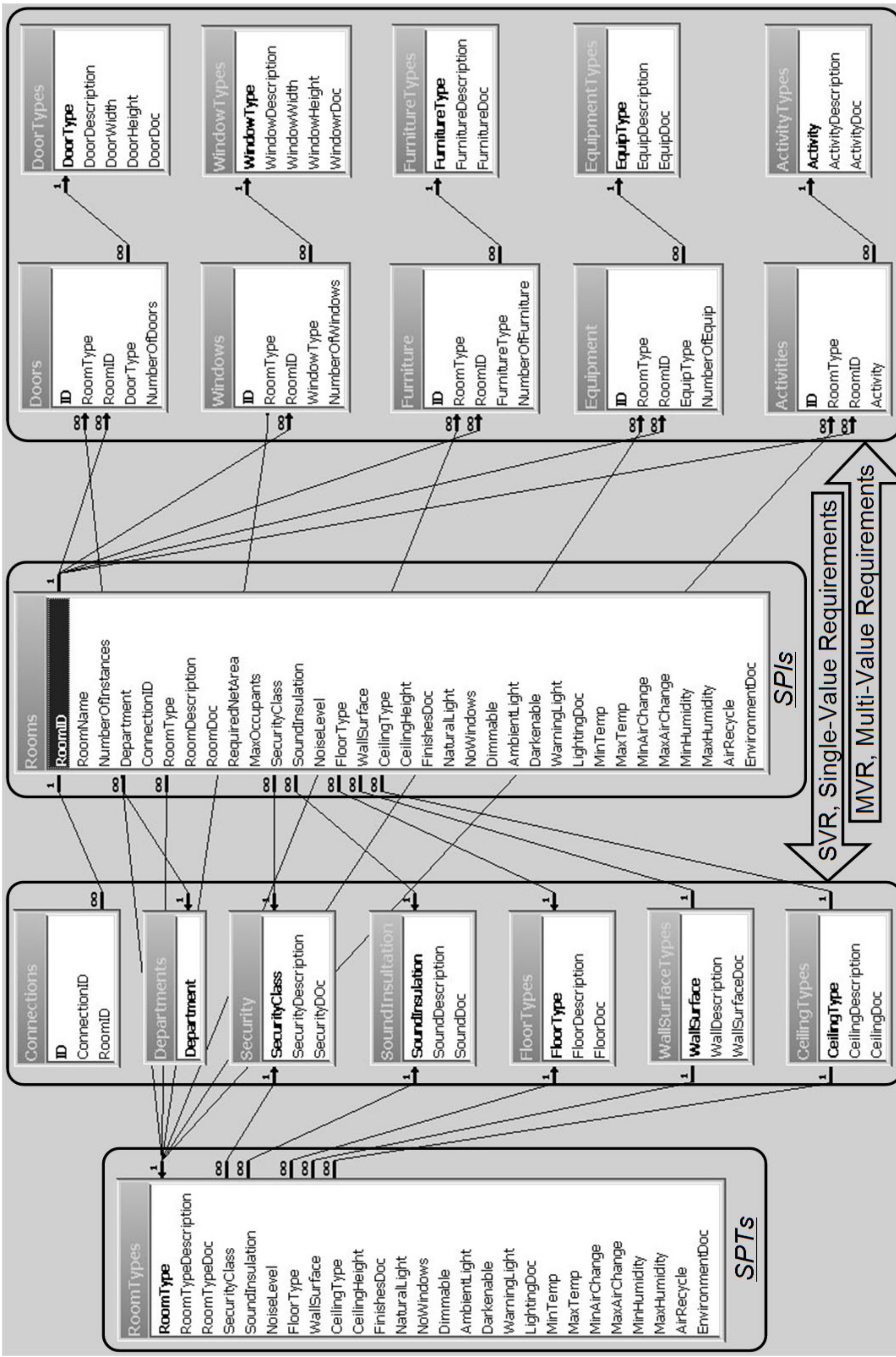

Figure 28: Relations in the ICL database

5.4 Data Groups and Conceptual Model of the Prototypes

During the two prototype tests I grouped the *Space*-related *Client Requirements* into the preliminary main sets presented in Figure 29 and Table 5.

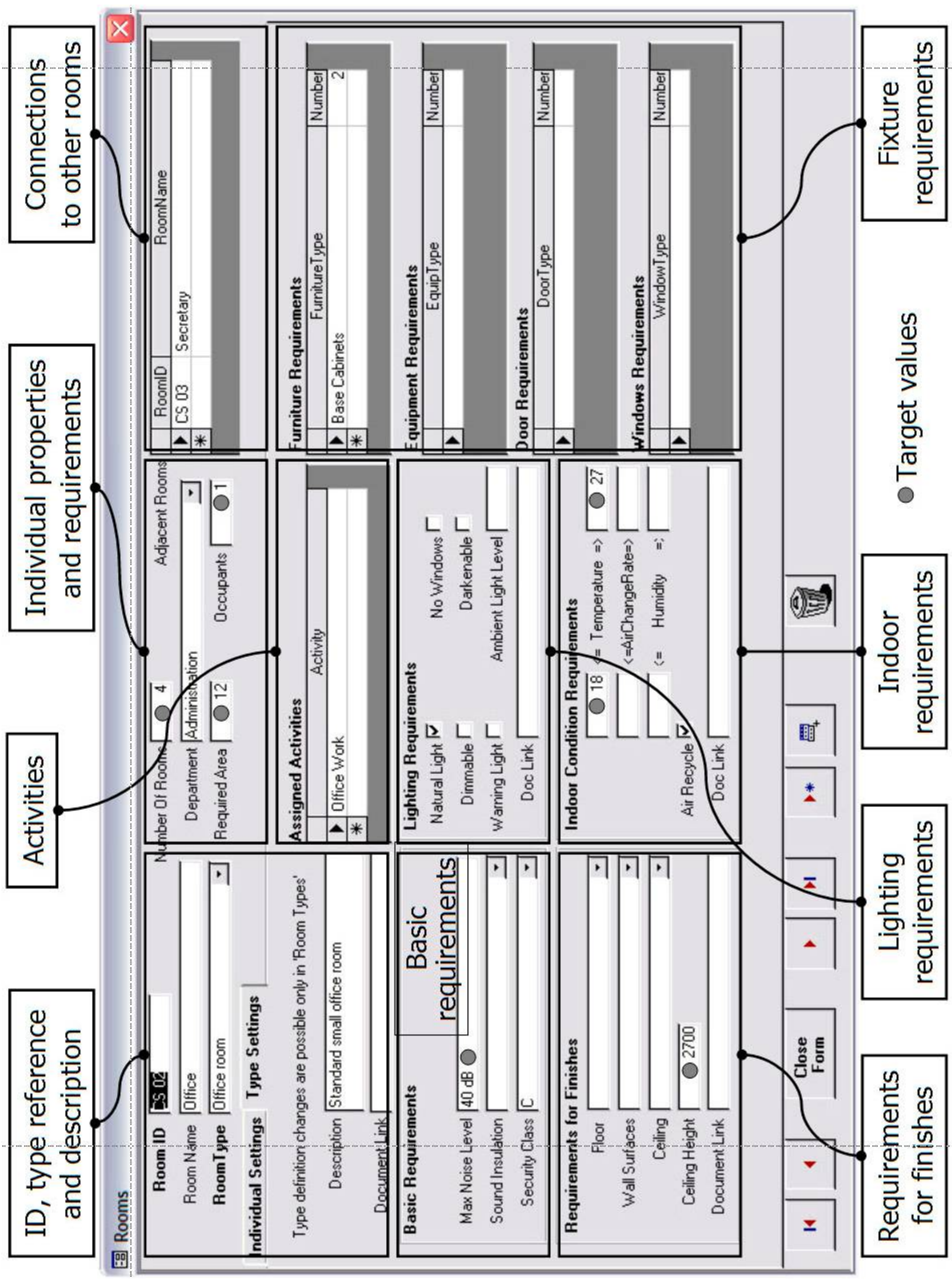

Figure 29: Form showing the *Requirements* groups in the rapid prototyping UI

The main groups are (Figure 29):

- Identification attributes (*Space* ID, type reference and description)
- Activities (use of the *Space*)
- Individual *Properties* and *Requirements* (number of *Spaces*, area, occupants)
- *Requirements* shared with a possible type:
    - Basic *Requirements* (sound, security)
    - Surface *Requirements*
    - Lighting *Requirements*
    - Environment *Requirements* (temperature, humidity, etc.)

Fixture *Requirements* (windows, doors, furniture, equipment)

Table 5 contains information on how often these *Requirements* were used in these two projects. Only three *Properties* or *Requirements* were defined for all *Spaces* in all databases: name of the *Space*, area of the *Space*, and number of the *Spaces*. Also department (98 %) and *Space Type* (73 %) were defined often, but all other *Properties* or *Requirements* only seldom. The comprehensive analysis of *Requirements* types and their usage is in Chapter 4.

Table 5: Database elements, types and usage in test projects

Property	Requirement	Room	RoomType	SVR	MVR	Data Type	Bounding elements	Systems	*LCE, PM*	*LCE, PA*	*ICL*	Average
Identification and overall definition												
RoomID		m		x		UID, string	x	x	*62%*	*92%*	*100%*	88%
RoomName		o		x		String			*100%*	*100%*	*100%*	100%
RoomType		o	m	x		UID, string			*46%*		*100%*	73%
RoomDescription		o	o	x		String						
Document		o	o	x		Hyperlink						
Individual properties and requirements												
Department		o		x		Enum			*92%*	*100%*	*100%*	98%
	NumberOfRooms	m		x		Integer			*100%*	*100%*	*100%*	100%
	RequiredArea	o		x		Real				*100%*	*100%*	100%
	MaxOccupants	o		x		Integer		x			*100%*	50%
Basic Properties												
	MaxNoiseLevel	o	o			Integer		x	*38%*			19%
	SoundInsulation	o	o			Enum	x	x				
	SecurityClass	o	o			Enum	x	x				
Connections, activities, furniture, equipment, doors and windows												
	Connections	o			x	Ref to UID			*46%*		*28%*	37%
	AssignedActivities	o	o		x	Enum list		x	*85%*			42%
	Furniture	o	o		x	Enum list			*62%*		*1%*	31%
	Equipment	o	o		x	Enum list		x	*38%*		*3%*	21%
	Doors	o	o		x	Enum list	x	x	*100%*			50%
	Windows	o	o		x	Enum list	x	x				
Finishes												
	Floor	o	o	x		Enum			*92%*			46%
	Walls	o	o	x		Enum			*100%*			50%
	Ceiling	o	o	x		Enum			*100%*			50%
	Ceiling height	o	o	x		Real		x	*92%*			46%
Document		o	o	x		Hyperlink						
Lighting												
	NaturalLight	o	o	x		Yes/No		x	*77%*			38%
	NoWindows	o	o	x		Yes/No	x	x				
	Dimmable	o	o	x		Yes/No		x				
	Darkenable	o	o	x		Yes/No		x				
	WarningLight	o	o	x		Yes/No		x				
	AmbientLightLevel	o	o	x		Real		x				
Document		o	o	x		Hyperlink	x	x				
Environmental Conditions												
	MinTemperature	o	o	x		Real	x	x	*46%*			23%
	MaxTemperature	o	o	x		Real	x	x	*46%*		*2%*	24%
	MinAirChangeRate	o	o	x		Real		x	*92%*			46%
	MaxAirChangeRate	o	o	x		Real		x				
	MinHumidity	o	o	x		Real	x	x				
	MaxHumidity	o	o	x		Real	x	x				
	AirRecycle	o	o	x		Yes/No		x	*62%*			31%
Document		o	o	x		Hyperlink	x	x				

m = mandatory field
o = optional field

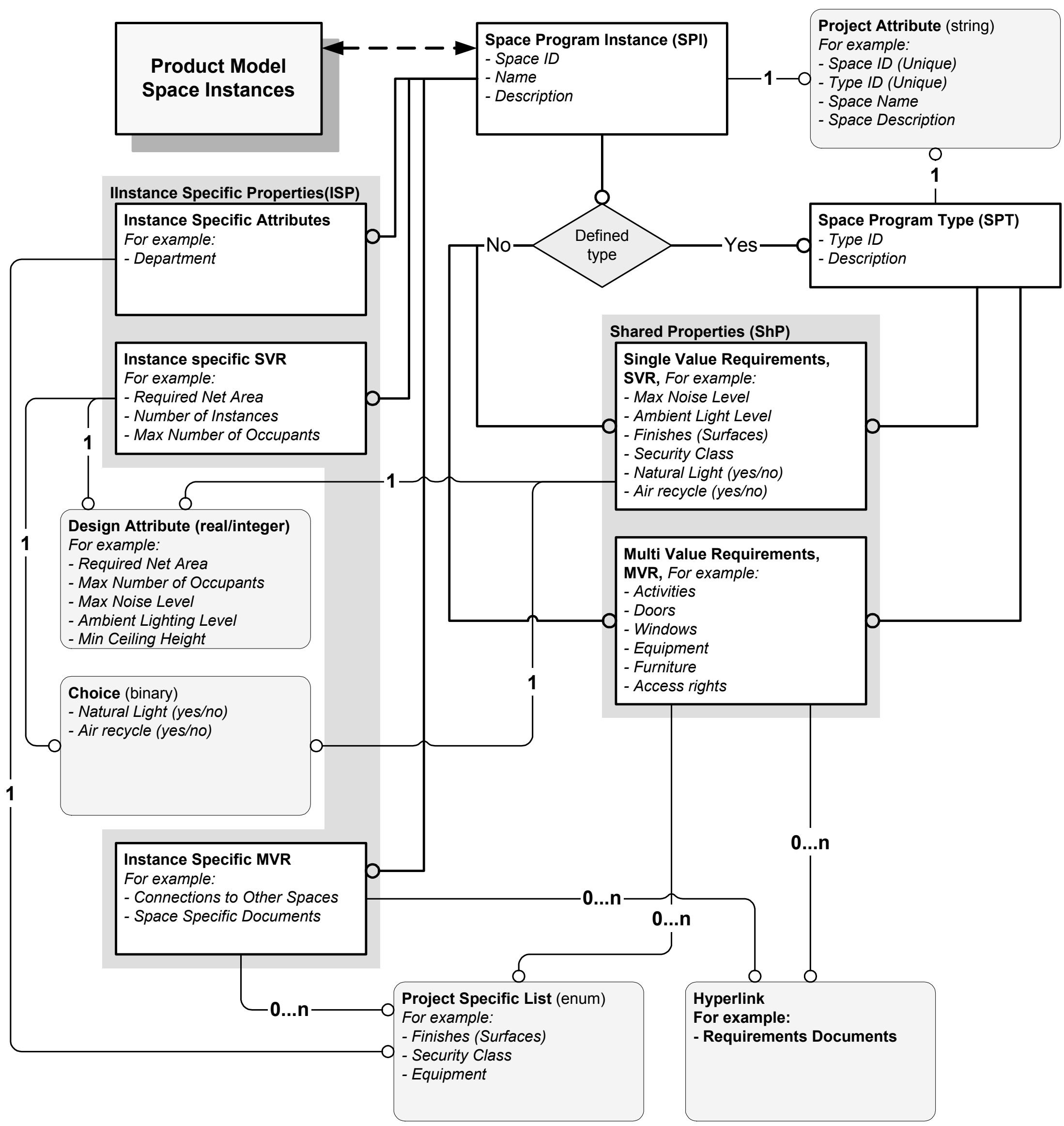

Figure 30: Conceptual structure for *Space*-related *Client Requirements* in the rapid prototyping

Based on these results I made the conceptual structure presented in Figure 30. The main ideas in the *Conceptual Model* for *Space*-related *Client Requirements* in the rapid prototyping are:

- Use of *Space Program Type* to define *Requirements* which are identical for several *Space Program Instances* in the *Requirements Database*
- Separation of the *Requirements* which are always *Instance-Specific Properties* (*ISP*) and which can be *Shared Properties* (*ShP*) defined either at the *SPI* or *SPT* level.

- Separation of the *SVRs* and *MVRs*, as described in Section 5.2.
- Flexible framework which enables additional project-specific *Requirements* (Sections 3.5.1)

However, in the development of the final *Requirements Model Specification* it became clear that the use of *ShP* type is not necessary. Section 5.5 documents the final solution.

5.5 Simplified Database Structure and Conceptual Model

The key idea to simplify the database structure is the use of *Virtual SPTs* to create an individual *Space Program Type* for *Space Program Instances* always when the user defines a *SPI* and does not associate it to some defined type. This prevents duplication of the same data fields in *SPI* and *SPT* databases and simplifies the database structure (Figure 31) and *Conceptual Model* (Figure 32) significantly compared to the rapid prototyping database structures (Figure 26 – Figure 28) and *Conceptual Model* (Figure 30). The ID of the *Virtual SPT* can be based automatically on the ID of the SPI.

This simplification does not change the basic idea presented in Figure 23 and the information content in the database is exactly the same as in the databases used for the rapid prototyping phase. The only addition in Figure 23 would be a *Virtual SPT* for the Meeting Space M1, and it would be generated automatically without the need for end-users to know about the concept.

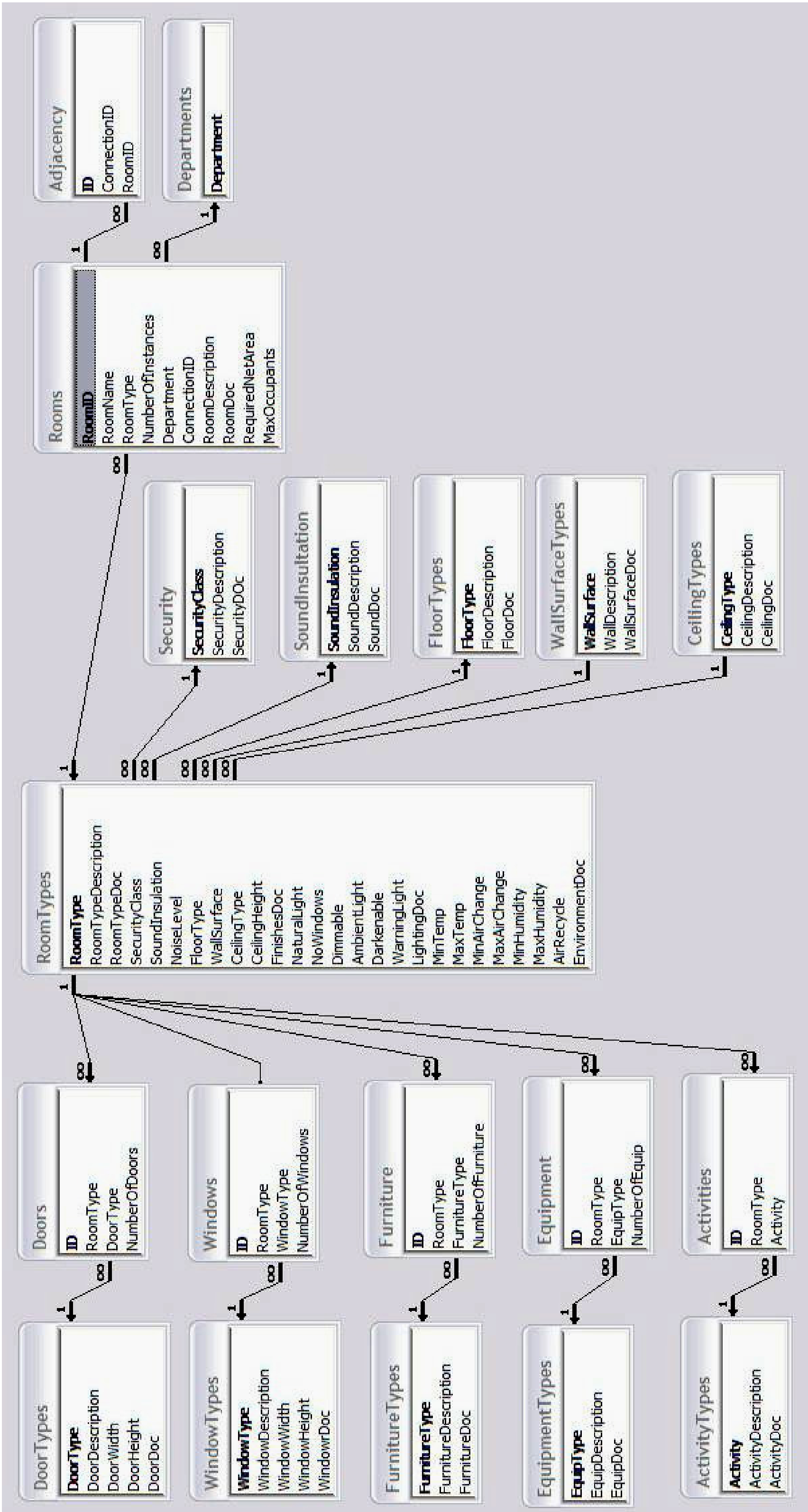

Figure 31: Simplified database structure based on *Virtual SPT*

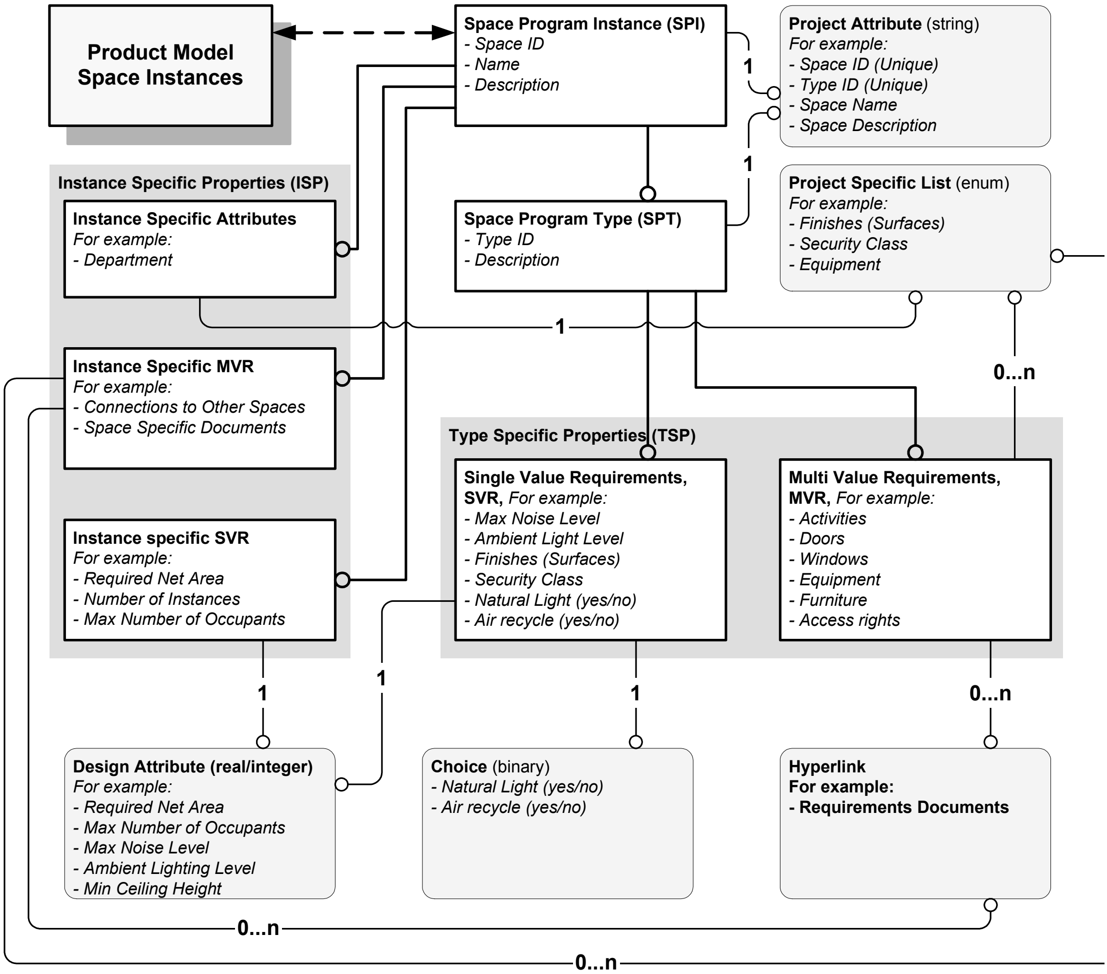

Figure 32: Conceptual structure for simplified database structure

5.6 Connection to the Building Product Model

I did not implement the actual connection of the *Requirements Database* to the *Building-Product-Model*-based design software in the rapid prototyping; I only developed a mock-up presenting the idea of such a connection from design applications to the *Requirements Database*. However, I made the rapid prototyping effort with a thorough understanding of the *IFC Specification* and of *Building-Product-Model*-based design software capabilities. The basic idea is that by selecting objects, e.g., *Spaces* and *Bounding Elements*, in the design software the user can see all related *Requirements* in the *Requirements Database* (Figure 33 – Figure 36).

In this solution, "RoomID" is the connecting element between the *Requirements Database* and the *Building Product Model*. The links between the *Space Requirements* and *Space Instances* in the *Building Product Model* are direct, but the *Bounding Elements* related to a *Space* must be identified in the *Building Product Model* and the connection to the *Requirements Database* is based on the "RoomID" of identified *Space*.

The user-interface mock-up in Figure 33 – Figure 36 demonstrates how to access the *Requirements Database* from design software by adding a *Requirements View* to its user-interface. This functionality is naturally a requisite for the use of *Requirements* directly from the design software, but is not necessary for the use of the *Requirements and Design Models*. It is possible to make the links and comparisons between these two *Models* just for control purposes, as demonstrated in Sections 1.2.1 and 7.1.1.

Depending on the use scenario, the modifications of the *Requirements* from the design interface can be either allowed or denied; in some projects the *Client* might delegate the *Requirements Management* to the designers, in some projects it might be the task for the PM or the *Client's* own representative. The access control for the *Requirements Database* is one of the reasons to separate the *Requirements and Design Models* (Section 5.1.1).

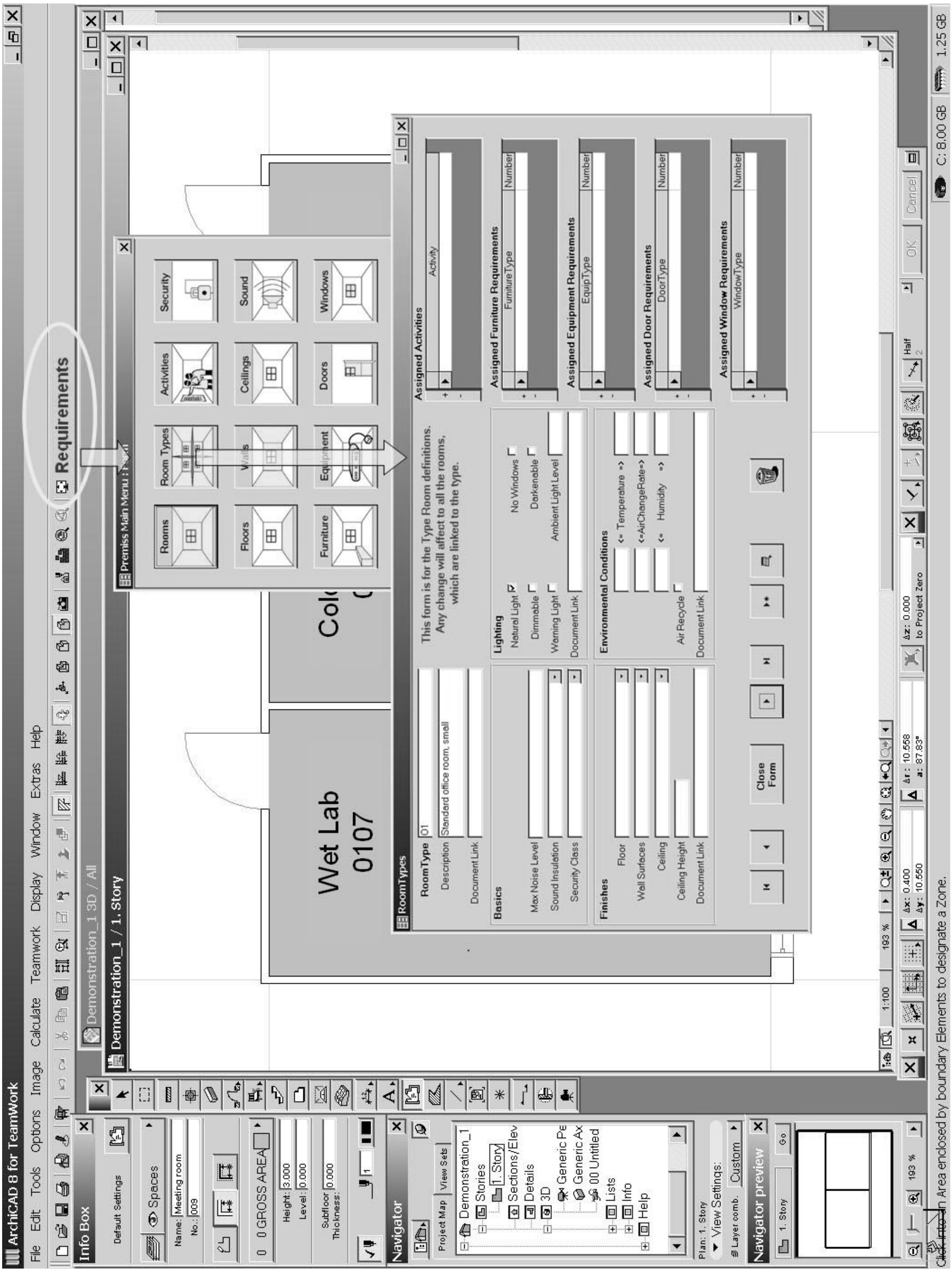

Figure 33: *Requirements* UI mock-up: *Requirements Management* interface from design software to the *Requirements Database* definitions, such as *Spaces*, *Space Types*, activities, security, and equipment.

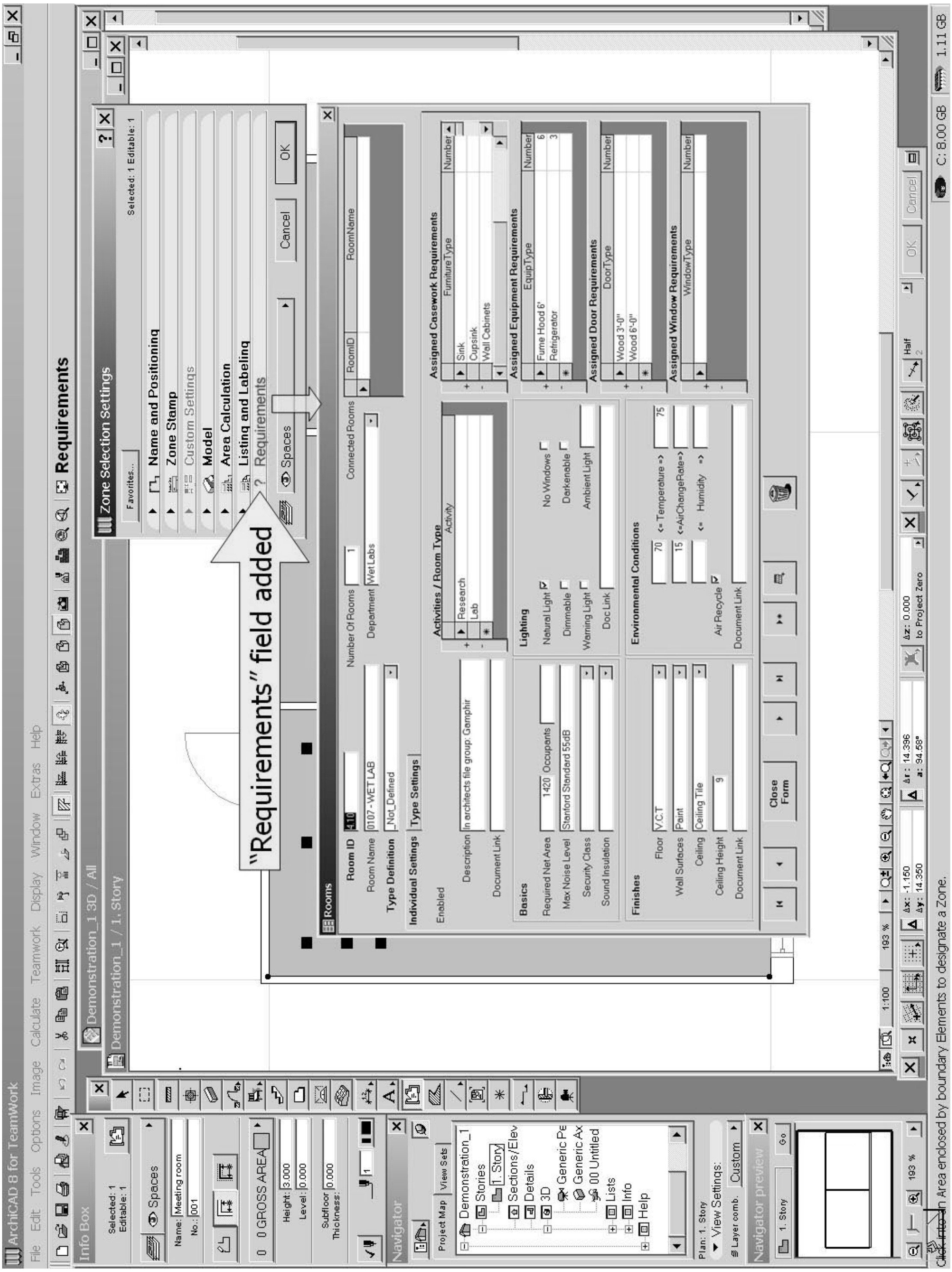

Figure 34: *Requirements* UI, mock-up: *Space*
By selecting a *Space* and then the *Requirements View*, the software shows all the defined *Requirements* for the selected *Space*.

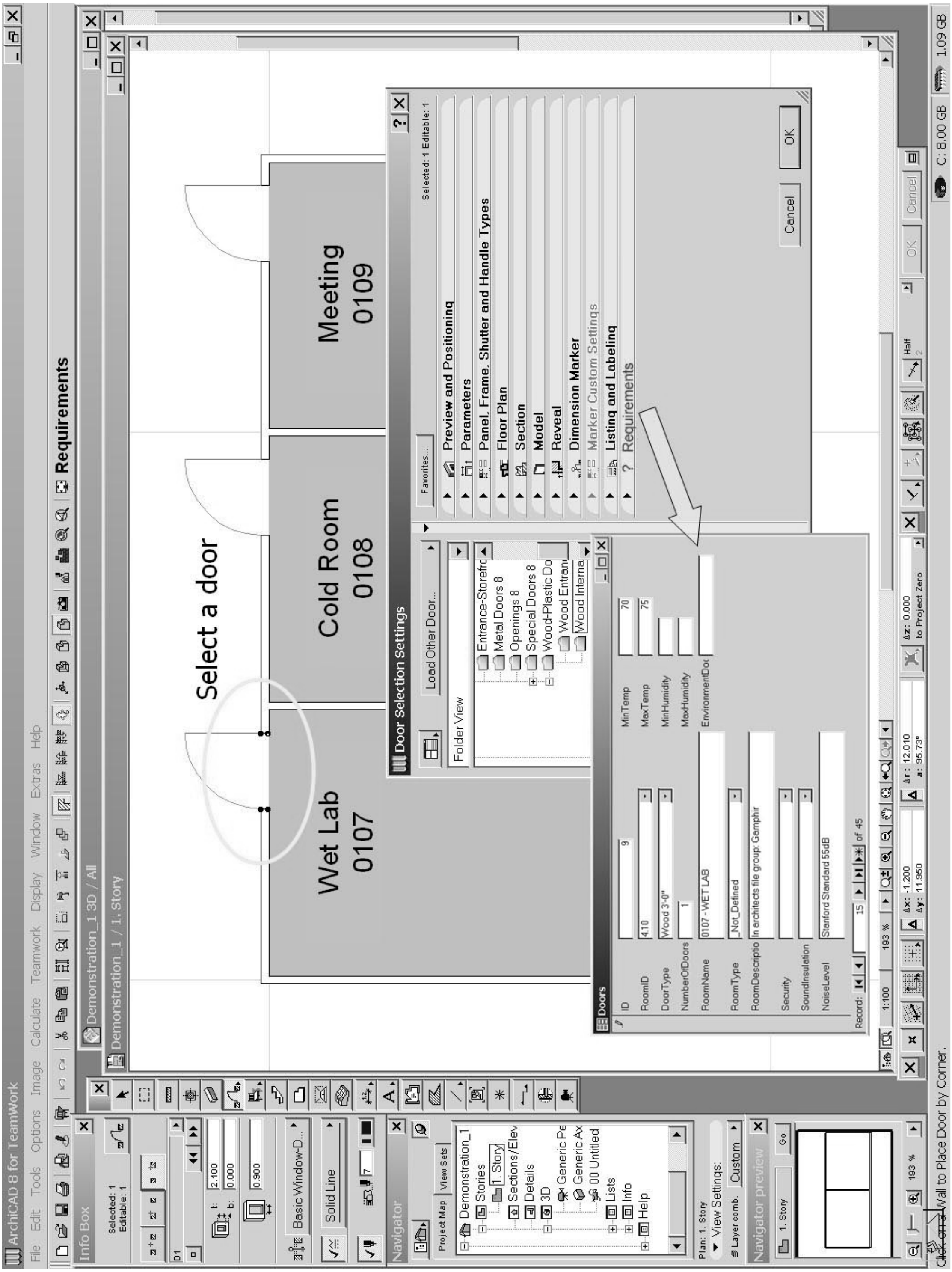

Figure 35: *Requirements* UI, mock-up: Door
By selecting a door and then the *Requirements View*, the software shows all door-related *Requirements* (Table 5) from the related *Space*(s).

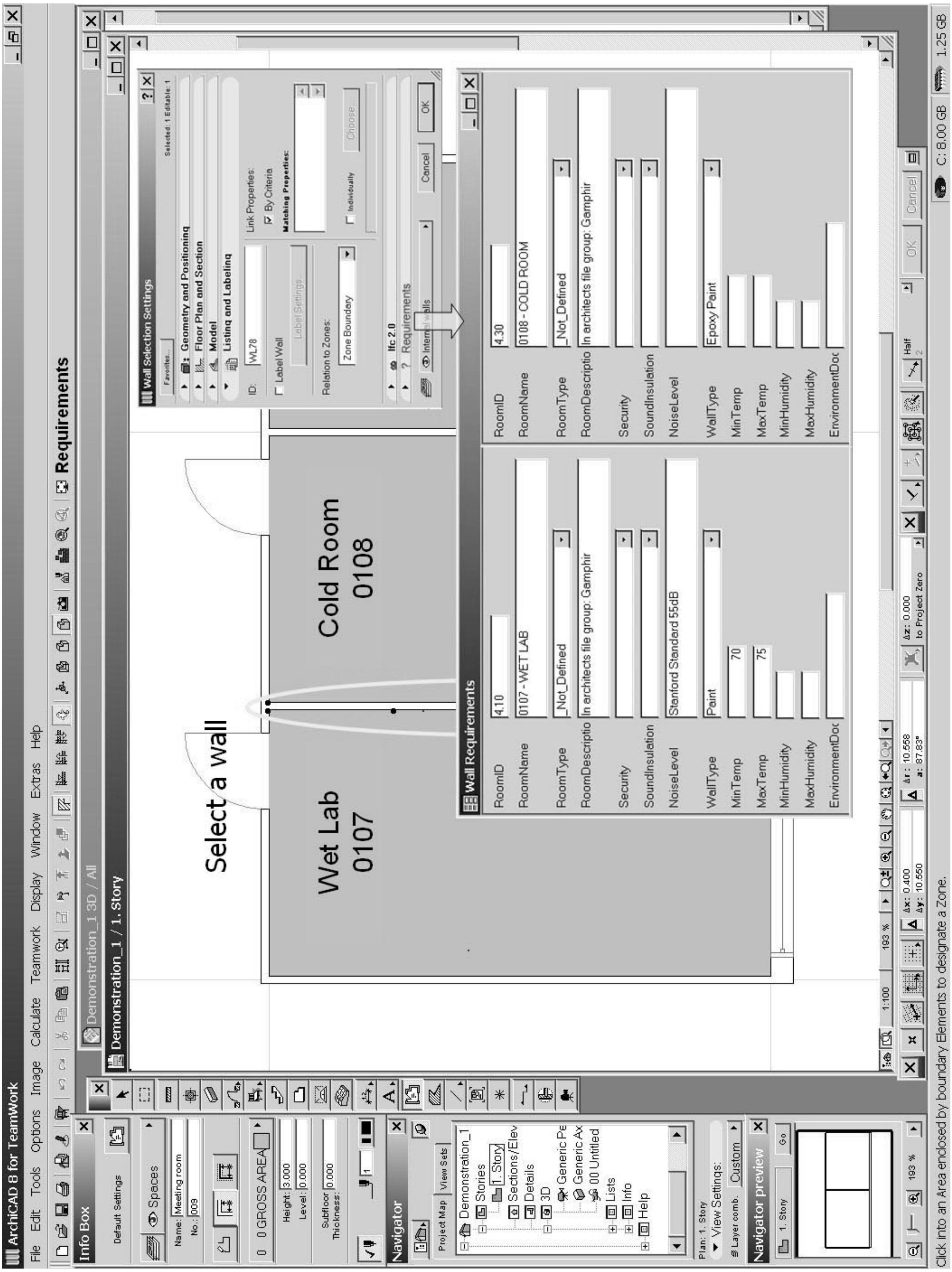

Figure 36: *Requirements* UI, mock-up: Wall
By selecting a wall and then the *Requirements View*, the software shows all wall-related *Requirements* (Table 5) from the related *Space*(s).

5.7 Conclusions from the Rapid Prototyping

The main results of the rapid prototyping phase were:

- The basic concept of an *Integrated Project Information Model* divided into four main *Models* on the *Instance* level: *Requirements, Design, Production and Maintenance Models*.
- A structure for *Cascading Requirements: Space Program Type (SPT) and Space Program Instance (SPI).*
- The detailed data content and types of *Space*-related *Client Requirements:* how the *Requirements* should be divided into *SPI* and *SPT* levels.
- Proof of concept: The implementability of the *Requirements Database* and the basic idea of the link between the *Requirements and Design Models*.
- The structure and content needs for the formal *Requirements Model Specification* documented in Chapter 6.

In addition, the rapid prototyping phase highlighted some implementation issues discussed in Sections 6.4 and 8.2.2 and in Appendix C.

6 Requirements Model Specification

This Chapter documents my *Requirements Model Specification* in detail. Section 6.1 documents features of a good solution for a *Requirements Model*, describes the basic concepts of my *Requirements Model* and analyzes three alternative *Requirements Model* solutions: (1) generic *Requirements* objects, (2) *Requirement Attributes* attached to the *Design Objects*, and (3) detailed *Requirements Model Specification*. Section 6.2 analyzes in detail the usefulness of existing elements in the *IFC Specifications* for the *Requirements Model Specification*. Section 6.3 describes the *Requirements Model Specification* in detail, and Section 6.4 documents the expanded *Implementation View* of the *Requirements Model Specification.*

The *Requirements Model Specification* defines the structure of the *Model*, and it is intended as the basis for software development; for AEC professionals it is useful only if implemented into software products.

As explained in Section 3.1, a semantic building *Model*, i.e., a *Building Product Model*, is a mandatory starting point to link a *Requirements Model* to design solutions. Traditional drawings and other design documents are not software interpretable. Thus, the *Requirements* cannot be linked in a meaningful way to their content. The *IFC Specifications* are the official and de-facto standard for *Building Product Model* in the AEC industry (Section 3.4), and thus they provide a good starting point for a link between the *Requirements and Design Models*, although the *Requirements Model Specification* itself is independent of the *Design Model Specification* as discussed in Section 5.1.1. However, the integration on the *Specification* level provides some benefits, such as the ability to use existing resources and to define the links unambiguously.

I propose my *Requirements Model Specification* as a basis for an extension of the *IFC Specifications* (Section 8.2.4). Thus, I use the existing IFC elements when applicable. However, my *Requirements Model Specification* is not a part of the *IFC Specifications;* the approval process for that demands official acceptance and consensus about the *Requirements Model* content in the IAI, and also significant integration work. Thus, I use the “New” prefix instead of the standard “Ifc” prefix in all new elements in my *Specification*. Otherwise, the notation I use in this Chapter follows the naming convention of the *IFC Specifications*. I write all the object names chained with a capital letter in the beginning of each element, such as IfcSpace and HvacSystem. This is also the reason for the use of the term “BuildingStorey” in some places; in the *IFC Specifications* the object is named “IfcBuildingStorey.” In the normal text I use the US spelling “story.”

The formal language I have used for the *Requirements Model Specification* is Express (ISO Standard 10303 Part 11), the same language which is used for the *IFC Specifications* and other product model *Specifications* of ISO.

However, all the concepts of my *Requirements Model Specification* are applicable to any semantically meaningful *Building Product Model* which includes representation of the following entities: Project, site, building, building story, *Space*, building envelope and various technical systems. Only the programming-language-specific definitions would be different.

6.1 Conceptual Requirements Model

6.1.1 Features of a Good Solution for the Requirements Model

Based on the case studies (Sections 1.2 and 7.1), *Requirements* analysis (Chapter 4), and rapid prototyping (Chapter 5) features of a good solution for the *Requirements Model* are:

- Separation of *Requirements and Design Models*. The reasons for this are (Section 5.1.1):
    - Different structures and content of the *Models*.
    - Need to compare alternative design solutions to the *Requirements*.
    - Access control: only authorized users can change *Requirements* although the *Requirements* are visible to the whole *Project Team*.
    - Reliability: *Requirements* are not attributes of *Design Objects* but independent entities, i.e., if the design changes so that a *Design*

Object, such as a *Space*, is removed, its *Requirements* should remain unless the need for the *Space* has changed too.

- Automated linkage between the *Requirements and Design Models:*
    - The links between the *Models* provide possibilities to compare *Requirements* and design solutions rapidly and efficiently. However, one project can include thousands of links. Thus, the solution must enable automated creation and maintenance of the links.
- One shared *Requirements Model:*
    - A shared *Requirements Model* which is accessible to all *Project Team* members can improve the communication between stakeholders.
- Organized *Requirements* structure:
    - An organized structure enables different views of the *Requirements* and the possibility of finding the relevant *Requirements* related to different tasks easily.
- Accountability:
    - Access control by the *Requirement's* owner provides the possibility of tracing the source and history of each individual *Requirement:* when and who changed the *Requirement*.
- Granularity:
    - Each *Requirement* can have its own owner, source and history.
- Flexibility:
    - Even if a *Requirements Model Specification* is inclusive it is highly unlikely to cover everything. However, *IFC Specifications* already have a method to add attributes to the defined objects without the need to change the *Specification: Property Sets*. If the *Requirements Model Specification* is an extension of the *IFC Specifications*, the flexibility is inherited from the common structure.
- Support of the *Cascading Requirements* for *Spaces:*
    - The use of *Space Types; Requirements* are not multiplied for all the *Space Instances*.

6.1.2 Basic Concept: Direct and Indirect Requirements

The basic concept of my *Requirements Model* is very simple. The starting point is the defined *Project Requirements*. These *Project Requirements* can be organized into subsets of *Requirements* which are related to a specific *Design Object* on some level; project, site, building, building story, *Space*, and systems. In these sets of *Requirements* there can be subsets of *Requirements* which affect some system or systems serving this specific *Design Object*. In the *Requirements* for *Spaces*, there can also be subsets of *Requirements* which affect the *Bounding Elements* of the *Space* (Figure 37).

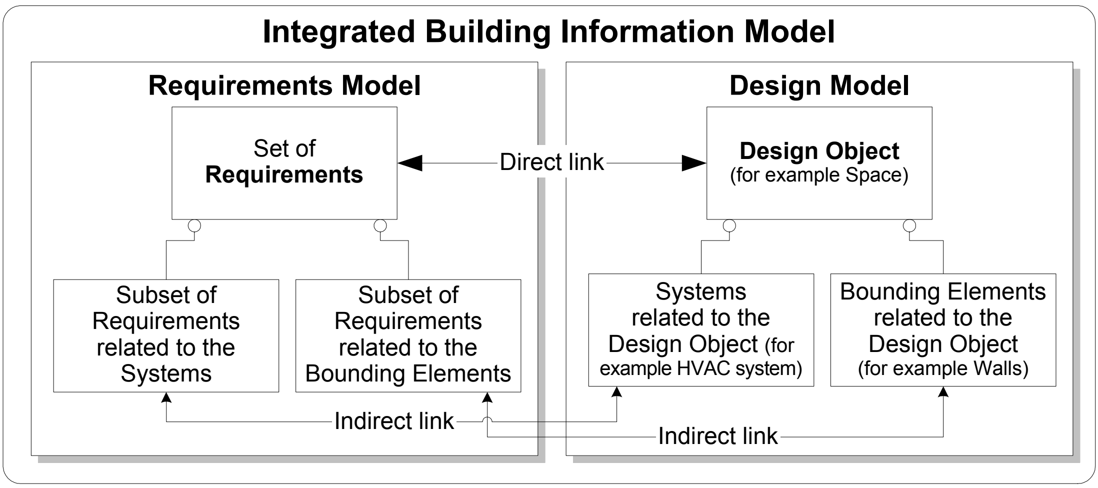

Figure 37: *Conceptual Requirements Model*

A practical example to illustrate this is a *Room* which has the following *Requirements*:

- Area 20 m^2
- Temperature 19–25 °C
- Sound insulation 40 dB

All these *Requirements* are linked to the *Room* (*Direct Requirements*). However, only the area *Requirement* affects the *Room* object itself directly, the other *Requirements* affect the conditions in the *Room* indirectly. The temperature *Requirement* affects primarily the HVAC system, but, depending on the design solution, it can also affect *Bounding Elements*. For example, if the *Room* has windows and is located on the South side of the building, it is obvious that the windows should have some shading mechanism so that the cooling system can maintain the temperature in the required area. The sound insulation *Requirement* definitely affects the *Bounding Elements,* the walls and doors, possibly also the windows depending again on the design solution. However, it is not possible to know these potential effects when the *Requirements* are defined, because the design solution does not yet exist. Even during the design, the situation can change, if, for example, the *Room* is moved to another *Location*. Therefore, the *Requirements Model* must contain the links to all elements which the set of *Requirements* potentially can affect. This has fundamental effects on the *Requirements Model Specification*, as well as in its implementation and use; if the indirect links are not predefined in the *Specification*, the *Requirements Management* software developer or the end-user of the software must define them.

The only physical link between the *Models* is the direct link; in this example it is the link between the *Requirements Object* and the *Space* object. In addition, the information about necessary indirect links is in the *Requirements Object*, and the software recognizes the affected objects inside the *Design Model*; in this example these affected objects are the *Bounding Elements* and HVAC system which relate to the *Space* (Section 6.1.6).

There are three alternative ways to define a *Requirements Model* for the described purpose:

1) Use of generic *Requirements Objects*
2) Use of attribute sets which contain the *Requirements* and attach them directly to building elements
3) Use of a detailed *Requirements Model Specification* which specifies the relations between the *Requirements* and building elements.

Sections 6.1.3–6.1.5 document the benefits and problems of each of these alternatives.

6.1.3 Generic Requirements Object

As mentioned in Chapter 4, one possible solution for a *Requirements Model* could be a totally generic, consisting of one *Requirements Object* which could be linked to any objects in the *Design Model*. The obvious advantage of this solution is its simplicity and flexibility; one object could contain any *Requirements*.

A *Generic Requirements Object* could consist of a couple of data fields which could contain a “place holder” for different data types; numeric, textual and hyperlink fields would cover most relevant needs in documenting *Requirements* (Section 4.2.2). The user could link such a *Generic Requirements Object* to any *Design Object* with a direct link; this would demand some additional effort from the end-users of the system but might still be possible to do as a part of regular project work.

However, the main problem is in the indirect linkage. It is difficult to anticipate all the objects which a *Requirement* can potentially affect, and it is not likely that the designers would want to use a system where they would have to define all the indirect links for every *Requirement*. In addition, any grouping of *Requirements* would have to be done manually. This link definition and grouping effort would increase the amount of work significantly; it would demand in each project similar work to that which I have presented in Section 6.3 defining the *Requirements Model Specification*. Creation and maintenance of such a *Model* during the design and construction process would be practically impossible; the additional work versus the benefits would probably not give a reasonable pay-off compared to the current practice.

6.1.4 Property Sets: Requirements in the Building Element Attributes

In the *IFC Specifications* versions 2.x and earlier, the IfcSpaceProgram object contained very few attributes (Figure 17). In the *IFC Specification* 2x2 the Pset_SpaceProgramCommon *Property Set* has been expanded significantly. The detailed analysis of these attributes is documented in Section 6.2.2. However, the IfcSpaceCommon *Property Set*, which is attached to the IfcSpace, also includes several *Requirements* (Section 6.2.2.3), and some of these are redundant with the Pset_SpaceProgramCommon attributes.

Attaching *Requirements* to the actual *Space* objects in the *Design Model* creates a fundamental problem related to the *Requirements Management*. The *IFC Specifications* allow shared *Property Sets*, e.g., one IfcSpaceCommon could be assigned to several IfcSpace objects. However, all known *IFC implementations* use instance-specific attribute sets, because the internal structure of design software does not support shared attributes. In practice this means that if *Requirements* are stored in the *Design Model*, the same *Requirements* are multiplied in all *Instances*. This multiplication can cause serious problems for *Requirements Management* when *Requirements* evolve and must be updated. This is one of the reasons why *Requirements* should not be stored in the *Design Model* (Section 5.1.1). Thus, the *Requirements* should be in separate objects which can be linked to each other in the *Requirements Model* and related objects in the *DPM Models*.

In addition, the *Property Sets* attached to the building objects in the *Design Model* will have to be either generic or based on a detailed *Specification*. In both cases the solution would share all potential problems of the selected approach discussed in Sections 6.1.3 and 6.1.5. Since it has all of the disadvantages of those methods, and no specific advantages, using *Property Sets* attached to building objects in the *Design Model* is the worst solution to the problem.

6.1.5 Detailed Requirements Model Specification

The third possible solution is a detailed *Requirements Model Specification*. The main benefit is a pre-defined structure including the links for *Direct and Indirect Requirements*. As briefly discussed in Section 4.3, the main problem is the difficulty of identifying a necessary set of *Requirements* which can satisfy the needs for different projects, but still be manageable for the users. The amount of possible *Requirements* is high, and, as documented in Section 4.2, only few of them are used on most projects. The set of commonly used *Requirements* is relatively small, but it does not necessarily mean that the seldom-used *Requirements* could be left out from the *Requirements Model Specification*.

The content of a *Requirements Model Specification* is an issue which can be discussed indefinitely. There is no “correct” answer because the needs in different projects inevitably differ. However, the only way to create a usable *Requirements Management* application is to use a detailed *Specification*; otherwise the definition of relations is too difficult and time-consuming for the end-user of the *Requirements Management* application, as discussed in Section 6.1.3. Thus, I base my solution on the analysis of two existing *Requirements* hierarchies (Section 3.2.2) and *Requirements* in various *Space Programs* (Chapter 4).

The content in my *Requirements Model Specification* (Section 6.3):

- Relates to the problems identified in this research (Section 7.1),
- Covers all *Requirements* identified in this research, e.g., it is general (Section 7.2),
- Is implementable (Section 7.3).

Thus, I believe that my *Requirements Model Specification*

- Is a valid scientific contribution (Section 8.1)
- Has practical implications (Section 8.2)
- Forms a basis for future development (Section 8.3).

6.1.6 Indirect and Direct Links

As documented in Section 6.1.2, many *Requirements* indirectly affect other building elements, than the ones with which they are directly associated. The *IFC Specifications* include mechanisms for the indirect links between objects in one *Model*, which are widely used in the *IFC Specifications*, and they have also the inverse option, which means that the relation can be recognized in both directions:

- To which building elements or systems a spatial element is linked, and
- Which is the spatial element to which a building element or system is linked.

These links are used, for example, in thermal simulation software products which have to recognize the *Bounding Elements* of each *Space* from the *Model*. An almost similar mechanism, IfcSystem, enables the aggregation of systems in the *IFC Specifications* (Section 6.2.5, Figure 41 and Figure 42). This means that the indirect links for *Requirements* are recognized in the *Design Model* based on the indirect link information in the *Requirements Object*. This recognition is a function of applications, not a property of the *Model Specification*, although this information can be written into the IFC file and used in the data exchange and sharing. The *Specification* only defines which objects should have the indirect link.

Because *Requirements Models* and *Design Models* are separate data sets (Section 5.1.1), the direct link between a *Requirements Object* and an object in the *Design Model* cannot use the same type of links which are used inside one *Model*. The link between the *Models* must be based on a different mechanism. As documented in Section 6.2.3.3, Globally Unique ID, GUID, is a widely used mechanism to identify objects in file exchange, but it has some serious problems in linkage, and in addition it does not contain the address and purpose of the linked *Model*, which are necessary information for the link. Section 6.3.2 documents my solution for the link. The descriptions and diagrams in Section 6.3 show both direct and indirect links for each *Requirements Object*.

6.1.7 Cascading Requirements: Space Instance and Type

As documented in the *Requirements* analysis, the *Space Requirements* are clearly the most often defined *Requirements* in the *Building Programs* (Chapter 4). As the rapid prototyping demonstrates, there is also a need to create a *Cascading Requirements* structure for *Spaces* (Chapter 5). This structure is based on two main *Requirements Objects: Space Program Instance* (*SPI*) and *Space Program Type* (*SPT*). In our "everyday language" the *SPI* is often called *Space Type*, and *SPT* is called category, "super-type" for *Space Type* (Figure 38). The reason for this naming is that categories and types have several different meanings and thus I wanted to use names which identify *SPI* and *SPT* exactly. *SPI* is not a type in the *Requirements Model;* it is a type only in relation with the *Spaces* in the *Design Model*.

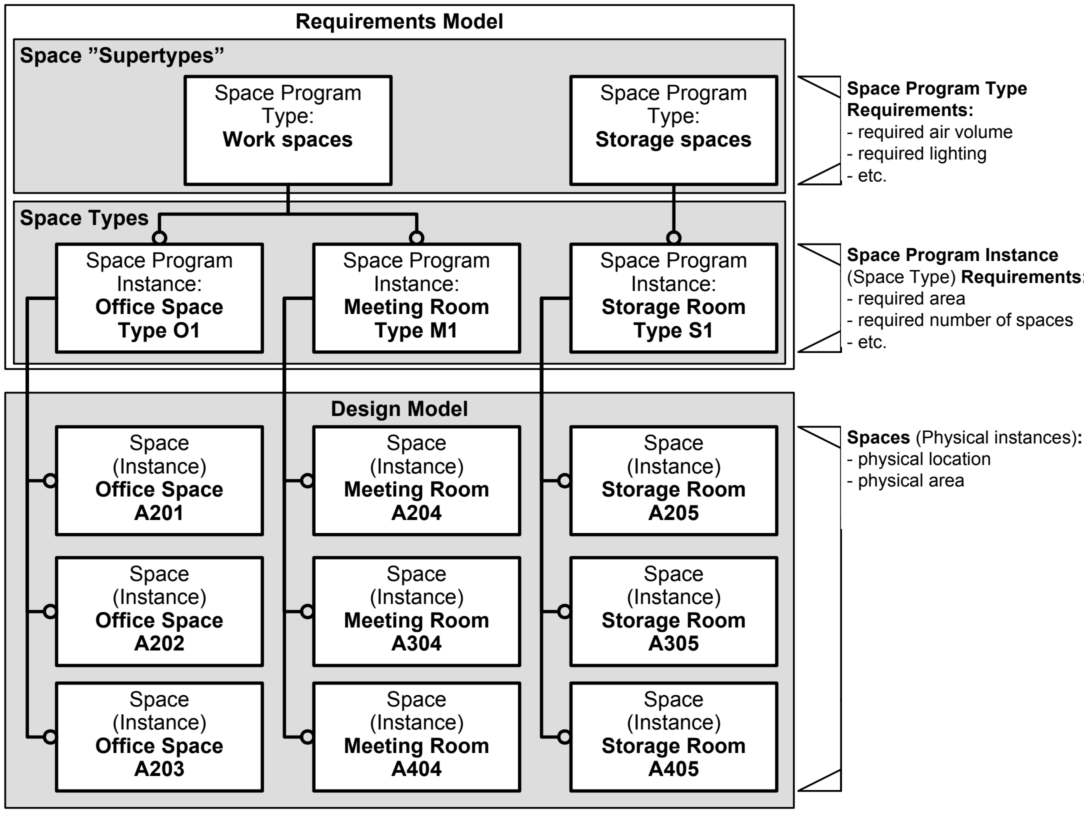

Figure 38: The hierarchy of *Space Program Types*, *Space Program Instances* and *Spaces*

A similar concept is often used in building projects, but because it is usually not formalized, this structure can be confusing. An example can illustrate the idea; we can think of categories as "super-types" of *Spaces (SPT)*, such as work spaces, storage spaces, and laboratories. These "super-types" define the standard *Requirements* for each *Space Type (SPI)*, for example, air volume, temperature, and lighting *Requirements*. A *Space Type (SPI)* has all the *Requirements* defined in its "super-type," *SPT*. These *Space Types (SPIs)* have additional *Requirements*, such as the number of *Spaces*, required area, adjacent *Spaces*, and department. These *Space Types*, *SPIs*, could be, for example, a 12 m^2 office room, an 8 m^2 storage room or a 100m^2 research laboratory. Each *Space Type* can be linked to several *Space Instances* in the *Design Model*.

This structure, *Cascading Requirements*, has significant benefits in *Requirements Capturing and Management;* standard *Requirements* are defined only for a few different "super-types," *SPT*, instead of defining them for every *Space Type*, *SPI*. The amount of work in defining and updating the *Requirements* is significantly smaller. However, this structure could be even deeper; there could be also a "super-type" for *SPTs*. This is one of the proposed future research issues (Section 8.3.2.4).

6.2 Existing Requirements Elements in the IFC Specifications

As my *Requirements Model Specification* is a potential extension for the *IFC Specifications*, it is important to analyze the existing elements of the *Specifications* to recognize what is missing relative to the identified problems. In this Section a large part of the text is directly from the *IFC Specifications*. The directly copied parts are indicated by the use of Times New Roman Font and there is always a reference to the source in the "IFC 2x2 Addendum 1" web pages.

As documented in Section 3.4 the *IFC Specifications* include only a few *Space* related *Requirements*, some generic *Requirements* objects and several *Property Sets* for *Requirements*. This section analyzes all these elements and in addition the other elements of the *IFC Specification* which are relevant for the *Requirements Model Specification*. These elements are:

- The generic constraint object: IfcConstraint (Section 6.2.1)
- *Space* related *Requirements:* IfcControl, IfcSpaceProgram and *Space* related *Requirements'* PropertySets (Section 6.2.2)
- Ownership and identification of the objects: IfcOwnerHistory and IfcGloballyUniqueID (Section 6.2.3)
- *Requirements* intent, design intent and approval status: IfcApprovalStatus (Section 6.2.4)
- References to external documents: IfcDocumentReference (Section 6.2.5)
- *Bounding Elements* and building systems: IfcRelSpaceBoundary, IfcSystem (Section 6.2.6)

6.2.1 Constraint Object in the IFC Specification

The current *IFC specifications* already include a *Generic Requirements Object*, IfcConstraint, which has two subtypes, IfcObjective and IfcMetric. IfcObjective captures qualitative information for an objective-based constraint, and IfcMetric captures quantitative resultant metrics that can be applied to objectives.

Definition and description from IAI [*IFC 2004a* [75]]:

"An IfcConstraint is used to define a constraint or limiting value or boundary condition that may be applied to an object or to the value of a property. IfcConstraint may be associated with any subtype of IfcObject through the IfcRelAssociatesConstraint relationship in the IfcControlExtension schema. A constraint may aggregate other constraints through the IfcConstraintAggregationRelationship through which a logical association between constraints may be applied. A constraint must have a name applied through the IfcConstraint.Name attribute and optionally, a description through IfcConstraint.Description."

EXPRESS specification:

```
ENTITY IfcConstraint
    ABSTRACT SUPERTYPE OF  (ONEOF(IfcObjective, IfcMetric));
        Name   :   IfcLabel;
        Description   :   OPTIONAL IfcText;
        ConstraintGrade   :   IfcConstraintEnum;
        ConstraintSource   :   OPTIONAL IfcLabel;
        CreatingActor   :   OPTIONAL IfcActorSelect;
        CreationTime   :   OPTIONAL IfcDateTimeSelect;
        UserDefinedGrade   :   OPTIONAL IfcLabel;
    INVERSE
        ClassifiedAs   :   SET OF IfcConstraintClassificationRelationship FOR ClassifiedConstraint;
        RelatesConstraints   :   SET OF IfcConstraintRelationship FOR RelatingConstraint;
        IsRelatedWith   :   SET OF IfcConstraintRelationship FOR RelatedConstraints;
        PropertiesForConstraint   :   SET OF IfcPropertyConstraintRelationship FOR
        RelatingConstraint;
        Aggregates   :   SET OF IfcConstraintAggregationRelationship FOR RelatingConstraint;
        IsAggregatedIn   :   SET OF IfcConstraintAggregationRelationship FOR RelatedConstraints;
    WHERE
```

```
        WR11   :   (ConstraintGrade <> IfcConstraintEnum.USERDEFINED) OR ((ConstraintGrade
        = IfcConstraintEnum.USERDEFINED) AND
        EXISTS(SELF\IfcConstraint.UserDefinedGrade));
END_ENTITY;
```

Based on my analysis (Chapter 4), the current IfcConstraint has some problems compared to the IfcControl object. The main issue is that the IfcConstraint is not a subtype of IfcObject, and thus it does not share the common linking resources of IfcObject (IfcRelAssociates: Section 6.3.2). Another issue is that IfcConstraint cannot include external references. However, drawings or other traditional documents are used as *Requirements* and they include important information for the design process. The most common data types for *Requirements* were textual descriptions (33 %) and numeric values (30 %), but also links to external documents were often used (20 %). Therefore the *Requirements Objects* in the *Requirements Model* should also support this data type.

The use of IfcConstraint as a *Generic Requirements Object* would include all the difficulties of the indirect linkage described in the Section 6.1.3. The new *Requirements Object* specified in Section 6.3.3 could of course be a subtype of IfcConstraint, but, in my opinion, IfcControl has more benefits and the current *IFC Specifications* already use it as the super-type of *Space Requirements* in (Section 6.2.2). Thus, I chose to use IfcControl as the basis for the new *Requirements Object* (Section 6.3.3).

6.2.2 Space-Related Requirements in the IFC Specifications

The other potential solution mentioned in Section 6.2.1 to the identified problems in the current *IFC specifications* is the IfcControl, and specifically one of its subtypes, IfcSpaceProgram. Compared to IfcConstraint, IfcControl provides a more flexible structure (Figure 39). The *Property Set* IfcSpaceProgramCommon was extended in IFC 2x2 Addendum 1 during my research, and this Section documents the *Space*-related *Requirement elements* in the *IFC Specification* 2x2 Addendum 1. Section 6.2.2.4 documents the conclusions of these existing elements.

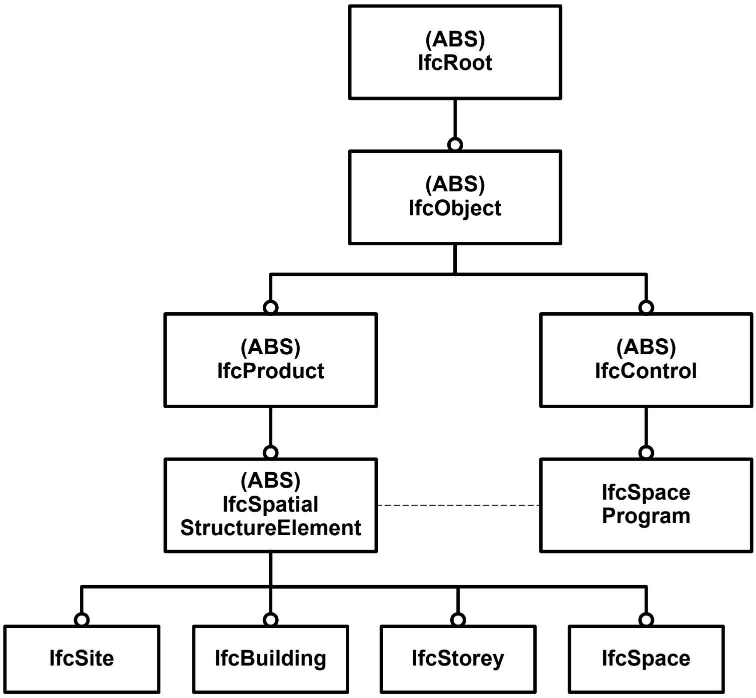

Figure 39: IfcSpaceProgram and its relation to spatial elements

6.2.2.1 Current IfcControl

Definition and description from IAI [*IFC 2004b* [76]]:

"The IfcControl is the abstract generalization of all concepts that control or constrain Products or Processes in general. It can be seen as a specification, regulation, constraint or other requirement applied to a product or process whose requirements and provisions must be fulfilled. Controls are assigned to products, processes, or other objects by using the IfcRelAssignsToControl relationship.

Examples for the use of IfcControls are space program, construction guides, etc. Some basic items, such as cost value, approval, or constraint are directly attachable to products and processes using the association relationship subtypes of IfcRelAssociates. IfcControl is defined in the IfcKernel but will be reused and specialized in other schemas."

EXPRESS specification:

```
ENTITY IfcControl;
        ENTITY IfcRoot;
                GlobalId  :  IfcGloballyUniqueId;
                OwnerHistory  :  IfcOwnerHistory;
```

```
            Name  :  OPTIONAL IfcLabel;
            Description  :  OPTIONAL IfcText;
        ENTITY IfcObject;
            ObjectType  :  OPTIONAL IfcLabel;
        INVERSE
            IsDefinedBy  :  SET OF IfcRelDefines FOR RelatedObjects;
            HasAssociations  :  SET OF IfcRelAssociates FOR RelatedObjects;
            HasAssignments  :  SET OF IfcRelAssigns FOR RelatedObjects;
            Decomposes  :  SET [0:1] OF IfcRelDecomposes FOR RelatedObjects;
            IsDecomposedBy  :  SET OF IfcRelDecomposes FOR RelatingObject;
        ENTITY IfcControl;
        INVERSE
            Controls  :  SET OF IfcRelAssignsToControl FOR RelatingControl;
END_ENTITY;
```

6.2.2.2 Current IfcSpaceProgram and Pset_SpaceProgramCommon

IfcSpaceProgram is a subtype of IfcControl, and it clearly addresses some of the problems identified in my research. During the research the IfcSpaceProgram has also developed compared to the point of departure documented in Section 3.4. The two latest versions, IFC 2x2 and 2x2 Addendum 1, include an attribute set, Pset_SpaceProgramCommon, which covers some of the information needs identified in my analysis (Table 6). These existing elements are used as a part of my *Space Requirements* objects (Section 6.3.10).

Definition and description from IAI [*IFC 2004c* [77]]

IfcSpaceProgram is "Architectural program for a space in the building or facility being designed; essentially the requirements definition for such a building space.

IfcSpaceProgram class is used to define:

- the architectural program for a space in the building or facility being designed;
- the standard for space allocation that can be assigned to persons within an organization.

As the architectural program, the IfcSpaceProgram class sets down the requirements definition for a space in the building or facility being designed. Used in this way, it defines the client requirements for the space before the building in designed. Space programs can change over the life cycle of a building, after the building is occupied.

Changes to space programs take place in the facilities management/operations phase of the building life cycle.

As a space standard for facilities management (FM), the IfcSpaceProgram class defines the requirements for usage of a space according to the roles of persons that will occupy the space. This could take into account role driven elements such as whether the space should be a single person office, corner space, glazing on two sides etc. In order to use the class as an FM space standard, a classification of spaces must have been established. This does not mean that each individual space needs to have a classification although for locating persons having an assigned space standard, this would be desirable."

EXPRESS specification:

```
ENTITY IfcSpaceProgram;
    SUBTYPE OF (IfcControl);
        SpaceProgramIdentifier : IfcIdentifier;
        MaxRequiredArea : OPTIONAL IfcAreaMeasure;
        MinRequiredArea : OPTIONAL IfcAreaMeasure;
        RequestedLocation : OPTIONAL IfcSpatialStructureElement;
        StandardRequiredArea : OPTIONAL IfcAreaMeasure;
    INVERSE
        HasInteractionReqsFrom : SET OF IfcRelInteractionRequirements FOR
        RelatedSpaceProgram;
        HasInteractionReqsTo : SET OF IfcRelInteractionRequirements FOR
        RelatingSpaceProgram;
END_ENTITY;
```

Name	Definition
SpaceProgramIdentifier	Identifier for this space program. It often refers to a number (or code) assigned to the space program. Example: R-001.
MaxRequiredArea	The maximum floor area programmed for this space (according to client requirements)
MinRequiredArea	The minimum floor area programmed for this space (according to client requirements)
RequestedLocation	Location within the building structure, requested for the space.
StandardRequiredArea	The floor area programmed for this space (according to client requirements)
HasInteractionReqsFrom	Set of inverse relationships to space or work interaction requirement objects (FOR RelatedObject)
HasInteractionReqsTo	Set of inverse relationships to space or work interaction requirements (FOR RelatingObject)

Table 6: Pset_SpaceProgramCommon attributes

Name	Property Type	Data Type	Definition
Location	IfcPropertySingleValue	IfcLabel	General description of the required location for the space (e.g. "third floor south").
Function Requirement	IfcPropertySingleValue	IfcLabel	General description of the functional requirement for the space (in addition to the space name).
Security Requirement	IfcPropertySingleValue	IfcLabel	General description of the security requirement for the space (in addition to the function requirement).
Privacy Requirement	IfcPropertySingleValue	IfcLabel	General description of the privacy requirement for the space (in addition to the security requirement).
Lighting Requirement	IfcPropertySingleValue	IfcLabel	General description of the lighting requirement for the space (e.g. "natural lighting required").
FFEType Requirement	IfcPropertySingleValue	IfcLabel	General description of the Furniture, Fixtures and Equipment requirement for this space.
Employee Type	IfcPropertySingleValue	IfcLabel	General description of the employee type that will occupy the space (e.g. manager, programmer, secretary, etc.). The type classification depends on the company based terms for employee types.
Occupancy Type	IfcPropertySingleValue	IfcLabel	Occupancy type for this object. It is defined according to the presiding national building code.
Occupancy Number	IfcPropertySingleValue	IfcCount Measure	Maximum number of occupants for the designed usage of the space.

6.2.2.3 Requirements Property Sets for IfcSpace

Another entity containing *Space*-related *Requirements* in the *IFC Specifications* is IfcSpace. IfcSpace object has several *Property Sets*, but because my research concentrates on *Requirements*, I document here only the *Property Sets* related to the *Requirements*.

Definition and description from IAI [*IFC 2004d* [78]]:

Property Set Use Definition:

The property sets relating to the IfcSpace are defined by the IfcPropertySet and attached by the IfcRelDefinesByProperties relationship. It is accessible by the inverse IsDefinedBy relationship. The following property set definitions specific to the IfcSpace are part of this IFC release:

– Pset_SpaceCommon: common property set for all types of spaces

– Pset_SpaceParking: specific property set for only those spaces that are used to define parking spaces by ObjectType = 'Parking'

– Pset_SpaceParkingAisle: specific property set for only those spaces that are used to define parking aisle by ObjectType = 'ParkingAisle'

– Pset_SpaceFireSafetyRequirements: common property set for all types of spaces to capture the fire safety requirements

– Pset_SpaceLightingRequirements: common property set for all types of spaces to capture the lighting requirements

– Pset_SpaceOccupancyRequirements: common property set for all types of spaces to capture the occupancy requirements

– Pset_SpaceThermalRequirements: common property set for all types of spaces to capture the thermal requirements"

Because this solution based on several *Property Sets* attached to the *Spaces* causes a fundamental problem for *Requirements Management* (Section 6.1.4), I do not document the definitions here in detail. The content of these *Property Sets* is in Table 7 and my comments are recorded in the following Section 6.2.2.4.

6.2.2.4 Observations and Conclusions of the IfcControl, IfcSpaceProgram, Pset_SpaceProgramCommon and Requirements Property Sets for IfcSpace

There are several peculiarities, even mistakes, in the *Requirements* for *Spaces* in the current *IFC Specifications* as shown in Table 7. Some *Requirements* are in the IfcSpaceProgram, some in its Pset_SpaceProgramCommon, and in addition there are 7 *Property Sets* defining *Requirements* in the IfcSpace entity. It is obvious that the development of the *Requirements* in the *IFC Specifications* has been based on several ad-hoc additions in different places without any systematic plan for *Requirements Management*.

The main issues are that *Requirements* should not be in the *Space* objects in the *Design Model*, and that they should not be in attribute sets. These arguments are based on the conclusions in Sections 5.1.1 and 6.1.4. *Design Objects* do not exist when the *Requirements Capturing* process starts, and an efficient *Requirements Management* process requires that *Requirements* are not multiplied in separate attribute sets in every *Instance* in the *Design Model*.

Regardless of these principles, the *Space*-related *Requirements* in the current *IFC Specifications* are not logical, see Table 7. I have added the first two columns (number and purpose) to help identify different *Requirements* (for example, #1 is HandicapAccessible in Pset_SpaceCommon) and sort them into an order based on their use, the other information is directly from the *IFC Specification*.

Table 7: *Space*-related *Requirements* in the *IFC Specification 2x2 Addendum 1*

#	Use	IfcEntity	Name	Data Type	Definition
1	Accessibility	Pset_Space Common	Handicap Accessible	IfcBoolean	Indication whether this space (in case of e.g., a toilet) is designed to serve as an accessible space for handicapped people, e.g., for a public toilet (TRUE) or not (FALSE). This information is often used to declare the need for access for the disabled and for special design requirements of this space.
2	Accessibility	Pset_Space Parking	Handicap Accessible	IfcBoolean	Indication that this object is designed to be accessible by the handicapped. It is giving according to the requirements of the national building code.
3	Adjacency	IfcSpace Program	HasInteraction ReqsFrom	SET OF IfcRel Interaction Requirements	Set of inverse relationships to space or work interaction requirement objects (FOR RelatedObject).
4	Adjacency	IfcSpace Program	HasInteraction ReqsTo	SET OF IfcRel Interaction Requirements	Set of inverse relationships to space or work interaction requirements (FOR RelatingObject).
5	Aesthetics	Pset_Space Occupancy Requirements	IsOutlook Desirable	IfcBoolean	An indication of whether the outlook is desirable (TRUE) or not (FALSE)
6	Area	Pset_Space Common	GrossArea Planned	IfcArea Measure	Total planned area for the space. Used for programming the space.
7	Area	IfcSpace Program	MaxRequired Area	IfcArea Measure	The maximum floor area programmed for this space (according to client requirements).
8	Area	IfcSpace Program	MinRequired Area	IfcArea Measure	The minimum floor area programmed for this space (according to client requirements).
9	Area	IfcSpace Program	Standard RequiredArea	IfcArea Measure	The floor area programmed for this space (according to client requirements).
10	Fire safety	Pset_Space FireSafety Requirements	Ancillary FireUse	IfcLabel	Ancillary fire use for the space which is assigned from the fire use classification table as given by the relevant national building code.

#	Use	IfcEntity	Name	Data Type	Definition
11	Fire safety	Pset_Space FireSafety Requirements	FireExit	IfcBoolean	Indication whether this object is designed to serve as an exit in the case of fire (TRUE) or not (FALSE). Here whether the space (in case of e.g., a corridor) is designed to serve as an exit space, e.g., for fire escape purposes.
12	Fire safety	Pset_Space FireSafety Requirements	FireHazard Factor	IfcLabel	Fire hazard code of the space. The coding depends on the national fire safety regulations.
13	Fire safety	Pset_Space FireSafety Requirements	FireRisk Factor	IfcLabel	Fire Risk factor assigned to the space according to local building regulations.
14	Fire safety	Pset_Space FireSafety Requirements	Flammable Storage	IfcBoolean	Indication whether the space is in-tended to serve as storage of flamma-ble material (which is regarded as such by the presiding building code. (TRUE) indicates yes, (FALSE) otherwise.
15	Fire safety	Pset_Space FireSafety Requirements	MainFireUse	IfcLabel	Main fire use for the space which is assigned from the fire use classification table as given by the relevant national building code.
16	Fire safety	Pset_Space FireSafety Requirements	Sprinkler Protection	IfcBoolean	Indication whether the space is sprinkler protected (true) or not (false).
17	Fire safety	Pset_Space FireSafety Requirements	Sprinkler Protection Automatic	IfcBoolean	Indication whether the space has an automatic sprinkler protection (true) or not (false). It should only be given, if the property "SprinklerProtection" is set to TRUE.
18	Function	Pset_Space Program Common	Function Requirement	IfcLabel	General description of the functional requirement for the space (in addition to the space name)
19	Furniture	Pset_Space Program Common	FFEType Requirement	IfcLabel	General description of the Furniture, Fixtures and Equipment requirement for this space.
20	Height	Pset_Space Occupancy Requirements	Minimum Headroom	IfcLength Measure	Headroom required for the activity assigned to this space.
21	HVAC	Pset_Space Thermal Requirements	Air Conditioning	IfcBoolean	Indication whether this space requires air conditioning provided (TRUE) or not (FALSE).
22	HVAC	Pset_Space Thermal Requirements	Air Conditioning Central	IfcBoolean	Indication whether the space requires a central air conditioning provided (TRUE) or not (FALSE). It should only be given, if the property "AirConditioning" is set to TRUE.
23	HVAC	Pset_Space FireSafety Requirements	Air Pressurization	IfcBoolean	Indication whether the space is required to have pressurized air (TRUE) or not (FALSE).
24	HVAC	Pset_Space Thermal Requirements	Discontinued Heating	IfcBoolean	Indication whether discontinued heating is required/desirable from user/designer view point. (True) if yes, (FALSE) otherwise.

#	Use	IfcEntity	Name	Data Type	Definition
25	HVAC	Pset_Space Common	Mechanical Ventilation Rate	IfcCount Measure	Indication of the requirement of a particular mechanical air ventilation rate, given in air changes per hour.
26	HVAC	Pset_Space Common	Natural Ventilation	IfcBoolean	Indication whether the space is required to have natural ventilation (true) or mechanical ventilation (false).
27	HVAC	Pset_Space Common	Natural Ventilation Rate	IfcCount Measure	Indication of the requirement of a particular natural air ventilation rate, given in air changes per hour.
28	HVAC	Pset_Space Thermal Requirements	Space Humidity	IfcRatio Measure	Humidity of the space or zone that is required from user/designer view point. If no summer or winter space humidity requirements are given, it applies all year, otherwise for the intermediate period.
29	HVAC	Pset_Space Thermal Requirements	Space Humidity Summer	IfcRatio Measure	Humidity of the space or zone for the hot (summer) period, that is required from user/designer view point.
30	HVAC	Pset_Space Thermal Requirements	Space Humidity Winter	IfcRatio Measure	Humidity of the space or zone for the cold (winter) period that is required from user/designer view point.
31	HVAC	Pset_Space Thermal Requirements	Space Temperature Summer	IfcThermo dynamic Temperature Measure	Temperature of the space or zone for the hot (summer) period, that is required from user/designer view point.
32	HVAC	Pset_Space Thermal Requirements	Space Temperature Winter	IfcThermo dynamic Temperature Measure	Temperature of the space or zone for the cold (winter) period, that is required from user/designer view point.
33	HVAC	Pset_Space Thermal Requirements	Space Temperature Max	IfcThermo dynamic Temperature Measure	Temperature of the space or zone, that is required from user/designer view point. If no summer or winter space temperature requirements are given, it applies all year, otherwise for the intermediate period.
34	HVAC	Pset_Space Thermal Requirements	Space Temperature Min	IfcThermo dynamic Temperature Measure	Minimal Temperature of the space or zone, that is required from user/designer view point. It applies all year.
35	Lighting	Pset_Space Lighting Requirements	Artificial Lighting	IfcBoolean	Indication whether this space requires artificial lighting (as natural lighting would be not sufficient). (TRUE) indicates yes (FALSE) otherwise.
36	Lighting	Pset_Space Lighting Requirements	Illuminance	IfcIlluminance Measure	Required average illuminance value for this space.
37	Lighting	Pset_Space Program Common	Lighting Requirement	IfcLabel	General description of the lighting requirement for the space (e.g. "natural lighting required")
38	Location	Pset_Space Program Common	Location	IfcLabel	General description of the required location for the space (e.g. "third floor south")

#	Use	IfcEntity	Name	Data Type	Definition
39	Location	IfcSpace Program	Requested Location	IfcSpatial Structure Element	Location within the building structure, requested for the space.
40	Occupancy	Pset_Space Occupancy Requirements	AreaPer Occupant	IfcArea Measure	Design occupancy loading for this type of usage assigned to this space.
41	Occupancy	Pset_Space Program Common	Employee Type	IfcLabel	General description of the employee type that will occupy the space (e.g. manager, programmer, secretary, etc.).
42	Occupancy	Pset_Space Common	Occupancy Number	IfcCount Measure	Maximum number of occupants for the designed usage of the space.
43	Occupancy	Pset_Space Occupancy Requirements	Occupancy Number	IfcCount Measure	Number of people required for the activity assigned to this space.
44	Occupancy	Pset_Space Program Common	Occupancy Number	IfcCount Measure	Maximum number of occupants for the designed usage of the space.
45	Occupancy	Pset_Space Occupancy Requirements	Occupancy NumberPeak	IfcCount Measure	Maximal number of people required for the activity assigned to this space in peak time.
46	Occupancy	Pset_Space Occupancy Requirements	Occupancy TimePerDay	IfcTime Measure	The amount of time during the day that the activity is required within this space.
47	Occupancy	Pset_Space Common	Occupancy Type	IfcLabel	Occupancy type for this object. It is defined according to the presiding national building code.
48	Occupancy	Pset_Space Occupancy Requirements	Occupancy Type	IfcLabel	Occupancy type for this object. It is defined according to the presiding national building code.
49	Occupancy	Pset_Space Program Common	Occupancy Type	IfcLabel	Occupancy type for this object. It is defined according to the presiding national building code.
50	Privacy	Pset_Space Program Common	Privacy Requirement	IfcLabel	General description of the privacy requirement for the space (in addition to the security requirement)
51	Privacy	Pset_Space Common	Publicly Accessible	IfcBoolean	Indication whether this space (in case of e.g., a toilet) is designed to serve as a publicly accessible space, e.g., for a public toilet (TRUE) or not (FALSE).
52	Reference	Pset_Space Common	Reference	IfcIdentifier	Reference ID for this specified type in this project (e.g. type 'A-1')
53	Security	Pset_Space Program Common	Security Requirement	IfcLabel	General description of the security requirement for the space (in addition to the function requirement)
54	Technical	Pset_Space Common	Concealed	IfcBoolean	Indication whether this space is declared to be a concealed space (TRUE) or not (FALSE). A concealed space is normally meant to be the space between a slab and a ceiling, or beneath a raised floor.
55	Traffic	Pset_Space ParkingAisle	IsOneWay	IfcBoolean	Indicates whether the parking aisle is designed for one-way traffic (TRUE) or two-way traffic (FALSE).

Detailed observations of the structure and content of *Space*-related *Requirements* in the current *IFC Specifications:*

- The first observation is the difficulty to find the *Space*-related *Requirements* in the *IFC Specifications*, because they are scattered in many places in the *Specification*. This has obviously caused difficulties even to the people making the *IFC Specification*, because there are several overlapping definitions; especially in the occupancy *Requirements*. The decisions of which *Requirements* are in the IfcSpaceProgram, Pset_SpaceProgramCommon, Pset_SpaceCommon and Pset_SpaceOccupancyRequirements seems haphazard; there is no logic in their content. This is confusing, and leads easily to multiplication, which is already evident in the *IFC Specifications*. In addition, this can lead to the situation where different software products use different attribute for the same information, so the IAI's main goal, interoperability, is missed.
- Pset_SpaceThermalRequirements, Pset_SpaceLightingRequirements and Pset_SpaceFireSafetyRequirements are logical. The only issue is that they should not be in the IfcSpace object, but in the IfcSpaceProgram, or some other *Requirements Object*.
- "HandicapAccessible" is in two IfcSpace *Property Sets*; Pset_SpaceCommon (#1) and Pset_SpaceParking (#2). There is no logical reason why both exist; there is no conceptual difference between the accessibility *Requirement* for a *Room* or for a parking space for handicapped people. In addition, "HandicapAccessible" is the only attribute in the Pset_SpaceParking (#2), which makes the whole *Property Set* obsolete if this redundant attribute is removed from it.
- The IfcSpaceProgram entity includes three different areas, MaxRequiredArea (#7), MinRequiredArea (#8) and StandardRequiredArea (#9). This is logical, because different organizations can define the area *Requirements* using different methods. However, having a "GrossAreaPlanned" (#6) attribute in the Pset_SpaceCommon is redundant with StandardRequiredArea (#9).

- Location (#38) is in IfcSpaceProgram. Thus, the “RequestedLocation” (#39) entity in the Pset_SpaceProgramCommon appears redundant although the mechanisms to specify the requested *Location* in these two *Requirements* are totally different. In any case, two different places for *Location Requirements* can cause confusion in the use of the *Specifications*. The use of a simple description to define the required *Location* seems more practical. In addition, I propose in my *Requirements Model Specification* a list of *adjacent Spaces* for additional *Location Requirements*. Thus, I propose that the “RequestedLocation” should be removed from the IfcSpaceProgram.
- Both “OccupancyNumber” and “OccupancyType” have three locations in the *IFC Specification;* they are in Pset_SpaceCommon (#42 and #47), Pset_SpaceOccupancyRequirements (#43 and #48), and Pset_SpaceProgramCommon (#44 and #49). There is no reason for this. The proposed use for these three attributes is the same, although the Pset_SpaceOccupancyRequirements have a slightly different description than the two others which have exactly the same description.
- “PrivacyRequirement” is defined in Pset_Space ProgramCommon and “PubliclyAccessible” in Pset_SpaceCommon. They are not overlapping, but nevertheless having *Requirements* in the same category in two different objects and two different *Property Sets* is not logical.

As a short-term correction, I propose that in the next version of *IFC Specifications*

- All overlapping definitions be removed, and
- All *Space*-related *Requirements* in the *IFC Specification* be placed into the IfcSpaceProgram entity, grouped into four categories: Common, Thermal, Lighting, and Fire Safety *Requirements*.

In the long term, I believe that the correct solution is to have a systematic set of *Requirements Objects* in the *IFC Specifications*. However, despite the logical errors in the *Space*-related *Requirements* in the latest *IFC Specification*, the IfcControl entity and its special case, IfcSpaceProgram, provide the basic methods for my *Requirements Model Specification* (Section 6.3) which I propose as the basis for the future IFC work.

6.2.3 Requirements Ownership and Requirements History

Two important elements in the *Requirements Management* process are the ownership and change history of *Requirements*. As described in Section 1.1, *Requirements* evolve during the design and construction process, and it is crucial to know the source of *Requirements* as well as being able to trace their evolution.

All subtypes of IfcRoot in the *IFC Specifications* have two elements which enable these two important features in *Requirements Management*. They are IfcOwnerHistory (Section 6.2.3.1) and IfcGloballyUniqueID (Section 6.2.3.2). This means that each IFC entity has a specified owner and can be identified by its unique ID. For *Requirements Management* this means that the evolution of the *Requirements* can be stored in a "history part" of the *Requirements Model* by storing all previous versions of the *Requirements Objects* using the unique ID as the identifier of the different versions of the same *Requirements Object*.

6.2.3.1 IfcOwnerHistory

Definition and description from IAI [*IFC 2004e* [79]]:

"IfcOwnerHistory defines all history and identification related information. In order to provide fast access it is directly attached to all independent objects, relationships and properties.

IfcOwnerHistory is used to identify the creating and owning application and user for the associated object, as well as capture the last modifying application and user."

EXPRESS specification:

```
ENTITY IfcOwnerHistory;
    OwningUser : IfcPersonAndOrganization;
    OwningApplication : IfcApplication;
    State : OPTIONAL IfcStateEnum;
    ChangeAction : IfcChangeActionEnum;
    LastModifiedDate : OPTIONAL IfcTimeStamp;
    LastModifyingUser : OPTIONAL IfcPersonAndOrganization;
    LastModifyingApplication : OPTIONAL IfcApplication;
    CreationDate : IfcTimeStamp;
END_ENTITY;
```

Name	Description
OwningUser	*Direct reference to the end user who currently "owns" this object. Note that IFC includes the concept of ownership transfer from one user to another and therefore distinguishes between the Owning User and Creating User.*
OwningApplication	*Direct reference to the application which currently "Owns" this object on behalf of the owning user who uses this application. Note that IFC includes the concept of ownership transfer from one app to another and therefore distinguishes between the Owning Application and Creating Application.*
State	*Enumeration that defines the current access state of the object.*
ChangeAction	*Enumeration that defines the actions associated with changes made to the object.*
LastModifiedDate	*Date and Time at which the last modification occurred.*
LastModifyingUser	*User who carried out the last modification.*
LastModifyingApplication	*Application used to carry out the last modification.*
CreationDate	*Time and date of creation.*

6.2.3.2 IfcGloballyUniqueID

Definition and description from IAI [*IFC 2004f* [80]]

IfcGloballyUniqueID "Holds an identifier that is unique throughout the software world. This is also known as a Globally Unique Identifier (GUID) or Universal Unique Identifier (UUID) by the Open Group. The identifier is generated using an algorithm published by the Object Management Group. The algorithm is explained at the open group website." [GUI*D 2005* [81]]

EXPRESS specification:

```
TYPE IfcGloballyUniqueId = STRING (22) FIXED;
END_TYPE;
```

6.2.3.3 Limitations of Current GUID and IfcOwnerHistory Elements

However, there are also limitations caused by the structure of *IFC Specifications*. The first limitation is the granularity of information. Each object can have only one owner, one modifier, and one GUID. However, each *Requirements Object* includes several individual *Requirements* and in some cases the owner and/or modifier of these individual *Requirements* can be different. In addition, if only one *Requirement* in the *Requirements Object* is changed, the old version of the whole *Requirements Object* must be stored into the "history database."

Another, more severe problem relates to the use of the GUID. If the users make modifications by deleting objects and replacing them with new objects, all links based on GUIDs will break. In addition, some software products change the GUIDs when the project's data set is exchanged in IFC format even if the objects are not changed in the original *Model*.

For example, Table 8 documents the BLIS/IAI certification workshop results. The test was done by exporting a simple test *Model* in IFC format and then importing the exported IFC file to the same software. In this test all GUIDs should be identical. However, the results were that only 6 object types maintained their GUIDs in all three software products, and only one of the software products, NEC, maintained the GUIDs for all object types [*BLIS 2002* [82]].

Table 8: Official GUID tracking results in the BLIS/IAI certification workshop, Tokyo 2002
1 = no changes in GUIDs, 0 = GUIDs have changed, NA = software does not use the object type

Element	NEC	Fujitsu	Sumitomo	Average
IfcColumn	1	1	1	100%
IfcOpeningElement	1	1	1	100%
IfcSlab	1	1	1	100%
IfcSpace	1	1	1	100%
IfcWall	1	1	1	100%
IfcBeam	1	NA	NA	100%
IfcDoor	1	1	0	67%
IfcGridAxis	1	0	1	67%
IfcWindow	1	1	0	67%
IfcBuilding	1	0	0	33%
IfcBuildingStorey	1	0	0	33%
IfcConstrainedPlacement	1	0	0	33%
IfcConstraintRelIntersection	1	0	0	33%
IfcDesignGrid	1	0	0	33%
IfcExtensionPropertySet	1	0	0	33%
IfcGridIntersection	1	0	0	33%
IfcGridLevel	1	0	0	33%
IfcLocalPlacement	1	0	0	33%
IfcProject	1	0	0	33%
IfcPropertySet	1	0	0	33%
IfcRelAssignsProperties	1	0	0	33%
IfcRelContains	1	0	0	33%
IfcRelFillsElement	1	0	0	33%
IfcRelSeparatesSpaces	1	0	0	33%
IfcRelVoidsElement	1	0	0	33%
IfcSite	1	0	0	33%
IfcSpaceBoundary	1	0	0	33%
In total	**100%**	**27%**	**23%**	**52%**

Another example of GUID problems is the GUID report table from the Aurora 2 project [Table 9, *Senate 2004* [83]]. There are only 3 object types in which GUIDs have not changed and 21 object types in which all GUIDs have changed. In addition, there are 6 object types, in which some GUIDs have changed. At least some of these changes are results of design changes, but if the number of deleted and new GUIDs is the same, it is most likely because the architect has deleted an existing object and replaced it with another object instead of editing the existing object.

Table 9: GUID report from Aurora 2 project, Senate Properties 2004

GUID Report								
Created : 22.12.2004 11:39:16	**Status**					**Entity Count**		
Entity Type		**Match**	**New**	**Deleted**	**Changed**	**New**	**Old**	**Difference**
IfcBuilding	All Changed	0	1	1	0	1	1	0
IfcBuildingElementProxy	All Changed	0	900	900	32	932	932	0
IfcBuildingStorey	All Changed	0	6	6	0	6	6	0
IfcColumn	Some Changes	189	10	10	0	199	199	0
IfcDoor	Some Changes	440	7	7	0	447	447	0
IfcDoorLiningProperties	All Changed	0	447	447	0	447	447	0
IfcDoorPanelProperties	All Changed	0	447	447	0	447	447	0
IfcDoorStyle	All Changed	0	447	447	0	447	447	0
IfcElectricalElement	Not used	0	0	0	0	0	0	0
IfcElementQuantity	All New	0	4315	0	0	4315	0	4315
IfcFurnishingElement	**No Change**	**89**	**0**	**0**	**0**	**89**	**89**	**0**
IfcOpeningElement	Some Changes	44	1503	1503	0	1547	1547	0
IfcProject	All Changed	0	1	1	0	1	1	0
IfcPropertySet	All Changed	0	11249	6692	0	11249	6692	4557
IfcRelAggregates	All Changed	0	6	6	0	6	6	0
IfcRelAssociatesClassification	All Changed	0	11	11	0	11	11	0
IfcRelAssociatesMaterial	All Changed	0	2634	2602	0	2634	2602	32
IfcRelConnectsPathElements	All Changed	0	1093	1093	0	1093	1093	0
IfcRelContainedInSpatialStructure	All Changed	0	41	5	0	41	5	36
IfcRelDefinesByProperties	All Changed	0	15564	6692	0	15564	6692	8872
IfcRelDefinesByType	All Changed	0	1420	1420	0	1420	1420	0
IfcRelFillsElement	All Changed	0	1420	1420	0	1420	1420	0
IfcRelVoidsElement	All Changed	0	1547	1547	0	1547	1547	0
IfcSite	All Changed	0	1	1	0	1	1	0
IfcSlab	Some Changes	550	1	1	0	551	551	0
IfcSpace	**No Change**	**400**	**0**	**0**	**0**	**400**	**400**	**0**
IfcStair	**No Change**	**6**	**0**	**0**	**0**	**6**	**6**	**0**
IfcWallStandardCase	Some Changes	938	22	22	0	960	960	0
IfcWindow	Some Changes	972	1	1	0	973	973	0
IfcWindowLiningProperties	All Changed	0	973	973	0	973	973	0
IfcWindowPanelProperties	All Changed	0	973	973	0	973	973	0
IfcWindowStyle	All Changed	0	973	973	0	973	973	0
This report was generated by Information Model Reporter (IMR).								
Copyright © 2003-2004 qPartners Oy. All Rights Reserved.								

GUID would be a perfect method to link objects (1) if all software products would maintain them in the data exchange, and (2) if designers would never delete and add objects in the *Design Model* if they could make the changes by editing existing objects. Unfortunately neither is a realistic demand. This makes the GUID-based identification a vulnerable method to link objects between different *Models*.

In theory, end-user behavior can be influenced by education, but in practice limitations in the editing process will not work; whatever is the easiest way to make changes will be used. If the linking method is based on user-defined, understandable mechanism, such as a type code in *Spaces*, instead of a highly abstract GUID generated by the software, it is easier for the end-users to understand and remember the importance of correct editing methods when working with the *Models*. In addition, (1) the users have some way to correct the links by correcting the type codes, and (2) the software products cannot change the information in the data exchange. However, the IDs managed by the users of the software are also problematic. People easily make mistakes even if the software provides help in controlling the IDs. In addition, user-defined IDs are not usable for most building elements, such as walls and columns.

All object-based software products have internal IDs for the objects and the integrity of these IDs is well maintained. However, these IDs are usually unique only in each file. This means that different files, e.g. *Models,* will contain the same IDs and thus the links between *Models* cannot be based on the internal IDs.

These problems must be addressed when linking objects between *Models.* It is not possible to rebuild these links several times during the design process, because a *Model* can include thousands of linked objects. If the links break easily, the use of linked *Models* will be impossible. One solution is to build a mechanism based on the use of these three different IDs: GUID, user-defined IDs and the software's internal IDs. For example, a *Model Server* software could use the GUIDs as its internal IDs, combine the GUIDs with the internal and/or user-defined IDs in each *Sub-Model* and use this combination in the information exchange between the *Models* (Figure 40). This mechanism would solve most of the integrity problems, and help identify possibly broken links if the users delete linked objects in the *Sub-Models*.

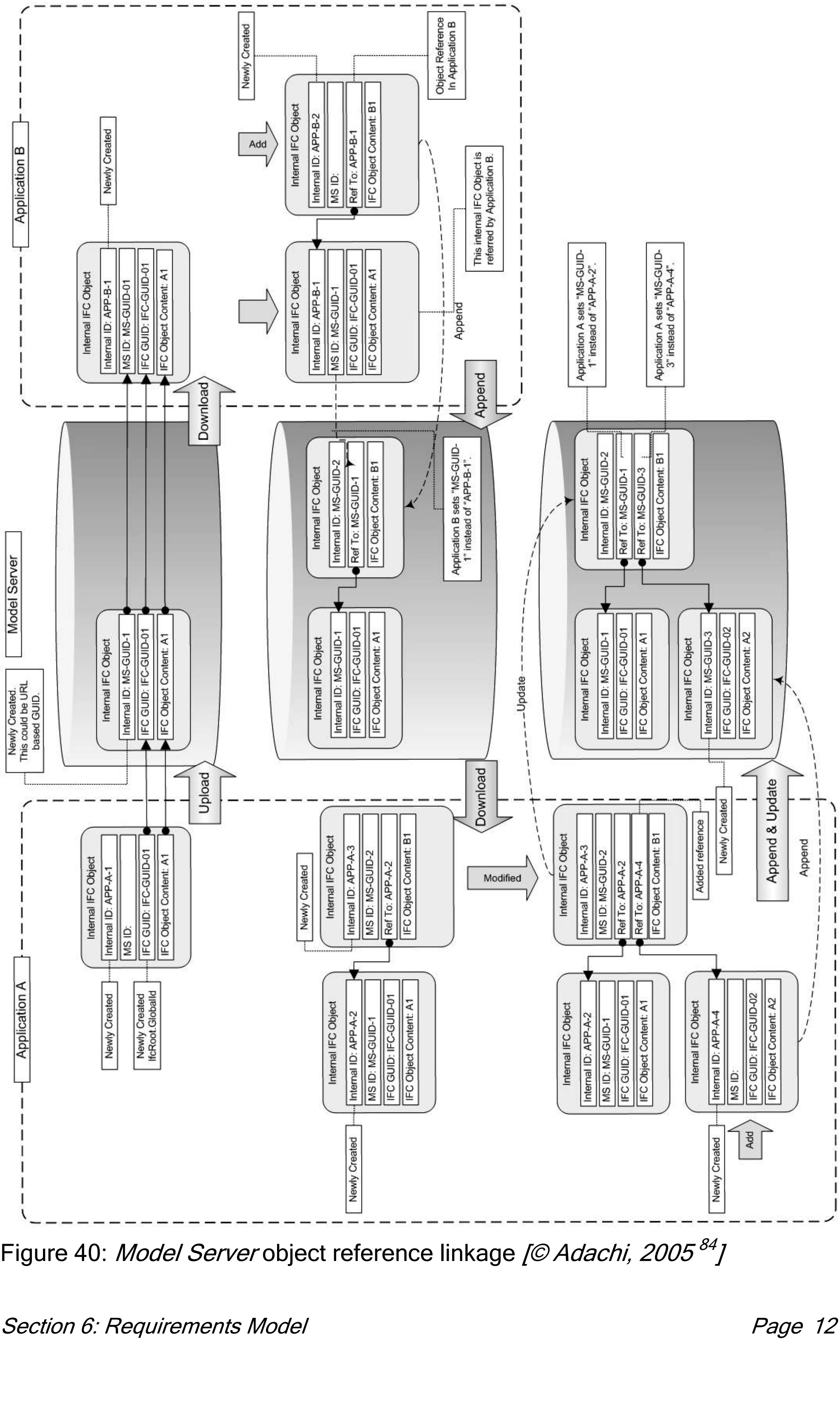

Figure 40: *Model Server* object reference linkage *[© Adachi, 2005[84]]*

6.2.3.4 Conclusions of Requirements Ownership and History

The granularity problem described in Section 6.2.3.3 can be solved in two ways, either by (1) forming a separate *Requirements Object* for each individual *Requirement*, or (2) creating a *Requirement* element which can store the necessary information separately for each *Requirement.* The first solution would create a large number of object definitions in the *Requirements Model Specification*, and it would be difficult to maintain. Thus, I propose a new Requirement Element described in Section 6.3.4. This new element enables identification of each individual *Requirement* and documentation of its owner, source and date.

My conclusion from the GUID problems is that the link between the *Requirements and DPM Models* should not be based solely on the GUIDs, but the combination of IDs described in Section 6.2.3.3. My solution for the link is documented in detail in Section 6.3.2. I also propose the GUID problems as topics for further research (Section 8.3.2.6).

6.2.4 Requirements Intent, Design Intent and Approval Status

Many *Requirements* are descriptions rather than exact values (Chapter 4). This “fuzzy,” only human-interpretable, content of *Requirements* often creates a need to document reasons for *Requirements* and/or design solutions. All *Requirements Objects* in my *Requirements Model Specification* contain a place to document both *Requirements* and design intent. Both elements use the structure of RequirementElement (Section 6.3.4).

In addition, it is often important to document the approval status; is a *Requirement* met fully, in part, or is it rejected. The current *IFC Specifications* contain an object for approval, IfcApprovalStatus, and it is used in all *Requirements Objects* in my *Requirements Model Specification*. I have placed the approval status on the *Requirements Object* level, but it could also be a part of the Requirement-Element (Section 6.3.4). This is one of the proposed topics for future research (Section 8.3.1.8).

6.2.4.1 IfcApprovalStatus

Definition and description from IAI [*IFC 2004h* [85]]

"An IfcApproval represents information about approval processes for a plan, a design, a proposal, a change order, etc., in a construction or facilities management project. IfcApproval is referenced by IfcRelAssociatesApproval in IfcControlExtension schema, and thereby can be related to all subtypes of IfcRoot."

EXPRESS specification:

```
ENTITY IfcApproval;
        Description : OPTIONAL IfcText;
        ApprovalDateTime : IfcDateTimeSelect;
        ApprovalStatus : OPTIONAL IfcLabel;
        ApprovalLevel : OPTIONAL IfcLabel;
        ApprovalQualifier : OPTIONAL IfcText;
        Name : IfcLabel;
        Identifier : IfcIdentifier;
    INVERSE
        Actors : SET OF IfcApprovalActorRelationship FOR Approval;
        IsRelatedWith : SET OF IfcApprovalRelationship FOR RelatedApproval;
        Relates : SET OF IfcApprovalRelationship FOR RelatingApproval;
END_ENTITY;
```

Name	Description
Description	*A general textural description of the Requirements and/or design solutions that is being approved for.*
ApprovalDateTime	*Date and time when the result of the approval process is produced.*
ApprovalStatus	*The result or current status of the approval, e.g., Requested, Processed, Approved, Not Approved, Rejected.*
ApprovalLevel	*Level of the approval e.g. Draft vs. Completed design.*
ApprovalQualifier	*Textual description of special constraints or conditions for the approval.*
Name	*A human-readable name given to an approval.*
Identifier	*A computer interpretable identifier by which the approval is known.*
Actors	*The set of relationships by which the actors acting in specified roles on this approval are known.*
IsRelatedWith	*The set of relationships by which this approval is related to others.*
Relates	*The set of relationships by which other approvals are related to this one.*

6.2.5 External Document References

Some *Client Requirements* can be defined in separate documents, such as specifications and other text documents, schematic drawings, and spreadsheets. In addition, building codes are practically never included in the project documentation; they are external documents. In the *Requirements* analysis, 21% of the *Requirements* were either references to external documents or hyperlinks (Section 4.2.2). This means that it is important to include this possibility in the *Requirements Model Specification*. The *IFC Specifications* have an element for this purpose, IfcDocumentReference, and it is used in my *Requirements Model Specification*.

6.2.5.1 IfcDocumentReference

Definition and description from IAI [*IFC 2004i* [86]]:

"IfcDocumentReference is a reference to the location of a document. The reference is given by a system interpretable Location attribute (e.g., an URL string) or by a human-readable location, where the document can be found, and an optional inherited internal reference ItemReference, which refers to a system interpretable position within the document. The optional inherited Name attribute is meant to have meaning for human readers. Optional document metadata can also be captured through reference to IfcDocumentInformation.

IfcDocumentReference provides a lightweight capability that enables a document to be identified solely by reference to a name by which it is commonly known. The reference can also be used to point to document information for more detail as required."

EXPRESS specification:

```
ENTITY IfcDocumentReference
    SUBTYPE OF (IfcExternalReference);
    INVERSE
        ReferenceToDocument : SET [0:1] OF IfcDocumentInformation FOR
        DocumentReferences;
    WHERE
        WR1 : EXISTS(Name) XOR EXISTS(ReferenceToDocument[1]);
END_ENTITY;
```

6.2.6 Bounding Elements and Building Systems

IFC Specifications include two mechanisms which are crucial for the linkage of *Indirect Requirements:* IfcRelSpaceBoundary defining the *Bounding Elements* for a *Space* and IfcSystem defining the systems as an organized combination of their parts. As described in Section 6.1.2, both are essential for the recognition of the objects affected by the *Direct Requirements* defined for a *Space*.

6.2.6.1 Bounding Elements

Definition and description from IAI [*IFC 2004j* [87]]:

"The space boundary (IfcRelSpaceBoundary) defines the physical or virtual delimiter of a space as its relationship to the surrounding elements.

In the case of physical space boundary, the placement and shape of the boundary may be given, and the building element, providing the boundary, is referenced. In the case of virtual space boundary, the placement and shape of the boundary may be given, but no building element is referenced. The exact definition of how space boundaries are broken down depends on the view, more detailed conventions on how space boundaries are decomposed can only be given at the domain or application type level.

Example: In an architectural or FM related view, a space boundary is defined from the inside of the space and does not take the providing building element into account. A plane area (even if the building element changes) is still seen as a single space boundary. In an HVAC related view, the decomposition of the space boundary depends on the material of the providing building element and the adjacent spaces behind."

IfcRelSpaceBoundary is related to my *Requirements Model Specification* as the method to find the *Indirect Requirements* for the *Bounding Elements*. These *Requirements* are not defined directly in the *Requirements Model; Indirect Requirements* are derived from related objects recognized in the *DPM Models* (Section 7.1.2.1 and Figure 86).

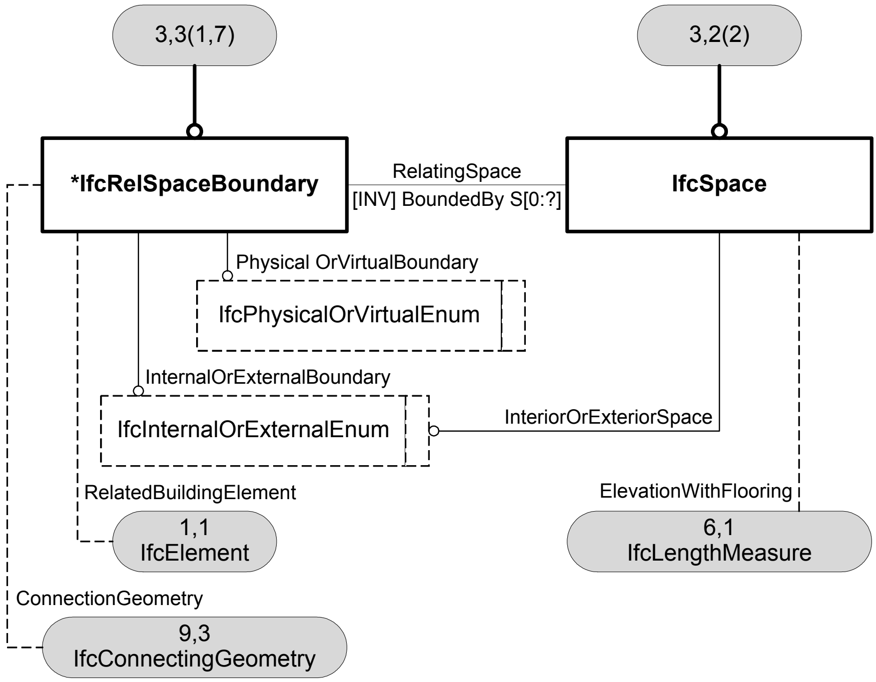

Figure 41: IfcRelSpaceBoundery relations

6.2.6.2 IfcSystem

Definition and description from IAI [*IFC 2004k* [88]]:

IfcSystem is "Organized combination of related parts within an AEC product, composed for a common purpose or function or to provide a service. System is essentially a functionally related aggregation of products. The grouping relationship to one or several instances of IfcProduct (the system members) is handled by IfcRelAssignsToGroup. The use of IfcSystem often applies to the representation of building services related systems, such as the piping system, cold water system, etc.

EXPRESS specification:

```
ENTITY IfcSystem
    SUBTYPE OF (IfcGroup);
    INVERSE
        ServicesBuildings   :   SET [0:1] OF IfcRelServicesBuildings FOR RelatingSystem;
    WHERE
        WR1   :   SIZEOF (QUERY (temp <* SELF\IfcGroup.IsGroupedBy.RelatedObjects |  NOT
        ('IFC PRODUCTEXTENSION.IFCELEMENT' IN TYPEOF(temp)))) = 0;
END_ENTITY;
```

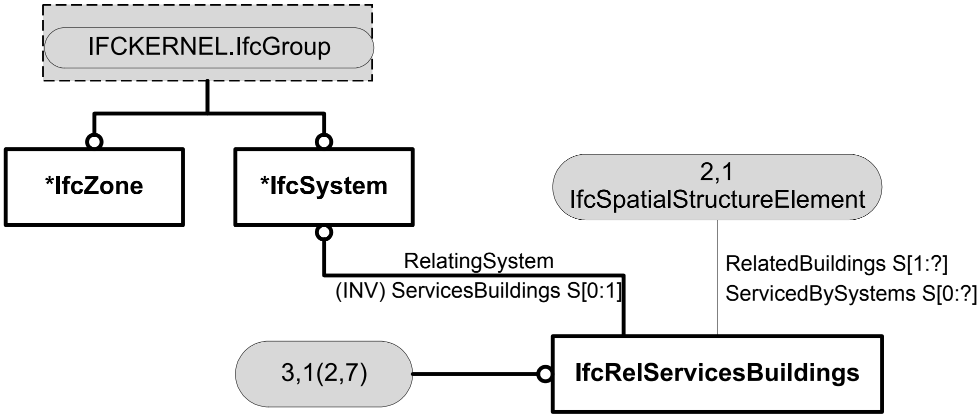

Figure 42: IfcSystem relations

6.2.6.3. SystemsUsed in my Requirements Model

IfcSystems are based on a generic grouping mechanism. A system is aggregated from the objects which are defined to be a part of the system in the *DPM Model*. There is no explicit list of the different systems in the *IFC Specification*. However, I believe that the definition and naming of different systems should be part of the standardization of the *IFC Specifications*, but it is not in the scope of my research. This standardization is proposed as an addition in *IFC Specifications* (Section 8.2.4.2). The only *Direct Requirements* for systems defined in my *Specification* are related to the BuildingEnvelope and CirculationSystem, which are part of the architectural design. However, to be able to show the connections to the other systems in *DPM Models* I have defined the following 12 systems in my *Requirements Model Specification:*

- BuildingEnvelope
- CirculationSystem
- StructuralSystem
- HvacSystem
- PlumbingSystem
- GasSupplySystem
- ElectricalSystem

- TelecomSystem
- ItNetworkSystem
- AudioSystem
- SecuritySystem
- FireSafetySystem

6.3 Requirements Model Specification

6.3.1 Requirements Model Hierarchy

The basic hierarchy of my *Requirements Model Specification* follows the structure of the *IFC Specifications*. The basic 5 levels are project, site, building, building story and *Space*. Systems are a separate group inside the project (Figure 43).

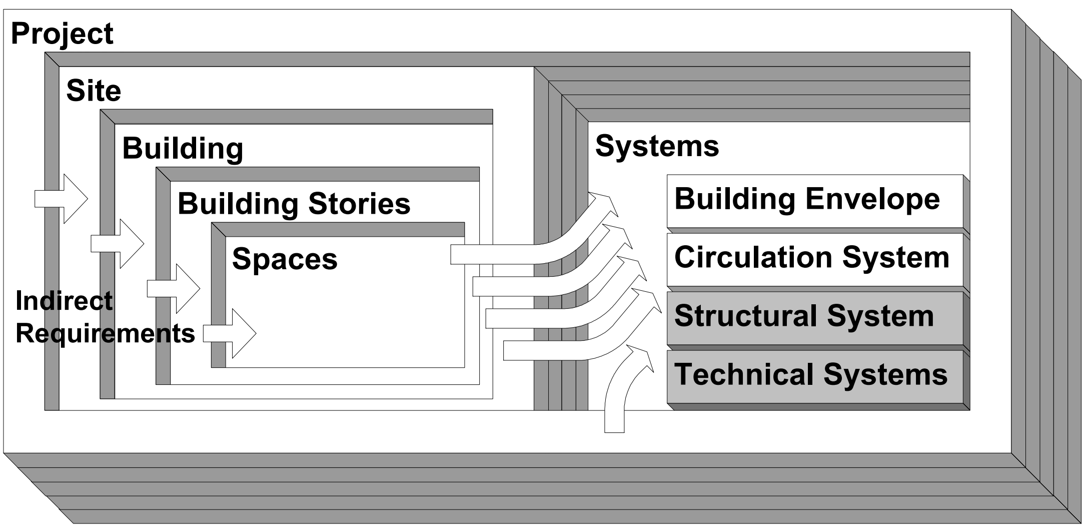

Figure 43: Levels of detail in the *Requirements Model Specification*

The principle in my *Requirements Model Specification* is that the *Indirect Requirements* cannot be linked to the objects on the upper levels in the hierarchy. That means, for example, that the site can create *Indirect Requirements* to the building, and building to the *Spaces*, but not vice versa. Any of these levels can create *Requirements* to any of the systems (Figure 43).

The *Specification* covers 300 *Requirements* in 14 main and 35 sub-categories (Appendix B3, Table 10). It is based on a synthesis of two large, widely used *Requirements Hierarchies* (Section 3.2.2), analysis of *Requirements* in five *Building Programs* (Chapter 4), and *Spatial Requirements* in the current *IFC Specifications* (Section 6.2). In addition, some *Requirements* are based on the comments from CSIRO [*Drogemuller, 2004* [89]].

These *Requirements* are organized into 7 main-level and 30 sub-level *Requirements Objects* which have direct links to 5 levels of detail and 2 systems in the *Building Product Model* plus indirect links to 4 levels of detail and 12 systems. These levels and systems are described in Sections 6.3.1.1–6.3.1.6 and *Requirements Objects* are documented in Sections 6.3.2–6.3.12. Some of the *Requirements*, such as load capacity, lighting *Requirements*, etc., relate more to the systems than to the architectural design. However, they are often defined in connection to a *Space*, and thus are included in my *Specification*. This issue is discussed in more detail in Section 8.3.1.2.

Table 10 documents the different *Requirement* types on different levels of detail in my *Requirements Model Specification*.

Table 10: *Requirement* type distribution in the *Requirements Model Specification*

	Requirements Attributes (RA)	Requirements Descriptions (RD)	**SVRs in total (RA+RD)**	Description lists = MVRs	**Direct Requirements in total**	Requirements objects with indirect links	Total number of indirect links
Project Requirements	31	20	**51**	11	**62**	30	30
Site Requirements	12	22	**34**	9	**43**	27	27
Building Requirements	51	41	**92**	6	**98**	90	113
Story Requirements	0	4	**4**	0	**4**	4	7
Space Requirements	40	20	**60**	14	**74**	39	39
Envelope Requirements	8	3	**11**	1	**12**	0	0
Circulation Requirements	1	0	**1**	6	**7**	0	0
In total	**143**	**110**	**253**	**47**	**300**	**190**	**216**

6.3.1.1 Project

Existing IFC entity: IfcProject

Requirement Types

- This group includes *Requirements* which affect the selection of the location (site). Some *Requirements* in this group are relevant only before the actual design process, but they should be stored in the *Requirements Model* for future evaluation purposes.

Typical Examples

- Required infrastructure: Roads, electrical and water supply, sewage system, etc.
- Services: Public transportation, commercial services, etc.

6.3.1.2 Site

Existing IFC entity: IfcSite

Requirement Types

- This group includes both *Requirements* and limitations. *Requirements* are *Properties* which are requested. Limitations are *Properties* which are not allowed or define limits to allowed solutions.

Typical Examples

- *Requirements:* Number of parking spaces, emergency, vehicular, bicycle and pedestrian access, outdoor spaces and activities, etc.
- Limitations: Building location, footprint and height, which can be even location-specific in different areas of the site, existing buildings and vegetation which must be preserved, maximum allowed noise level, etc.

6.3.1.3 Building

Existing IFC entity: IfcBuilding

Requirement Types

- This group includes *Requirements* defining the overall building performance and quality. These *Requirements* often affect the systems serving the building.

Typical Examples

- Total energy consumption, shading and glare *Properties* of the building, wind effects, emissions (odor, heat, and noise), flexibility, etc.

6.3.1.4 Story

Existing IFC entity: IfcBuildingStorey

Requirement Types

- This group includes story-specific *Requirements*, which often affect the building envelope *Requirements*.

Typical Examples

- In most cases accessibility and security *Requirements*, such as handicap access, window and door protection.

6.3.1.5 Space

Existing IFC entities: IfcSpace and IfcSpaceProgram

IfcSpace is defined very widely. It can be defined by physical or imaginary boundaries, and it can be also a group of *Spaces:*

Definition and description of *Space* from IAI [*IFC 2004d* [90]]

"A space represents an area or volume bounded actually or theoretically. Spaces are areas or volumes that provide for certain functions within a building.
A space is (if specified) associated to a building storey (or in case of exterior spaces to a site). A space may span over several connected spaces. Therefore a space group provides for a collection of spaces included in a storey. A space can also be decomposed in parts, where each part defines a partial space. This is defined by the composition type attribute of the supertype IfcSpatialStructureElement which is interpreted as follow:

- COMPLEX = space group
- ELEMENT = space
- PARTIAL = partial space"

As documented in Section 6.2.2.2, IfcSpace includes several *Property Sets* defining *Requirements* for the *Space*. I argue that this is a wrong solution;

Requirements should be part of the IfcSpaceProgram, not IfcSpace (Sections 6.1.4 and 6.2.2).

My *Requirements Model Specification* consists of *Space Program Instance* and *Space Program Type* (Sections 5.1.1 and 6.1.7). Most *Requirements* are related to the *Space Program Type*.

Requirement Types

- This group includes all *Space*-specific *Requirements*. Many of these *Requirements* often affect *Bounding Elements*, including the building envelope, and technical systems.

Typical Examples

- Area, adjacency *Requirements* to other *Spaces*, indoor air quality, lighting, materials, equipment, furniture, etc.

6.3.1.6 System

Existing IFC entity: IfcSystem

Definition of the IfcSystem is in Section 6.2.6.2.

Requirement Types

- The *Direct Requirements* for structural and technical systems are not in the scope of my research. However, the *Indirect Requirements* to these systems from the *Direct Requirements* for architectural design are shown in the *Requirements Model Specification*.

Typical Examples

- Building envelope: Thermal and sound insulation, solar protection, etc.
- Circulation system: Circulation area ratio compared to the programmed area, corridor, elevator and escalator *Requirements*, etc.

6.3.2 Links between the Requirements and DPM Models

The methods which are used in the *IFC Specification* to link objects to each other in one *Instantiated Model* cannot be used between objects in two *Instantiated*

Models. This means that those methods cannot be used between *Requirements and DPM Models*, because they are different *Models*, i.e., different data sets. This separation of *Instantiated Models* (data sets describing a project) is necessary for practical implementation, efficient data management, control of user rights, and comparison of *Requirements* and solutions (Section 5.1.1).

The *Project Requirements* are stored in their own *Model*, and they are linked to the *Design Model*, which is the "container" for design data. In addition, the solution must be able to support automated linkage between *Requirements and Design Models*, because manual linkage is a time-consuming and error-prone process when there can be hundreds, even thousands, of links between the *Instantiated Models*. The separation of *Instantiated Models* and the need for automated linkage means that the current *IFC Specifications* must be revised, because there is no appropriate method to do this.

The links between *Requirements and Design Models* are from the *Design Objects* to the *Requirements Objects*. However, the links between the different *Design Models* must be two-directional, for example, a column *Instance* in the architectural and structural *Model* must be linked in both directions (Figure 44).

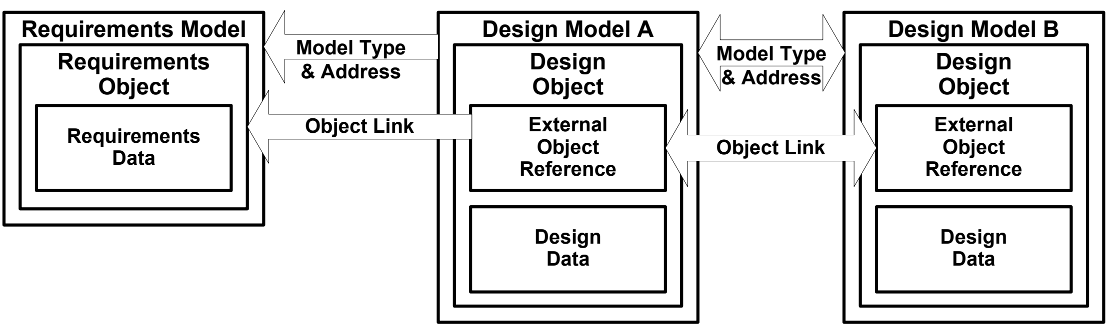

Figure 44: Links between objects in the *Requirements and Design Models*

The link information must include the following elements:

- Type of the *Model: Requirements, Design, Production, or Maintenance Model*. One object can be linked to several *Models*, because, as mentioned in Section 5.1.1, there is a need to have different *Models* for different design and contractor domains. This means that there will be a need to divide *Design and Production Models* further into several categories. This does not change the principle, and this division is not in my research scope. Thus, this, as well as some other aspects of the links between *Models*, is one the proposed topics for future research (Section 8.3.1.6).

- Location of the *Model:* Address where the linked *Model* is stored. This can be a URL, address in the *Model Server* database, or some other address depending on the technical solution.
- Object(s) which is/are linked: For example, each *Space* object in the *Design Model* is linked to the *Space Program Instance* which contains its *Requirements*. One object can also be linked to several objects in another *Model*, for example, a slab in the architectural *Design Model* can be divided into several parts in the structural *Design Model* or in the *Production Model*.

The structure of the external *Model* link in my *Requirements Model Specification* is based on the existing IfcExternalReference and IfcRelAssociates objects (Figure 45). I have added one new subclass into both: NewExternalObject-Reference (Section 6.3.2.3 and) and NewRelAssociatesExternalObject (Section 6.3.2.4). In addition, there is one new object and one new enumeration. The new object is NewModelInformation (Section 6.3.2.5) and the new enumeration is NewExternalReferencedObjectTypeEnum for the Object types, which now consists of an user-definable value and the four main *Model* types: *Requirements, Design, Production, and Maintenance Models* (Section 6.3.2.6). This enumeration is the only entity which needs to be expanded to enable several design and contractor domain *Models.* The "USERDEFINED" value can of course be used for temporary expansion purposes, but for standardization reasons the different *Model* types in the *Design and Production Model* categories should be included in the enumeration list (Section 8.3.1.6).

The abbreviation (ABS) in the EXPRESS-G graphics refers to an abstract super-type, i.e., an object which cannot be directly used in an *Instantiated Model*. These super-types define the common properties for their sub-types which can be instantiated in the *Model*. Grey entities in the EXPRESS-G graphics are existing IFC elements; white entities are new elements (Figure 45).

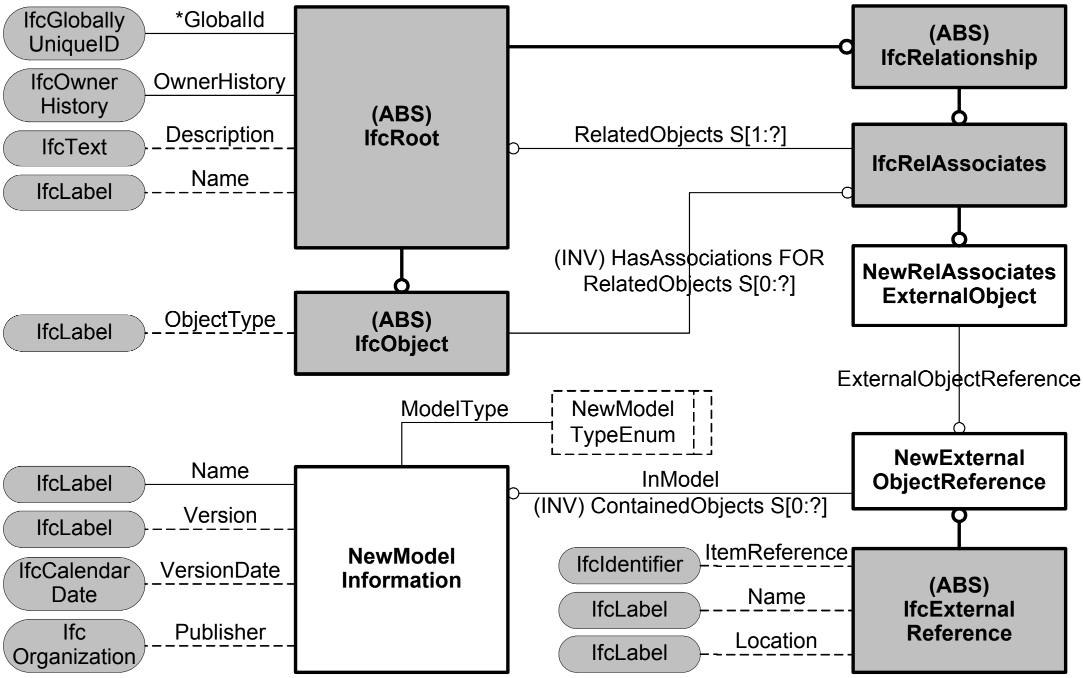

Figure 45: Location and structure of the object link between *Models* in the *IFC Specification*

6.3.2.1 Modified IfcExternalReference

ENTITY IfcExternalReference

ABSTRACT SUPERTYPE OF (ONE OF (IfcLibraryReference, IfcClassificationReference, IfcDocumentReference, ***NewExternalObjectReference***));

Location : OPTIONAL IfcLabel;

ItemReference : OPTIONAL IfcIdentifier;

Name : OPTIONAL IfcLabel;

WHERE

WR1 : EXISTS(ItemReference) OR EXISTS(Location) OR EXISTS(Name);

END_ENTITY;

6.3.2.2 Modified IfcRelAssociates

ENTITY IfcRelAssociates

SUPERTYPE OF (ONE OF (IfcRelAssociatesClassification, IfcRelAssociatesDocument, IfcRelAssociatesLibrary, ***NewRelAssociatesExternalObject***))

SUBTYPE OF (IfcRelationship);

RelatedObjects : SET [1:?] OF IfcRoot;

WHERE

WR1 : SIZEOF(QUERY(temp <* RelatedObjects | NOT(('IFCKERNEL.IFCOBJECT' IN TYPEOF(temp)) OR ('IFCKERNEL.IFCPROPERTYDEFINITION' IN TYPEOF(temp))))) = 0;

END_ENTITY;

6.3.2.3 NewExternalObjectReference

ENTITY NewExternalObjectReference

SUBTYPE OF (IfcExternalReference);

InModel : NewModelInformation;

END_ENTITY;

6.3.2.4 NewRelAssociatesExternalObject

ENTITY NewRelAssociatesExternalObject

SUBTYPE OF (IfcRelAssociates);

ExternalReference : NewExternalObjectReference;

END_ENTITY;

6.3.2.5 NewModelInformation

ENTITY NewModelInformation;

ModelType : IfcLabel;

Name : IfcLabel;

Version : OPTIONAL IfcLabel;

Publisher : OPTIONAL IfcOrganization;

VersionDate : OPTIONAL IfcCalendarDate;

INVERSE

ContainedObjects : SET [0:?] OF IfcExternalObjectReference FOR InModel;

END_ENTITY;

6.3.2.6 NewModelTypeEnum

TYPE NewModelTypeEnum = ENUMERATION OF

(REQUIREMENTSMODEL,

DESIGNMODEL,

PRODUCTIONMODEL,

MAINTENANCEMODEL,

USERDEFINED,

NOTDEFINED);

END_TYPE;

6.3.3 Requirement Object

The NewRequirement object is a subtype of IfcControl and an abstract supertype of all *Requirements Objects* (Figure 46). This decreases the duplication of the repeated elements in the *Requirements Objects*. The *NewRequirement* inherits the following elements from IfcRoot, IfcObject and IfcControlObject. Thus, they are not presented in the definition of NewRequirement object.

GlobalId : IfcGloballyUniqueId;
OwnerHistory : IfcOwnerHistory;
Name : OPTIONAL IfcLabel;
Description : OPTIONAL IfcText;
ObjectType : OPTIONAL IfcLabel;

The DocumentReference and CodeReference in the NewRequirement object are based on the IfcDocumentReference, and their purpose is to provide links to external documents (Section 6.2.5). The purpose of the RequirementsIntent, Design Intent and Approval Status elements is to provide a method to include additional information about the reason for the *Requirements*, explanation for the design solution, and information about the approval process (Section 6.2.4). The NewRequirementDescriptionList is documented in the NewRequirementElement (Section 6.3.4)

ENTITY ***NewRequirement****;*
ABSTRACT SUPERTYPE
SUBTYPE OF (IfcControl);
DocumentReference : OPTIONAL IfcDocumentReference;
CodeReference : OPTIONAL IfcDocumentReference;
RequirementsIntent : OPTIONAL NewRequirementDescriptionList;
DesignIntent : OPTIONAL NewRequirementDescriptionList;
ApprovalStatus : OPTIONAL IfcApprovalStatus;
END_ENTITY;

Name	**Description**
DocumentReference	*References to documents related to the requirements group in each Requirements Object*
CodeReference	*References to codes related to the requirements group in each Requirements Object*

RequirementsIntent	*Description of the intent of the defined requirements in each Requirements Object; a list which can contain an unlimited number of IfcTexts*
DesignIntent	*Description of the intent of design solutions related to the requirements group in each Requirements Object; a list which can contain an unlimited number of IfcTexts*
ApprovalStatus	*Approval status of the requirements group and/or related design solutions in each Requirements Object*

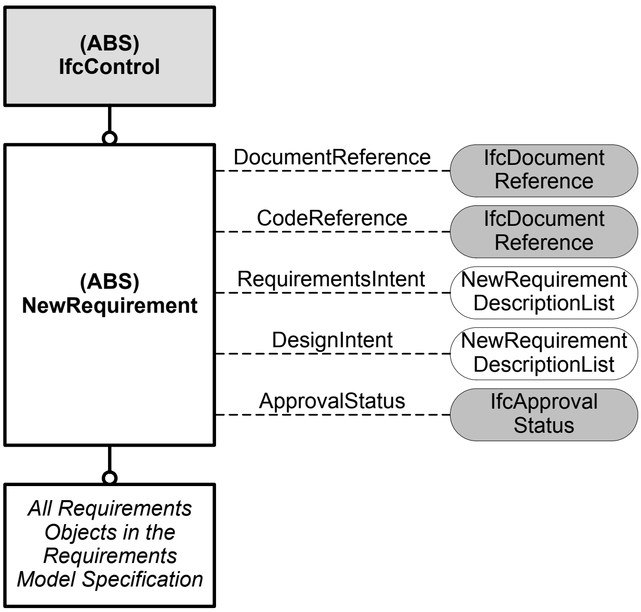

Figure 46: NewRequirement object

6.3.4 Requirement Element

As documented in Section 6.2.3 the current IfcOwnerHistory and GUID mechanisms cause granularity problems in the *Requirements Model Specification*. Each *Requirement* in a *Model* can have a different owner and source, and they can be edited separately, which means that storing the whole *Requirements Object* into the "history database" every time a *Requirement* is changed is not a good solution. Thus, I created a new RequirementElement object which contains the necessary information for *Requirements Management*. All *Requirements* are *Requirement Attributes* or *Requirement Descriptions*. In some *Requirements* the type is a list of *Requirement Descriptions*, which means that these *Requirements* can consist of several values (e.g., they are *Multi-Value Requirements*), others can have only a single value (*Single-Value Requirements*, Section 5.4 and Table 10). The whole structure consists of one super-type and 13 subtypes (Figure 47).

Because all *Requirements Attributes* have exact values which can be measured, it is in principle possible to verify if the *Design Model* and/or the building meet them. In some cases the verification is very simple, such as calculation or measurement of the *Space* area, but verification can also demand methods which are not widely used in the AEC industry, for example, extensive thermal or lighting simulation in the design stage or long-term measurements in the building, such as continuous commissioning.

The IfcIdentifier is a unique identifier for each individual *Requirement* [*IFC 2004g* [91]]. It can be based on a user-defined ID controlled by the application, an application's own automatic ID system, or it can be based on the GUID in the *IFC Specifications*. Its main purpose is to enable identification of the *Requirements* in the "history database," e.g., all versions of the same *Requirement* must have the same ID, but a different time. However, the ID can sometimes be useful reference information if it is not too complicated, such as a typical *Space* number. For this purpose GUID is not useful, because it is so long and complicated. The use of GUID for "history database" purposes is possible in this case in spite of the identified problems (Section 6.2.3.3), because (1) the *Requirements History* is recorded inside a *Requirements Management* application, and (2) the user-interface of the *Requirements Management* application can limit the end-user's possibilities to edit *Requirements* by deleting existing *Requirements* and adding new ones for the same purpose. In design applications, such as CAD software, this is not possible.

The subtypes of the NewRequirementElement enable the use of defined data-types. They are based on the analysis of the values used in different *Requirements* (Table 11). In the last subtype, NewRequirementValueSelect, the use of IfcValueSelect enables selection of any data type defined in the *IFC Specifications*. This means, for example, that instead of having just IfcReal as the value of a *Requirement*, the value can be specified to represent IfcBoolean or IfcLinear-VelocityMeasure, for example. This improves the usefulness of the values, because these values include the unit system used in the *Model*, such as metric or imperial units. Likewise, IfcDateAndTimeSelect enables the use of date and time information in a specified format. I used NewRequirementValueSelect only for datatypes which are used 1-2 times in the *Requirements Model Specification*.

Table 11: RequirementElement subtype and datatype occurrences in the *Specification*

Subtype	Datatype	Occurrences
NewRequirementsDescription	IfcText	112
NewRequirementsDescriptionList	List of IfcText	47
NewRequirementsArea	IfcAreaMeasure	9
NewRequirementsInteger	IfcInteger	8
NewRequirementsCost	IfcMonetaryMeasure	9
NewRequirementsDistance	IfcPositiveLengthMeasure	4
NewRequirementsRatio	IfcPositiveRatioMeasure	23
NewRequirementsPower	IfcPowerMeasure	8
NewRequirementsReal	IfcReal	46
NewRequirementsSound	IfcSoundProperties	9
NewRequirementsTemperature	IfcThermodynamicTemperatureMeasure	9
NewRequirementsVolume	IfcVolumeMeasure	3
NewRequirementsValueSelect	IfcValueSelect	13
In total		**300**

Similarly, IfcActorSelect enables references to a person and/or organization and IfcDocumentSelect to documents, defined once in the *Model*, thus preventing the multiplication of the same data in the *Model* and ensuring coherent data management. IfcActorSelect and IfcDocumentSelect are existing definitions in the current *IFC Specifications*.

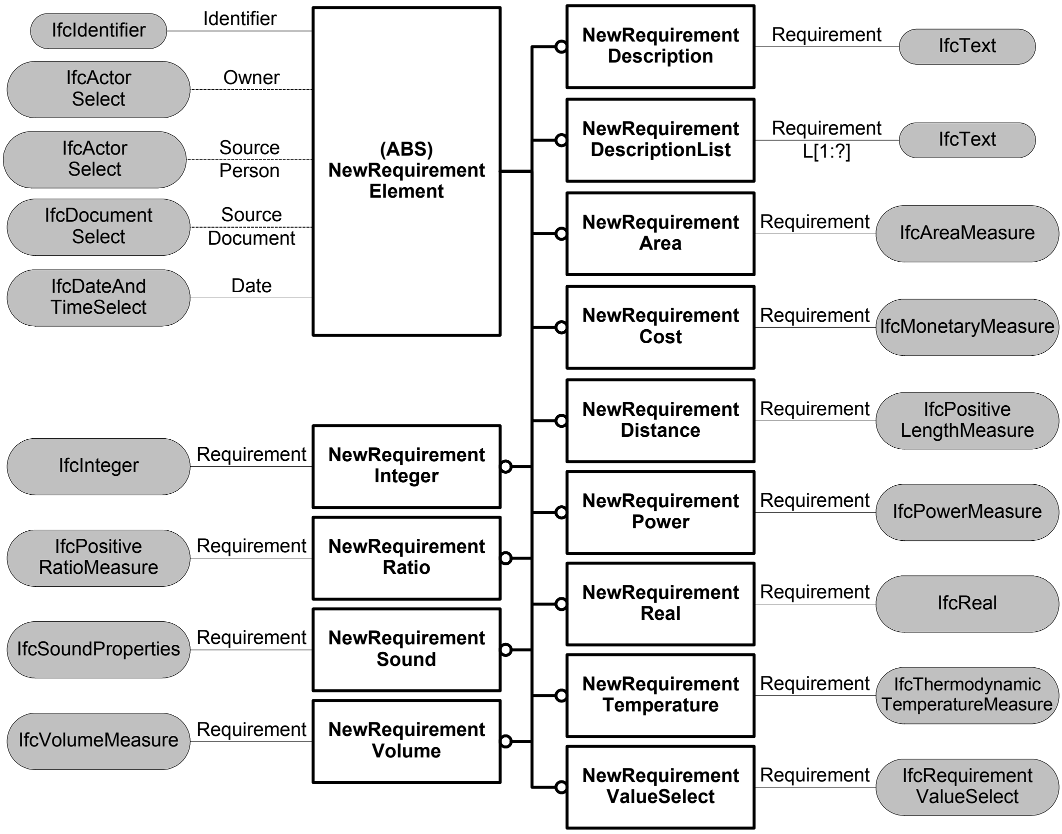

Figure 47: RequirementElement structure

6.3.4.1 Requirement Element

ENTITY NewRequirementElement

ABSTRACT SUPERTYPE OF (ONEOF(NewRequirementAttribute, NewRequirementDescription, NewRequirementDescriptionList));

Identifier : IfcIdentifier;

Owner : OPTIONAL IfcActorSelect;

SourcePerson : OPTIONAL IfcActorSelect;

SourceDocument : OPTIONAL IfcDocumentSelect;

Date : IfcDateAndTimeSelect;

END_ENTITY;

6.3.4.2 Requirement Description

ENTITY NewRequirementDescription

SUBTYPE OF NewRequirementElement;

Requirement : IfcText;

END_ENTITY;

6.3.4.3 Requirement Description List

ENTITY NewRequirementDescriptionList

SUBTYPE OF NewRequirementElement;

Requirement : LIST [1:?] OF IfcText;

END_ENTITY;

6.3.4.4 Requirement Area

ENTITY NewRequirementArea

SUBTYPE OF (NewRequirementElement);

Requirement : IfcAreaMeasure;

END_ENTITY;

6.3.4.5 Requirement Cost

ENTITY NewRequirementCost

SUBTYPE OF (NewRequirementElement);

Requirement : IfcMonetaryMeasure;

END_ENTITY;

6.3.4.6 Requirement Distance

ENTITY NewRequirementDistance

SUBTYPE OF (NewRequirementElement);

Requirement : IfcPositiveLengthMeasure;

END_ENTITY;

6.3.4.7 Requirement Integer

ENTITY NewRequirementInteger

SUBTYPE OF (NewRequirementElement);

Requirement : IfcInteger;

END_ENTITY;

6.3.4.8 Requirement Power

ENTITY NewRequirementPower

SUBTYPE OF (NewRequirementElement);

Requirement : IfcPowerMeasure;

END_ENTITY;

6.3.4.9 Requirement Ratio

ENTITY NewRequirementRatio

SUBTYPE OF (NewRequirementElement);

Requirement : IfcPositiveRatioMeasure;

END_ENTITY;

6 .3.4.10 Requirement Real

ENTITY NewRequirementReal

SUBTYPE OF (NewRequirementElement);

Requirement : IfcReal;

END_ENTITY;

6 .3.4.11 Requirement Sound

ENTITY NewRequirementSound

SUBTYPE OF (NewRequirementElement);

Requirement : IfcSoundProperties;

END_ENTITY;

6 .3.4.12 Requirement Temperature

ENTITY NewRequirementTemperature

SUBTYPE OF (NewRequirementElement);

Requirement : IfcThermodynamicTemperatureMeasure;

END_ENTITY;

6 .3.4.13 Requirement Volume

ENTITY NewRequirementVolume

SUBTYPE OF (NewRequirementElement);

Requirement : IfcVolumeMeasure;

END_ENTITY;

6 .3.4.14 Requirement Value Select

ENTITY NewRequirementValueSelect

SUBTYPE OF (NewRequirementElement);

Requirement : IfcValueSelect;

END_ENTITY;

6.3.5 Basic Relations between the Requirements and DPM Models

Section 6.3.3 defines the repeated standard elements of all *Requirements Objects*. The only *Requirements* elements which are from the current *IFC Specifications* are in the *Space Program Instance* (Section 6.3.10.1). These, as well as the existing data types, are presented in Times New Roman Normal. New entities are formatted using *Times New Roman Italic* in the EXPRESS definitions. All entities formatted using ***Times New Roman Bold Italic*** are references to new *Requirements Objects* which are separated from the main Object for linkage reasons.

The illustration of the *IFC Specifications*, "*Design, Production, and Maintenance Models*," is identical in all diagrams (Figure 48 – Figure 73). The left part representing the *Requirements Model Specification*, "Requirements Model," changes in these diagrams to illustrate the different parts of the *Specification*. The basic structure and direct links between the *Models* are presented in Figure 48.

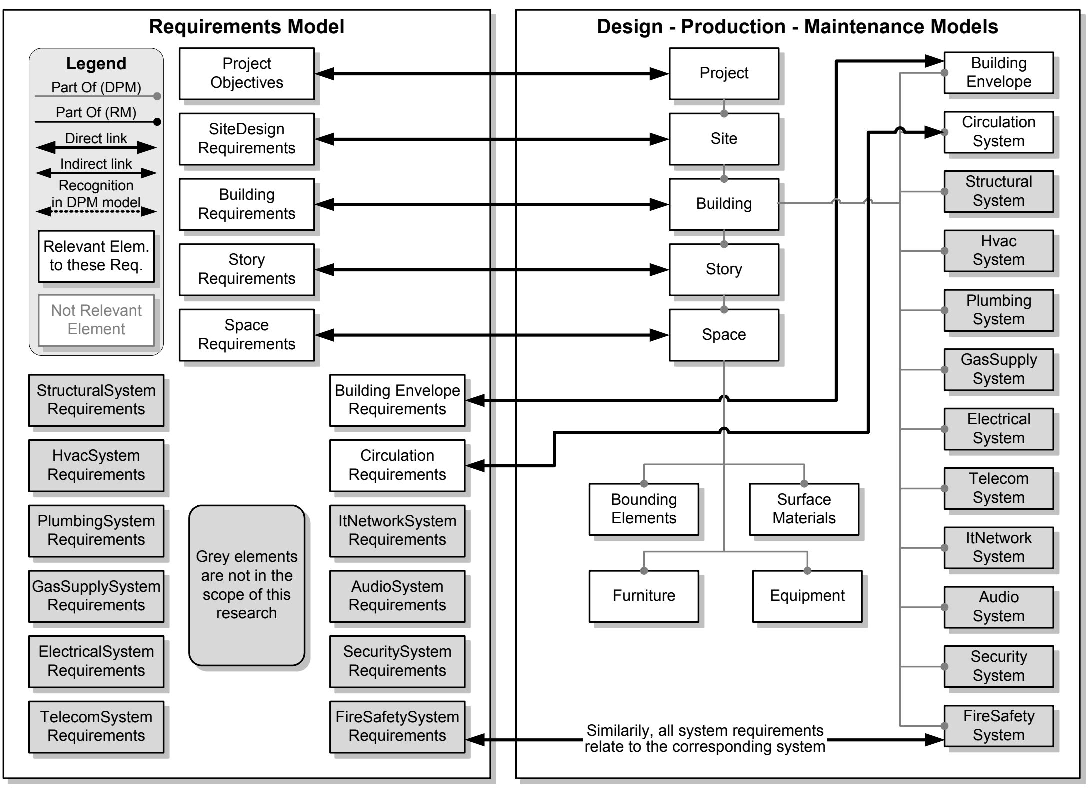

Figure 48: Basic relations between *Requirements and DPM models*

6.3.6 *Requirements Object: Project Objectives*

Project Objectives are *Requirements* which are used in a project already before the site selection stage. Some have impact in the design solutions on site, building and system level, but some are effective only at the site selection phase, and cannot be influenced afterwards by the project. Examples of such *Requirements* are Infrastructure, Services, Catastrophe Risks, etc. They are part of the selected environment and some of them can change by the actions of people outside of the project; for example, even if the availability of food services was a selection criterion of the site, the project can seldom influence the continued availability of these services after the site is selected.

6.3.6.1 Project Objectives

- Main object to which the other project *Requirements* are linked (Figure 49)
- Attribute set which defines general design objectives and size of the project
- Direct link to *Design Model:* Project (IfcProject)
- Indirect links: None

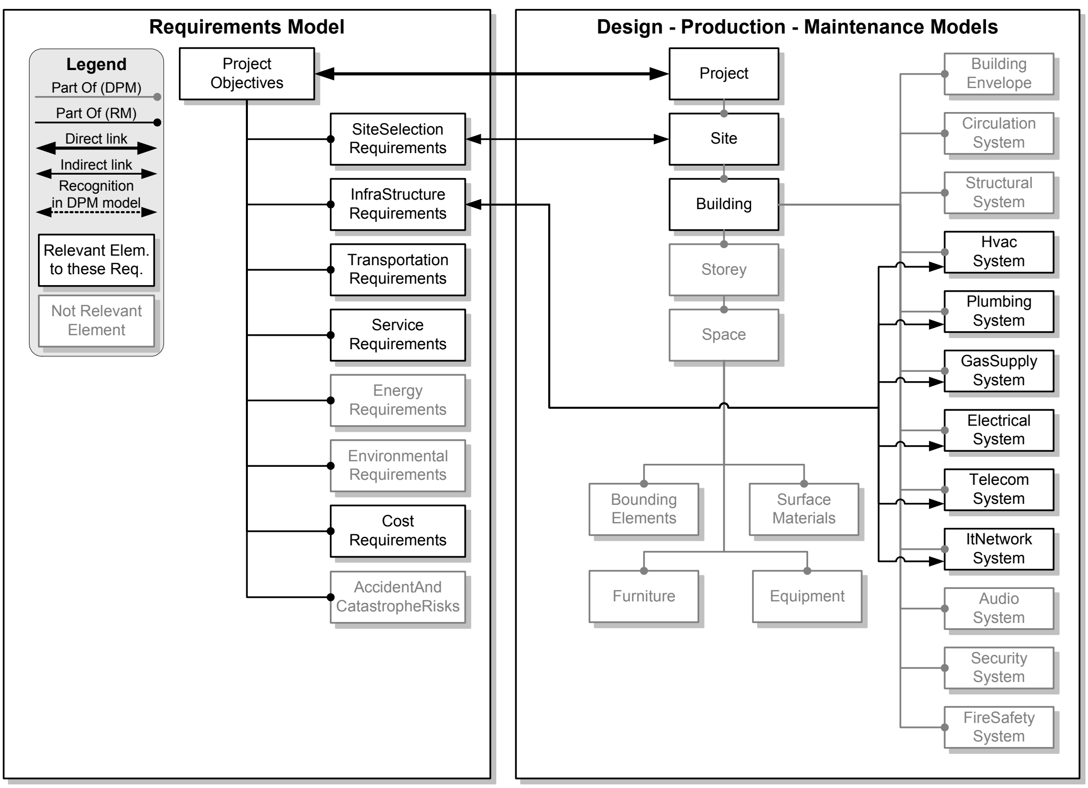

Figure 49: Project objectives 1/4

ENTITY ***NewProjectObjectives****;*

SUBTYPE OF (*NewRequirement*);

GeneralObjectives : OPTIONAL NewRequirementDescription;

SiteRequirements : OPTIONAL ***NewSiteRequirements****;*

InfrastrucutreRequirements : OPTIONAL ***NewInfrastructureRequirements****;*

TransportationRequirements : OPTIONAL ***NewTransportationRequirements****;*

ServiceRequirements : OPTIONAL ***NewServiceRequirements****;*

SustainablityRequirements : OPTIONAL ***NewSustainabilityRequirements****;*

CostRequirements : OPTIONAL ***NewCostRequirements****;*

ProjectRiskRequirements : OPTIONAL ***NewProjectRiskRequirements****;*

END_ENTITY;

Name	**Description**
GeneralObjectives	*Description of the general project objectives*

6.3.6.2 Site Selection Requirements

- Attribute set which defines *Requirements* for the site *Properties* serving as one site selection criterion (Figure 49)
- Direct link to *Design Model:* Project (IfcProject)
- Indirect links: Site (IfcSite)

ENTITY ***NewSiteSelectionRequirements****;*

SUBTYPE OF (*NewRequirement*);

GeographicalLocation : OPTIONAL NewRequirementDescription;

SiteArea : OPTIONAL NewRequirementArea;

SiteImage : OPTIONAL NewRequirementDescription;

SolarAvailability : OPTIONAL NewRequirementDescription;

SoilType : OPTIONAL NewRequirementDescription;

END_ENTITY;

Name	**Description**
Geographical Location	*Description of the geographical location requirements*
SiteArea	*Target value for the site area size*
SiteImage	*Description of the requirements for site image requirements*
SolarAvailability	*Description of the requirements for solar availability*
SoilType	*Description of the requirements for soil type (excavation and foundation requirements)*

6.3.6.3 Infrastructure Requirements

- Attribute set which defines *Requirements* for local infrastructure. Some *Requirements* can serve as basic information for design of technical systems; one of the site selection criteria (Figure 49)
- Direct link to *Design Model:* Project (IfcProject)
- Indirect links: Electrical, gas supply, HVAC, IT network, plumbing, and telecom systems (IfcSystem – ElectricalSystem, GasSupplySystem, HvacSystem, ItNetworkSystem, PlumbingSystem, TelecomSystem)

*ENTITY **NewInfrastructureRequirements**;*

SUBTYPE OF (*NewRequirement*);

ElectricityNetwork : OPTIONAL NewRequirementDescription;

ItNetwork : OPTIONAL NewRequirementDescription;

TelecomNetwork : OPTIONAL NewRequirementDescription;

GasSupplyInfra : OPTIONAL NewRequirementDescription;

CoolingSupplyInfra : OPTIONAL NewRequirementDescription;

HeatingSupplyInfra : OPTIONAL NewRequirementDescription;

WaterSupplyInfra : OPTIONAL NewRequirementDescription;

SewageInfra : OPTIONAL NewRequirementDescription;

RoadInfra : OPTIONAL NewRequirementDescription;

WasteInfra : OPTIONAL NewRequirementDescription;

END_ENTITY;

Name	**Description**
ElectricityNetwork	*Description of the requirements for the local electricity network infrastructure*
ITNetwork	*Description of the requirements for the local information network infrastructure*
TelecomNetwork	*Description of the requirements for the local telecommunication network infrastructure*
GasSupplyInfra	*Description of the requirements for the local gas supply infrastructure*
CoolingSupplyInfra	*Description of the requirements for the local cooling water supply infrastructure*
HeatingSupplyInfra	*Description of the requirements for the local heating water supply infrastructure*
WaterSupplyInfra	*Description of the requirements for the local water supply infrastructure*
SewageInfra	*Description of the requirements for the local sewage infrastructure*
RoadInfra	*Description of the requirements for the local road infrastructure*
WasteInfra	*Description of required local waste services*

6.3.6.4 Transportation Requirements

- Attribute set which defines the accessibility and transportation *Requirements* of the project serving as one of the site selection criteria (Figure 49)
- Direct link to *Design Model:* Project (IfcProject)
- Indirect links: None

ENTITY ***NewTransportationRequirements****;*

SUBTYPE OF (*NewRequirement*);

CarAccess : OPTIONAL NewRequirementDescription;

BikeAccess : OPTIONAL NewRequirementDescription;

PedestrianAccess : OPTIONAL NewRequirementDescription;

PublicTransportation : OPTIONAL NewRequirementDescription;

PublicTransportationDistance : OPTIONAL NewRequirementDistance;

PublicTransportationFrequency : OPTIONAL NewRequirementAttribute;

AirportDistance : OPTIONAL NewRequirementDistance;

END_ENTITY;

Name	**Description**
CarAccess	*Requirements for car access to the site*
BikeAccess	*Requirements for bike access to the site*
PedestrianAccess	*Requirements for pedestrian access to the site*
PublicTransportation	*Availability and other general requirements for public transportation*
PublicTransportation Distance	*Maximum allowed distance to local public transportation*
PublicTransportation Frequency	*Minimum frequency of local public transportation during the activity hours*
AirportDistance	*Maximum allowed distance to airport*

6.3.6.5 Service Requirements

- Attribute set which defines *Requirements* for local services serving as one of the site selection criteria (Figure 49)
- Direct link to *Design Model:* Project (IfcProject)
- Indirect links: None

ENTITY ***NewServiceRequirements****;*

SUBTYPE OF (*NewRequirement*);

BusinessServices : OPTIONAL NewRequirementDescriptionList;

DaycareServices : OPTIONAL NewRequirementDescriptionList;

CommercialServices : OPTIONAL NewRequirementDescriptionList;

CulturalServices : OPTIONAL NewRequirementDescriptionList;

FoodServices : OPTIONAL NewRequirementDescriptionList;

RecreationalServices : OPTIONAL NewRequirementDescriptionList;

WelfareServices : OPTIONAL NewRequirementDescriptionList;

SecurityServices : OPTIONAL NewRequirementDescriptionList;

END_ENTITY;

Name	**Description**
BusinessServices	*Description of required local business services, such as banking, copying, courier, and car rental; a list which can contain an unlimited number of IfcTexts*
DaycareServices	*Description of required local children daycare and school services; a list which can contain an unlimited number of IfcTexts*
CommercialServices	*Description of required local commercial services, such as gas stations, laundry, and shops; a list which can contain an unlimited number of IfcTexts*
CulturalServices	*Description of required local cultural services, such as libraries, movies, and theaters; a list which can contain an unlimited number of IfcTexts*
FoodServices	*Description of required local food services, such as groceries, restaurants, cafes, and fast food services; a list which can contain an unlimited number of IfcTexts*
RecreationalServices	*Description of required local recreational services, such as parks, swimming halls, and gyms; a list which can contain an unlimited number of IfcTexts*
WelfareServices	*Description of required local welfare and healthcare services, such as dentist, healthcare centers, and hospitals; a list which can contain an unlimited number of IfcTexts*
SecurityServices	*Description of required local security services, such as police and services of security companies; a list which can contain an unlimited number of IfcTexts*

6.3.6.6 Energy Requirements

- Attribute set which defines energy consumption *Requirements* of the project (Figure 50)
- Direct link to *Design Model:* Project (IfcProject)
- Indirect links: Building (IfcBuilding), gas supply, HVAC, and electrical systems (IfcSystem – GasSupplySystem, HvacSystem, ElectricalSystem)

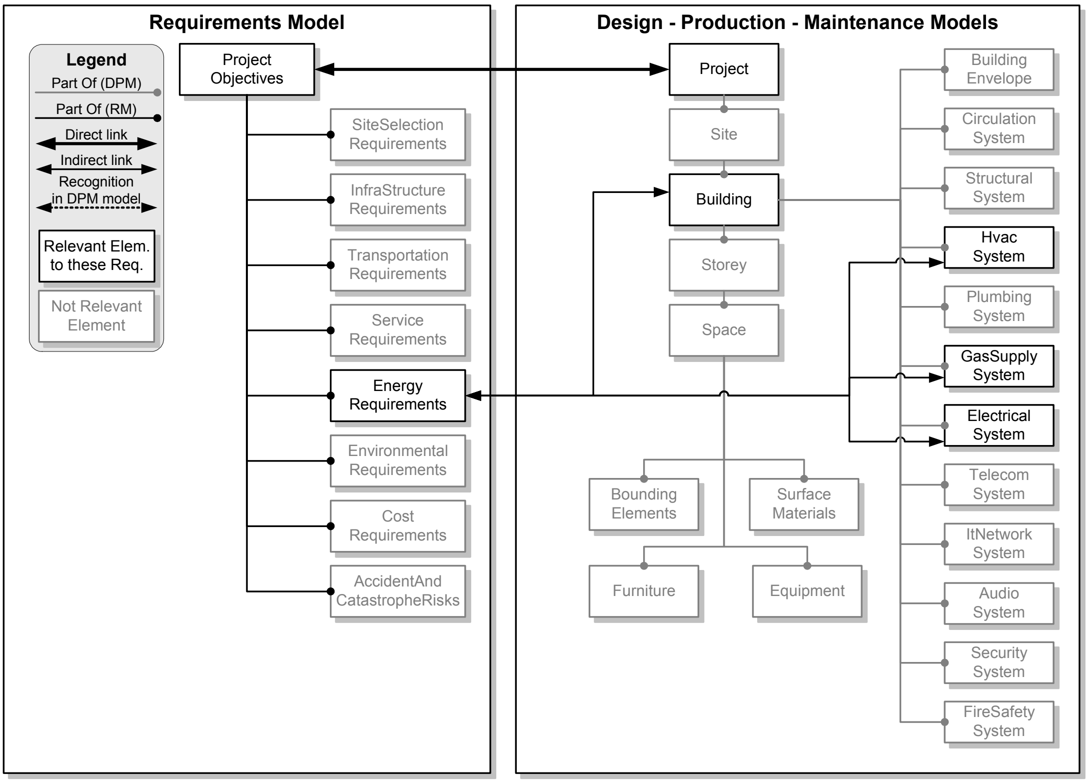

Figure 50: Project objectives 2/4 – Energy *Requirements*

ENTITY ***NewEnergyRequirements;***

SUBTYPE OF (*NewRequirement*);

TotalEnergyConsumption : OPTIONAL NewRequirementPower;

LightingEnergyConsumption : OPTIONAL NewRequirementPower;

TotalElectricalEnergyConsumption : OPTIONAL NewRequirementPower;

HeatingEnergyConsumption : OPTIONAL NewRequirementPower;

HeatingEnergySource : OPTIONAL NewRequirementDescription;

CoolingEnergyConsumption : OPTIONAL NewRequirementPower;

TotalHvacEnergyConsumption : OPTIONAL NewRequirementPower;

RecycledEnergy : OPTIONAL NewRequirementPower;

RenewableEnergyRatio : OPTIONAL NewRequirementRatio;

WaterConsumption : OPTIONAL NewRequirementVolume;

END_ENTITY;

Name	Description
TotalEnergy Consumption	*Target value for the yearly total energy consumption*
LightingEnergy Consumption	*Target value for the yearly lighting energy consumption*
TotalElectricalEnergy Consumption	*Target value for the yearly electrical energy consumption in total*
HeatingEnergy Source	*Description of the heating energy source requirements*
HeatingEnergy Consumption	*Target value for the yearly heating energy consumption*
CoolingEnergy Consumption	*Target value for the yearly cooling energy consumption*
TotalHvacEnergy Consumption	*Target value for the yearly HVAC energy consumption in total*
RecycledEnergy	*Target value for the yearly energy gain of air recycling and energy recovery systems*
RenewableEnergy Ratio	*Target ratio for the yearly use of solar and other renewable energy compared to the total energy consumption*
WaterConsumption	*Target value for the yearly water consumption*

6.3.6.7 Environmental Requirements

- Attribute set which defines *Requirements* for the targets for environmental pressure of the project, such as embedded resources and emissions (Figure 51)
- Direct link to *Design Model:* Project (IfcProject)
- Indirect links: Building (IfcBuilding)

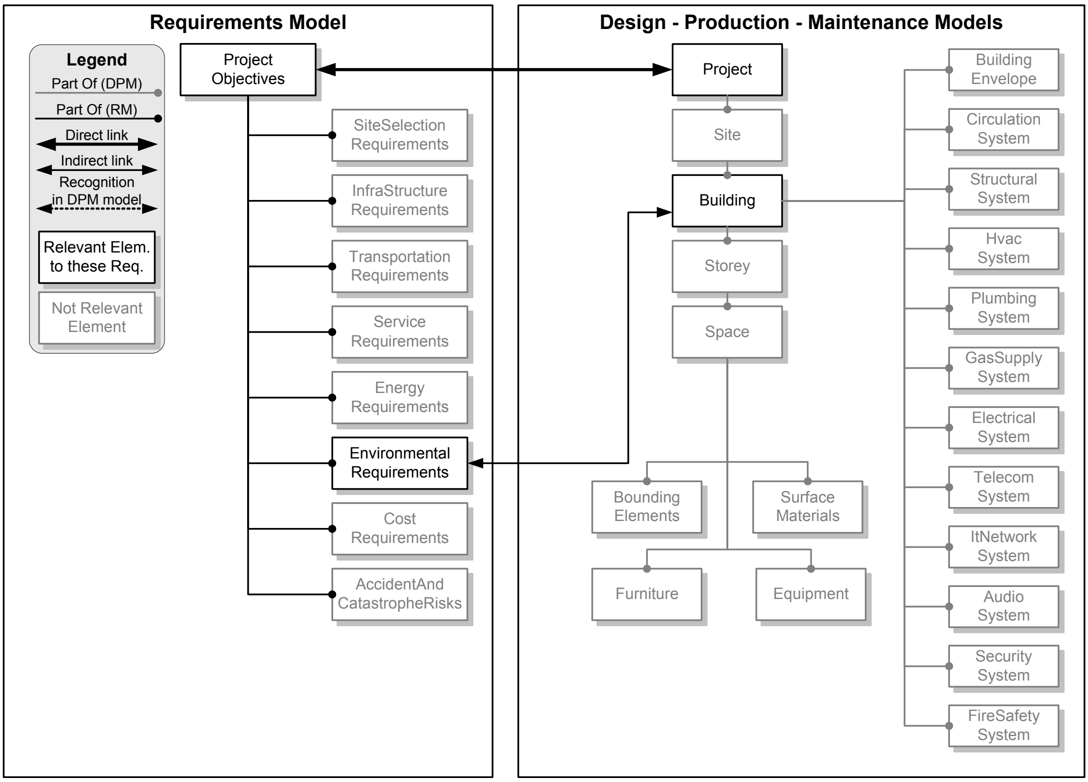

Figure 51: Project objectives 3/4 – Environmental *Requirements*

ENTITY ***NewEnvironmentalRequirements;***

SUBTYPE OF (*NewRequirement*);

MinRenewableMaterials : OPTIONAL NewRequirementRatio;

MaxNonRenewableMaterials : OPTIONAL NewRequirementRatio;

ProductionEfficiency : OPTIONAL NewRequirementRatio;

MaxC2H4eqEmissions : OPTIONAL NewRequirementAttribute;

MaxCO2eqEmissions : OPTIONAL NewRequirementAttribute;

MaxSO2eqEmissions : OPTIONAL NewRequirementAttribute;

END_ENTITY;

Name	Description
MinRenewable Materials	*Minimum percentage of renewable materials used in the project*
MaxNonRenewable Materials	*Maximum percentage of non-renewable materials used in the project*
ProductionEfficiency	*Target value for the production and distribution efficiency*
MaxC2H4eq Emissions	*Maximum C_2H_4eq emissions*
MaxCO2eq Emissions	*Maximum CO_2eq emissions*
MaxSO2eq Emissions	*Maximum SO_2eq emissions*

6.3.6.8 Cost Requirements

- Attribute set which defines targets for the different costs of the project (Figure 49)
- Direct link to *Design Model:* Project (IfcProject)
- Indirect links: None

ENTITY ***NewCostRequirements;***

SUBTYPE OF (*NewRequirement*);

InvestmentCosts : OPTIONAL NewRequirementCost;

SiteCosts : OPTIONAL NewRequirementCost;

DesignAndCMCosts : OPTIONAL NewRequirementCost;

ConstructionCosts : OPTIONAL NewRequirementCost;

OperationCosts : OPTIONAL NewRequirementCost;

MaintenanceCosts : OPTIONAL NewRequirementCost;

EnergyCosts : OPTIONAL NewRequirementCost;

DisposalCosts : OPTIONAL NewRequirementCost;

RecycleValue : OPTIONAL NewRequirementCost;

END_ENTITY;

Name	Description
InvestmentCosts	*Budgeted total investment cost*
SiteCosts	*Budgeted cost for the site acquisition*
DesignAndCMCosts	*Budgeted cost for the design and construction management*
ConstructionCosts	*Budgeted construction cost*
EnergyCosts	*Target value for the yearly energy costs*
OperationCosts	*Target value for the yearly operation costs*
MaintenanceCosts	*Target value for the yearly service and maintenance costs*
DisposalCosts	*Target value for the demolition costs*
RecycleValue	*Target value for the recyclable components and materials*

6.3.6.9 Accident and Catastrophe Risks

- Attribute set which defines *Requirements* for accident and natural catastrophe risks. The set identifies and/or limits possible risk factors related to the project location and planned activities. The accident risk description can include risks caused by the project environment or the project itself. Some of the issues can serve as basic information for design of structural and/or technical systems (Figure 52)
- Direct link to *Design Model:* Project (IfcProject)
- Indirect links: Building envelope, structural , HVAC, electrical, security and fire safety systems (IfcSystem – BuildingEnvelope, StructuralSystem, HvacSystem, ElectricalSystem, SecuritySystem, FireSafetySystem)

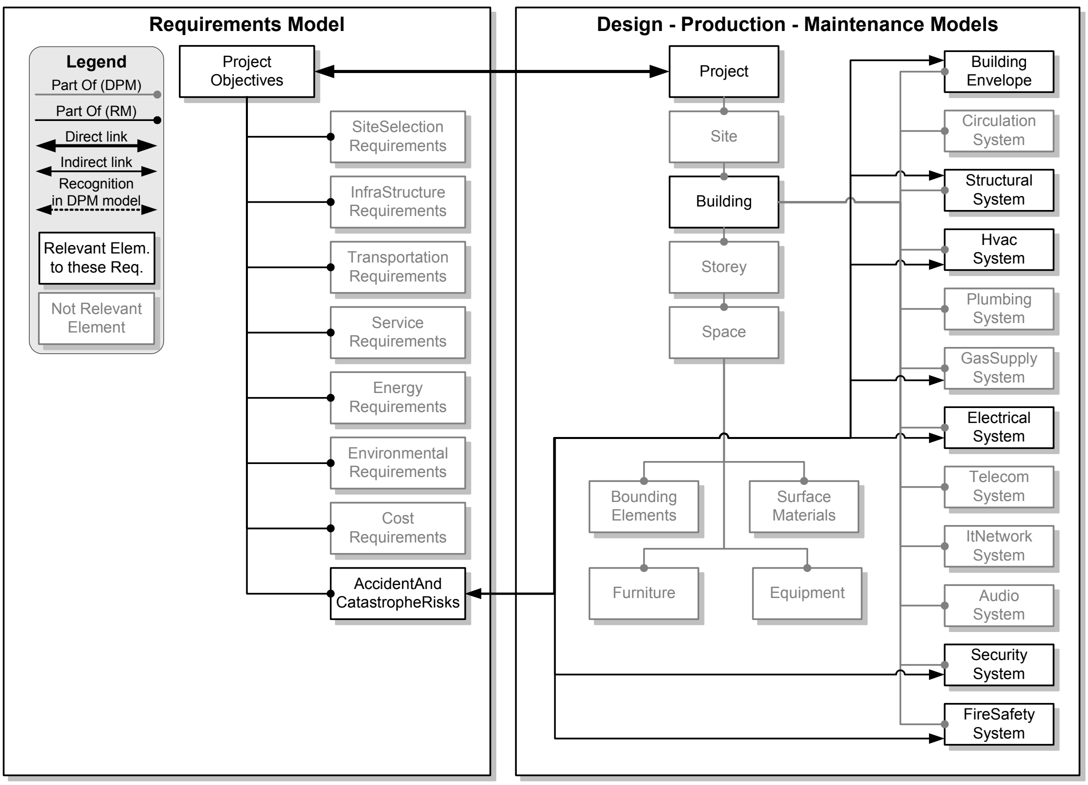

Figure 52: Project objectives 4/4 – Accident and Catastrophe Risks

ENTITY ***NewAccidentAndCatastropheRisks;***

SUBTYPE OF (*NewRequirement*);

AccidentRisks : OPTIONAL NewRequirementDescriptionList;

CatastropheRisks : OPTIONAL NewRequirementDescriptionList;

OtherRisks : OPTIONAL NewRequirementDescriptionList;

END_ENTITY;

Name	Description
AccidentRisks	*Description of different accident risk issues which might affect the site selection and/or design solutions, such as radiation accident and toxic substance leak; a list which can contain an unlimited number of IfcTexts*
CatastropheRisks	*Description of different catastrophe risk issues which might affect the site selection and/or design solutions, such as bush fire, earthquake, flood, storm, and volcanic activities; a list which can contain an unlimited number of IfcTexts*
OtherRisks	*Description of other identified risk issues which might affect the site selection and/or design solutions; a list which can contain an unlimited number of IfcTexts*

6.3.7 Requirements Object: Site Design Requirements

Attribute set which defines Requirements and limitations which relate to the site. Some of the attributes describe site limitations rather than *Requirements*, but they include important design information for the project. An example of a site *Requirement* is the minimum number of parking spaces, while site contamination can define limitations for the *Location* of the building or other uses of the site.

6.3.7.1 Site Design Requirements

- Main object to which the other site design *Requirements* and limitations are linked (Figure 53)
- Attribute set which defines *Requirements* for the design on the site
- Direct link to *Design Model:* Site (IfcSite)
- Indirect links: None

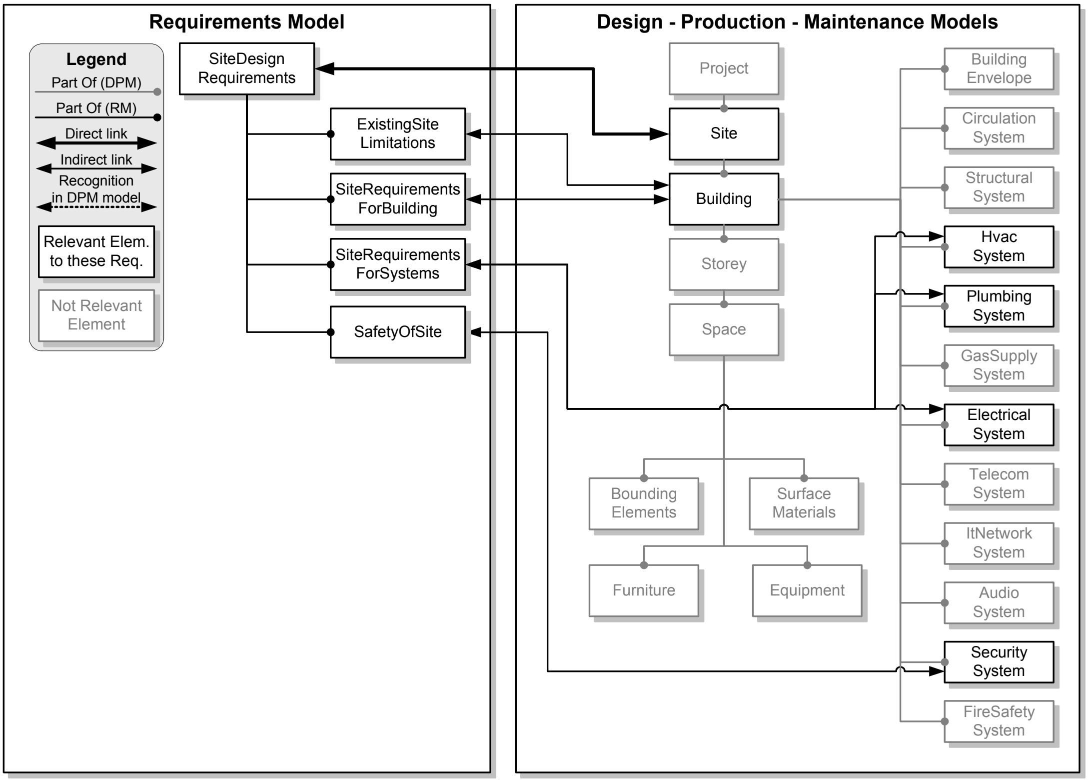

Figure 53: Site Design *Requirements*

ENTITY ***NewSiteDesignRequirements****;*

SUBTYPE OF (*NewRequirement*);

EmergencyVehicleAccess : OPTIONAL NewRequirementDescription;

VehicleAccess : OPTIONAL NewRequirementDescription;

SiteTrafficRequirements : OPTIONAL NewRequirementDescription;

MinCarParkingSpaces : OPTIONAL NewRequirementInteger;

MinBikeParkingSpaces: OPTIONAL NewRequirementInteger;

MinGreenSiteArea : OPTIONAL NewRequirementArea;

SiteAmenities : OPTIONAL NewRequirementDescriptionList;

ExistingSiteLimitations : OPTIONAL ***NewExistingSiteLimitations****;*

SiteRequirementsForBuilding : OPTIONAL ***NewSiteRequirementsForBuilding;***

SiteRequirementsForSystems : OPTIONAL ***NewSiteRequirementsForSystems;***

END_ENTITY;

Name	Description
EmergencyVehicle Access	*Description of the required emergency vehicle access on the site*
VehicleAccess	*Description of the required vehicle access on the site, such as delivery and customer traffic*
SiteTraffic Requirements	*Description of the traffic requirements on the site, for example, separation of pedestrian and vehicle traffic and speed limits*
MinCarParking Spaces	*Required minimum number of car parking spaces on the site*
MinBikeParking Spaces	*Required minimum number of bike parking spaces on the site*
MinGreenSiteArea	*Required minimum green area on the site*
SiteAmenities	*Required site amenities and accessories; a list which can contain an unlimited number of IfcTexts*

6.3.7.2 Existing Site Limitations

- Attribute set which defines the existing limitations on the site. Some limitations have effect to the site and building design (Figure 53)
- Direct link to *Design Model:* Site (IfcSite)
- Indirect links: Building (IfcBuilding)

ENTITY ***NewExistingSiteLimitations****;*

SUBTYPE OF (*NewRequirement*);

CommunityReference : OPTIONAL IfcDocumentReference;

CommunityRequirements : OPTIONAL NewRequirementDescriptionList;

CulturalValue : OPTIONAL NewRequirementDescription;

EcologicalSignificance : OPTIONAL NewRequirementDescription;

FaunaEffects : OPTIONAL NewRequirementDescriptionList;

ExistingBuildings : OPTIONAL NewRequirementDescriptionList;

RelatedBuildings : OPTIONAL NewRequirementDescriptionList;

BuildingsToPreserve : OPTIONAL NewRequirementDescriptionList;

BuildingsToDemolish : OPTIONAL NewRequirementDescriptionList;

ExistingVegetation : OPTIONAL NewRequirementDescriptionList;

PreservedVegetation : OPTIONAL NewRequirementDescriptionList;

SiteNoiseLevel : OPTIONAL NewRequirementSound;

SiteContamination : OPTIONAL NewRequirementDescription;

StormWater : OPTIONAL NewRequirementDescription;

END_ENTITY;

Name	Description
Community Reference	*References to requirements documents of the community related to the site*
Community Requirements	*Description of the community requirements related to the site; a list which can contain an unlimited number of IfcTexts*
CulturalValue	*Description of the cultural, historical or recreational value of the site which might be relevant for the design*
Ecological Significance	*Description of the ecological significance and uniqueness of the site which might be relevant for the design*
FaunaEffects	*Description of the limitations, how the building is allowed to effect to the fauna; a list which can contain an unlimited number of IfcTexts*
ExistingBuildings	*Description of the existing buildings; a list which can contain an unlimited number of IfcTexts*
RelatedBuildings	*Description of the existing buildings which will have related activities; a list which can contain an unlimited number of IfcTexts*
BuildingsToPreserve	*Description of the existing buildings which must be preserved; a list which can contain an unlimited number of IfcTexts*
BuildingsToDemolish	*Description of the existing buildings which can or must be demolished; a list which can contain an unlimited number of IfcTexts*
ExistingVegetation	*Description of the existing vegetation; quantity, condition, and extent; a list which can contain an unlimited number of IfcTexts*
PreservedVegetation	*Description of the existing vegetation which must be preserved; a list which can contain an unlimited number of IfcTexts*
SiteNoiseLevel	*Existing noise level on the site caused for example by traffic, airplanes, neighbors, etc.*
SiteContamination	*Description of the site contamination which might affect the excavation, site and slab structures, etc.*
StormWater	*Description of possible storm water problems and limitations on the site*

6.3.7.3 Site Requirements for Building

- Attribute set which defines *Requirements* for the design of the building related to the site (Figure 53)
- Direct link to *Design Model:* Site (IfcSite)
- Indirect links: Building (IfcBuilding)

ENTITY ***NewSiteRequirementsForBuilding;***

SUBTYPE OF (*NewRequirement*);

PermittedBuildingArea : OPTIONAL NewRequirementArea;

PermittedBuildingVolume : OPTIONAL NewRequirementVolume;

PermittedBuildingFootPrint : OPTIONAL NewRequirementArea;

PermittedBuildingLocation : OPTIONAL NewRequirementDescription;

PermittedBuildingHeight : OPTIONAL NewRequirementDistance;

PermittedNumberOfFloors : OPTIONAL NewRequirementInteger;

SurfaceGlare : OPTIONAL NewRequirementDescription;

ShadingEffects : OPTIONAL NewRequirementDescription;

WindEffects : OPTIONAL NewRequirementDescription;

MaxOutdoorNoise : OPTIONAL NewRequirementSound;

MaxOdorEmissions : OPTIONAL NewRequirementDescription;

MaxHeatEmissions : OPTIONAL NewRequirementPower;

MaxNoiseEmissions : OPTIONAL NewRequirementSound;

END_ENTITY;

Name	**Description**
PermittedBuilding Area	*Permitted maximum building area*
PermittedBuilding Volume	*Permitted maximum building volume*
PermittedBuilding Footprint	*Permitted maximum building footprint size*
PermittedBuilding Location	*Description of the permitted building location*
PermittedBuilding Height	*Permitted maximum height of the building*
PermittedNumberOf Floors	*Permitted maximum number of floors*
SurfaceGlare	*Permitted glare of the building surfaces*
ShadingEffects	*Permitted shading effects of the building*
WindEffects	*Permitted wind effects of the building*
MaxOutdoorNoise	*Permitted maximum noise level on the site including the noise from building and environment*
MaxOdorEmissions	*Permitted maximum odor emissions of the building*
MaxHeatEmissions	*Permitted maximum heat emissions of the building*
MaxNoiseEmissions	*Permitted maximum noise emissions of the building*

6.3.7.4 Site Requirements for Systems

- Attribute set which defines *Requirements* for the design of technical systems on the site (Figure 53)
- Direct link to *Design Model:* Site (IfcSite)
- Indirect links: Electrical, HVAC, and plumbing systems (IfcSystem – ElectricalSystem, HvacSystem, PlumbingSystem)

*ENTITY **NewSiteRequirementsForSystems**;*

SUBTYPE OF (*NewRequirement*);

OutdoorAreaComfort : OPTIONAL NewRequirementDescription;

SiteLighting : OPTIONAL NewRequirementDescription;

SiteHeating : OPTIONAL NewRequirementDescription;

SiteDrainage : OPTIONAL NewRequirementDescription;

END_ENTITY;

Name	**Description**
OutdoorAreaComfort	*Description of the required outdoor area comfort, usability and amenities*
SiteLighting	*Description of the site lighting requirements*
SiteHeating	*Description of the site heating requirements*
SiteDrainage	*Description of the required site drainage*

6.3.7.5 Safety of the Site

- Attribute set which defines *Requirements* for security of the site (Figure 53)
- Direct link to *Design Model:* Site (Ifcsite)
- Indirect links: Security system (IfcSystem – SecuritySystem)

*ENTITY **NewSafetyOfSite**;*

SUBTYPE OF (*NewRequirement*);

SiteSecurity : OPTIONAL NewRequirementDescription;

MonitoringOfSite : OPTIONAL NewRequirementDescription;

PerimeterControl : OPTIONAL NewRequirementDescription;

ProtectionFromAttack : OPTIONAL NewRequirementDescription;

ControlOfParking : OPTIONAL NewRequirementDescription;

ProtectionOfVehicles : OPTIONAL NewRequirementDescription;

END_ENTITY;

Name	Description
SiteSecurity	*Description of the security requirements for the site*
MonitoringOfSite	*Description of the security monitoring requirements for the site*
PerimeterControl	*Description of the site perimeter control requirements*
ProtectionFrom Attack	*Description of the attack protection requirements*
ControlOfParking	*Description of the parking area control requirements*
ProtectionOfVehicles	*Description of the parking area protection requirements*

6.3.8 Building Requirements

6.3.8.1 Building Requirements

- Main object to which the other building *Requirements* are linked (Figure 54)
- Attribute set which defines general *Requirements* for the building
- Direct link to *Design Model:* Building (IfcBuilding)
- Indirect links: None

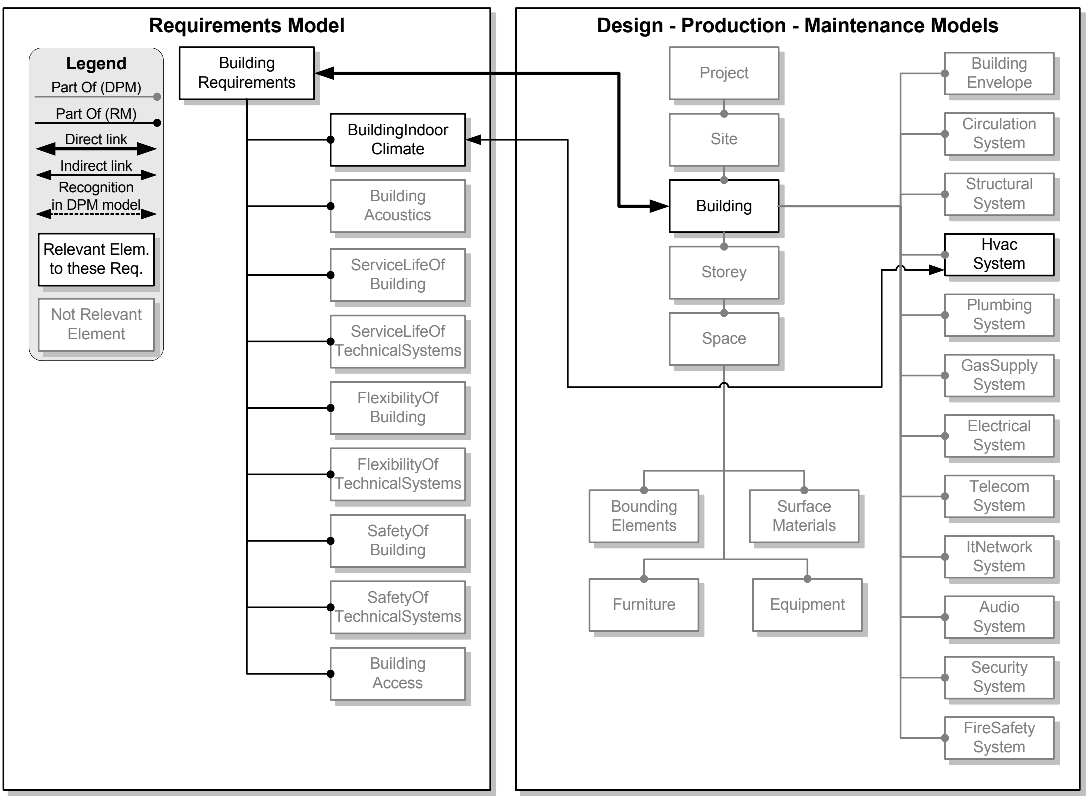

Figure 54: Building *Requirements* 1/9 – Indoor climate

*ENTITY **NewBuildingRequirements**;*

SUBTYPE OF (*NewRequirement*);

TotalBuildingVolume : OPTIONAL NewRequirementVolume;

TotalBuildingArea : OPTIONAL NewRequirementArea;

TotalProgramArea : OPTIONAL NewRequirementArea;

AestheticAppearance : OPTIONAL NewRequirementDescriptionList;

WayFinding : OPTIONAL NewRequirementDescriptionList;

*BuildingIndoorAirQuality : OPTIONAL **NewBuildingIndoorClimate**;*

*BuildingAcoustics : OPTIONAL **NewBuildingAcoustics**;*

*BuildingLighting : OPTIONAL **NewBuildingLighting**;*

*ServiceLifeOfBuilding : OPTIONAL **NewServiceLifeOfBuilding**;*

*ServiceLifeOfTechnicalSystems : OPTIONAL **NewServiceLifeOfTechnicalSystems**;*

*FlexibilityOfBuilding : OPTIONAL **NewFlexibilityOfBuilding**;*

*FlexibilityOfTechnicalSystems : OPTIONAL **NewFlexibilityOfTechnicalSystems**;*

*SafetyOfBuilding : OPTIONAL **NewSafetyOfBuilding**;*

*SafetyOfTechnicalSystems : OPTIONAL **NewSafetyOfTechnicalSystems**;*

*BuildingAccess : OPTIONAL **NewBuildingAccess**;*

END_ENTITY;

Name	**Description**
TotalBuildingVolume	*Maximum total volume of the building; design target*
TotalBuildingArea	*Maximum total area of the building; design target*
TotalProgramArea	*Maximum total program area; the value should be the same as the sum of all areas in the space program*
AestheticAppearence	*Description of the aesthetic requirements for the building; a list which can contain an unlimited number of IfcTexts*
WayFinding	*Description of the way-finding requirements for the building; a list which can contain an unlimited number of IfcTexts*

6.3.8.2 Building Indoor Climate

- Attribute set which defines the indoor air quality *Requirements* for the building (Figure 54)
- Direct link to *Design Model:* Building (IfcBuilding)
- Indirect links: HVAC system (IfcSystem – HvacSystem)

ENTITY ***NewBuildingIndoorClimate;***

SUBTYPE OF (*NewRequirement*);

MaxNH3 : OPTIONAL NewRequirementReal;

MaxCO2 : OPTIONAL NewRequirementReal;

MaxCO : OPTIONAL NewRequirementReal;

MaxH2CO : OPTIONAL NewRequirementReal;

MaxO3 : OPTIONAL NewRequirementReal;

MaxTVOC : OPTIONAL NewRequirementReal;

MaxRadon : OPTIONAL NewRequirementReal;

MaxOdorIntensity : OPTIONAL NewRequirementInteger;

MaxMicrobes : OPTIONAL NewRequirementReal;

MaxParticles : OPTIONAL NewRequirementReal;

NaturallyVentilated : OPTIONAL NewRequirementDescription;

END_ENTITY;

Name	**Description**
MaxNH3	*Allowable maximum level of ammonia/amines (NH_3) in the indoor air*
MaxCO2	*Allowable maximum level of carbon dioxide (CO_2) in the indoor air*
MaxCO	*Allowable maximum level of carbon monoxide (CO) in the indoor air*
MaxH2CO	*Allowable maximum level of formaldehyde (H_2CO) in the indoor air*
MaxO3	*Allowable maximum level of ozone (O_3) in the indoor air*
MaxTVOC	*Allowable maximum level of volatile organic compounds(TVOC) in the indoor air*
MaxRadon	*Allowable maximum level of radon in the indoor air*
MaxOdorIntensity	*Allowable maximum odor intensity (intensity scale)*
MaxMicrobes	*Allowable maximum level of microbes in the indoor air*
MaxParticles	*Allowable maximum level of airborne particles in the indoor air*
NaturallyVentilated	*Description of required natural ventilation system*

6.3.8.3 Building Acoustics

- Attribute set which defines acoustical *Requirements* for the building (Figure 55)
- Direct link to *Design Model:* Building (IfcBuilding)
- Indirect links: Structural and audio system and building envelope (IfcSystem – StructuralSystem, AudioSystem, BuildingEnvelope)

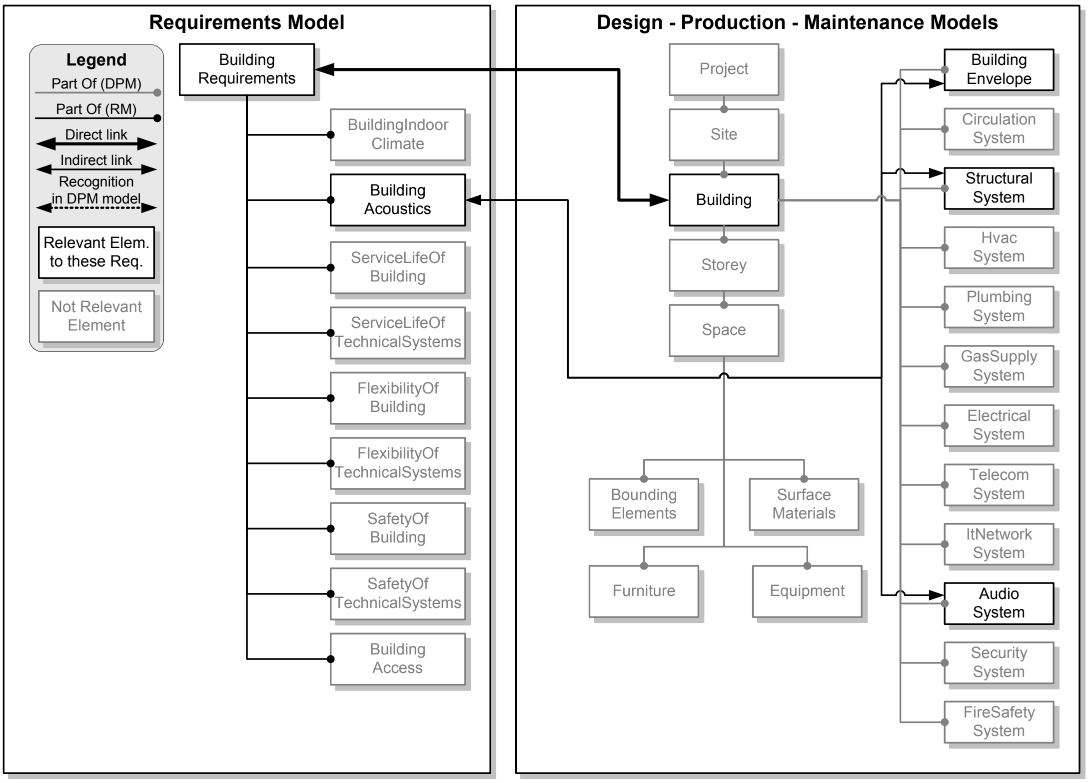

Figure 55: Building *Requirements* 2/9 – Acoustics

ENTITY ***NewBuildingAcoustics****;*

SUBTYPE OF (*NewRequirement*);

MinImpactSoundInsulation : OPTIONAL NewRequirementSound;

MinUnitSoundInsulation : OPTIONAL NewRequirementSound;

AudioSystem : OPTIONAL NewRequirementDescriptionList;

END_ENTITY;

Name	**Description**
MinImpactSound Insulation	*Required minimum impact sound insulation for floor structures in the building*
MinUnitSound Insulation	*Required minimum sound insulation between apartments or other functional units in the building (c.f. Space Acoustics)*
AudioSystem	*Description of the building audio system requirements; a list which can contain an unlimited number of IfcTexts*

6.3.8.4 Service Life of Building

- Attribute set which defines service life expectations to the building and its main components (Figure 56)
- Direct link to *Design Model:* Building (IfcBuilding)
- Indirect links: Structural system and building envelope (IfcSystem – StructuralSystem, BuildingEnvelope)

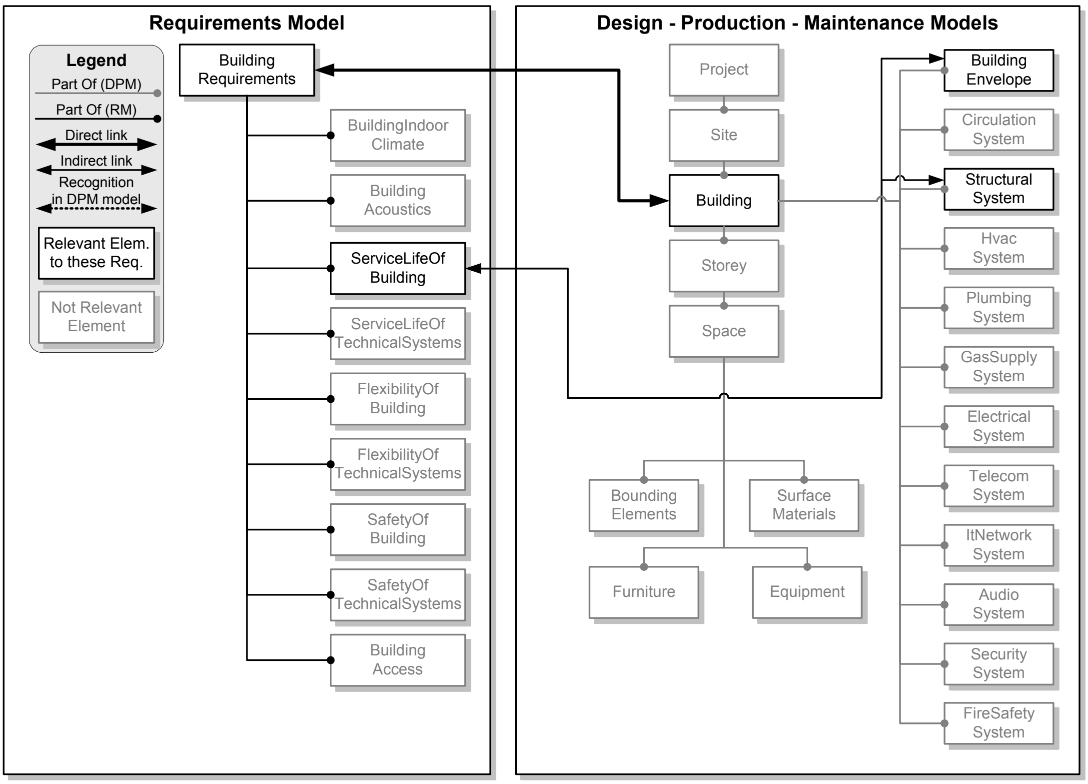

Figure 56: Building *Requirements* 3/9 – Building service life

ENTITY ***NewServiceLifeOfBuilding****;*

SUBTYPE OF (*NewRequirement*);

BuildingServiceLife : OPTIONAL NewRequirementReal;

StructureServiceLife : OPTIONAL NewRequirementReal;

EnvelopeServiceLife : OPTIONAL NewRequirementReal;

END_ENTITY;

Name	Description
BuildingServiceLife	*Expected building service life in years*
StructureServiceLife	*Expected service life for the structural system*
EnvelopeServiceLife	*Expected service life of major elements of the building envelope, such as cladding, windows, and external doors.*

6.3.8.5 Service Life of Technical Systems

- Attribute set which defines service life expectations to the technical systems of the building (Figure 57)
- Direct link to *Design Model:* Building (IfcBuilding)
- Indirect links: All technical systems (IfcSystem – AudioSystem, CirculationSystem, ElectricalSystem, FireSafetySystem, GasSupplySystem, HvacSystem, ItNetworkSystem, PlumbingSystem, SecuritySystem, TelecomSystem)

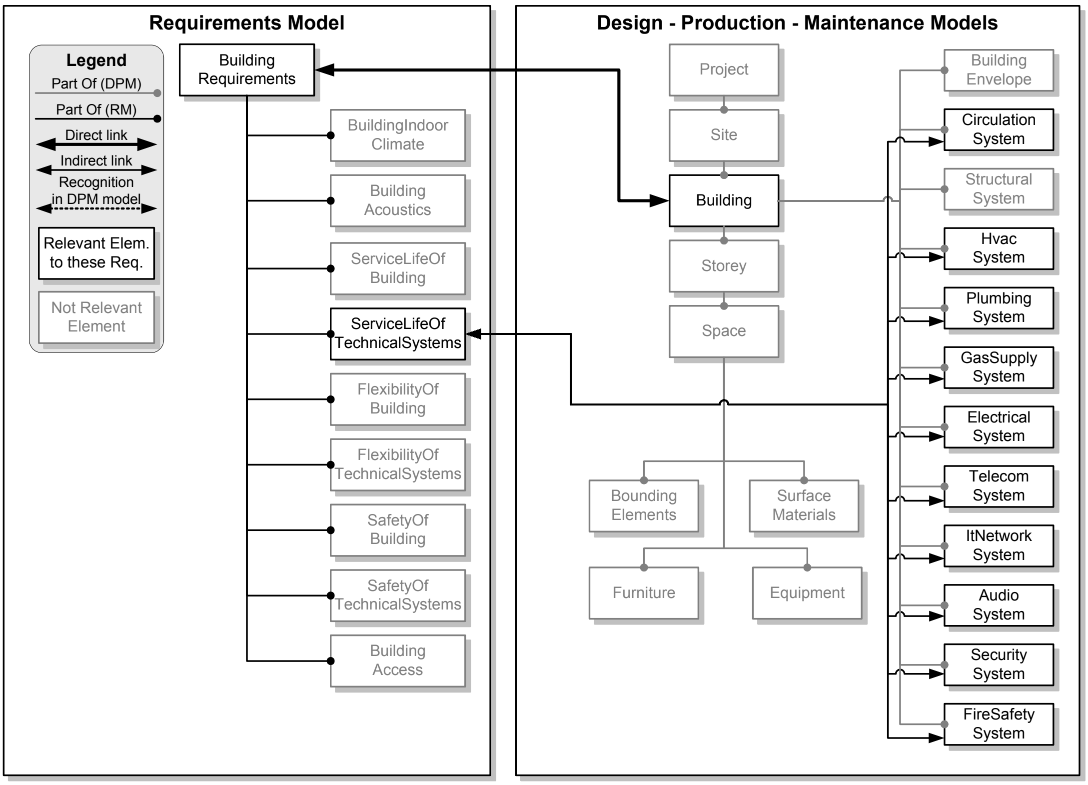

Figure 57: Building *Requirements* 4/9 – Service life of technical systems

ENTITY ***NewServiceLifeOfTechicalSystems****;*

SUBTYPE OF (*NewRequirement*);

ElevatorServiceLife : OPTIONAL NewRequirementReal;

EscalatorServiceLife : OPTIONAL NewRequirementReal;

HeatMachineryServiceLife : OPTIONAL NewRequirementReal;

HeatingDistributionSystemServiceLife : OPTIONAL NewRequirementReal;

RadiatorServiceLife : OPTIONAL NewRequirementReal;

PumpAndFanServiceLife : OPTIONAL NewRequirementReal;

AutomationControlsServiceLife : OPTIONAL NewRequirementReal;

AutomationCableServiceLife : OPTIONAL NewRequirementReal;

DuctServiceLife : OPTIONAL NewRequirementReal;
VisiblePipingServiceLife : OPTIONAL NewRequirementReal;
NonVisiblePipingServiceLife : OPTIONAL NewRequirementReal;
SewerSystemServiceLife : OPTIONAL NewRequirementReal;
WaterSystemServiceLife : OPTIONAL NewRequirementReal;
PlumbingSystemServiceLife : OPTIONAL NewRequirementReal;
GasSystemServiceLife : OPTIONAL NewRequirementReal;
ElectricalCableServiceLife : OPTIONAL NewRequirementReal;
ElectricalFittingsServiceLife : OPTIONAL NewRequirementReal;
LightSourceServiceLife : OPTIONAL NewRequirementReal;
ItCableServiceLife : OPTIONAL NewRequirementReal;
TelecomCableServiceLife : OPTIONAL NewRequirementReal;
AudioSystemServiceLife : OPTIONAL NewRequirementReal;
FireSafetySystemServiceLife : OPTIONAL NewRequirementReal;
SecuritySystemServiceLife : OPTIONAL NewRequirementReal;

END_ENTITY;

Name	Description
ElevatorServiceLife	*Expected service life of elevator system*
EscalatorServiceLife	*Expected service life of escalator system*
HeatMachinery ServiceLife	*Expected service life for the heat yield machinery, such as heat transfer casing and boilers, accumulators, and oil tanks*
HeatingDistribution SystemServiceLife	*Expected service life of water circulation heat distribution system (steel pipes and batteries)*
RadiatorServiceLife	*Expected service life for the heating and cooling radiators*
PumpAndFan ServiceLife	*Expected service life for the HVAC pumps and fans*
AutomationControls ServiceLife	*Expected service life for HVAC automation control and setting devices*
AutomationCable ServiceLife	*Expected service life for HVAC and building automation cabling*
DuctServiceLife	*Expected service life for ventilation and air conditioning ducts*
VisiblePiping ServiceLife	*Expected service life for visible piping*
NonVisiblePiping ServiceLife	*Expected service life for non-visible piping (inside or behind structures)*
SewerSystem ServiceLife	*Expected service life for sewer system*
WaterSystem ServiceLife	*Expected service life for water and sewer system components, such as wash basins, WC-seats, and bath tubs*
PlumbingSystem ServiceLife	*Expected service life for plumbing system components, such as sealing and control valves, and mixers*

GasSystem ServiceLife	*Expected service life for gas supply system*
ElectricalCable ServiceLife	*Expected service life for electrical cabling*
ElectricalFittings ServiceLife	*Expected service life for electrical fittings, such as light fittings, outlets, and switches*
LightSource ServiceLife	*Expected average service life for the light sources (lamps)*
ItCableServiceLife	*Expected service life for IT cabling*
TelecomCable ServiceLife	*Expected service life for telecommunication cabling*
AudioSystemService Life	*Expected service life for the main components of audio system*
FireSafetySystem ServiceLife	*Expected service life for the main components of fire safety system*
SecuritySystem ServiceLife	*Expected service life for the main components of security system*

6.3.8.6 Flexibility of Building

- Attribute set which defines the flexibility *Requirements* for a building and its main components (Figure 58)
- Direct link to *Design Model:* Building (IfcBuilding)
- Indirect links: Structural system and building envelope (IfcSystem – StructuralSystem, BuildingEnvelope)

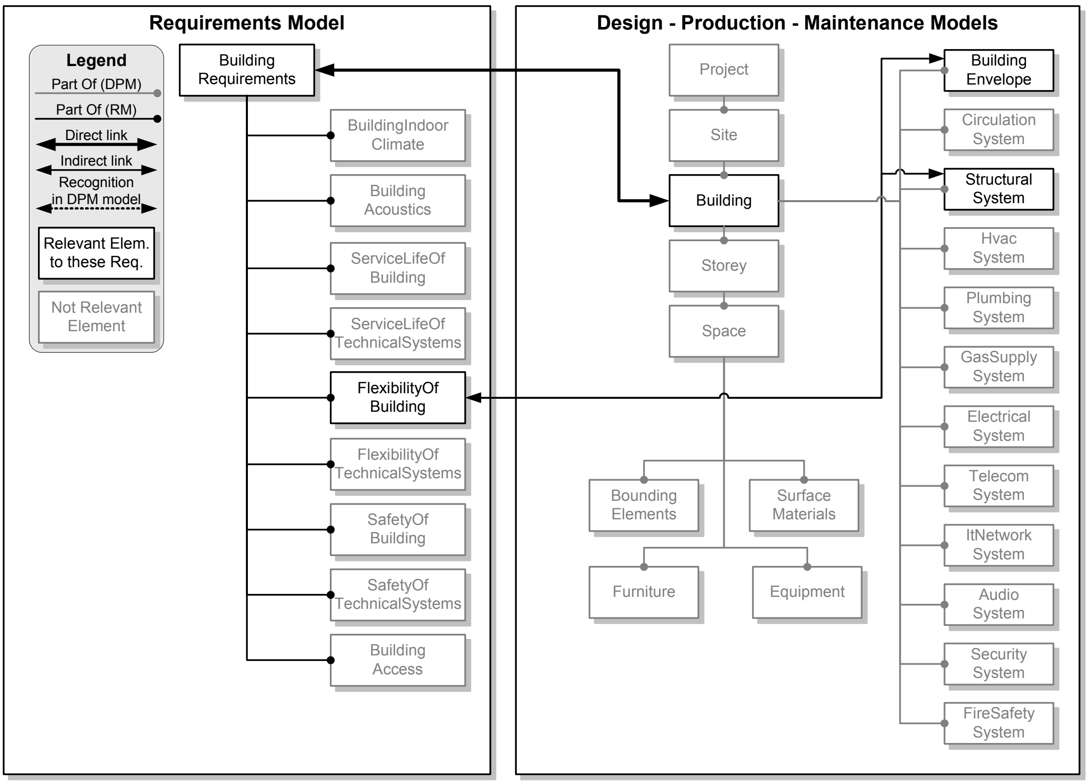

Figure 58: Building *Requirements* 5/9 – Flexibility of building

*ENTITY **NewFlexibilityOfBuilding**;*

SUBTYPE OF (*NewRequirement*);

Expandability : OPTIONAL NewRequirementDescription;

BuildingFlexibility : OPTIONAL NewRequirementDescription;

FrameFlexibility : OPTIONAL NewRequirementDescription;

FloorFlexibility : OPTIONAL NewRequirementDescription;

EnvelopeFlexibility : OPTIONAL NewRequirementDescription;

PartitionFlexibility : OPTIONAL NewRequirementDescription;

DesignFlexibility : OPTIONAL NewRequirementDescription;

OccupancyFlexibility : OPTIONAL NewRequirementDescription;

END_ENTITY;

Name	Description
Expandability	*Description of the expandability requirements for the building*
BuildingFlexibility	*Description of the requirements for changes in the use of the building afterwards*
FrameFlexibility	*Description of the flexibility requirements for the structural frame of the building*
FloorFlexibility	*Description of the flexibility requirements for the floor structures of the building*
EnvelopeFlexibility	*Description of the flexibility requirements for the building envelope*
PartitionFlexibility	*Description of the flexibility requirements for the partition walls*
DesignFlexibility	*Description of the requirements for individual choices by the initial users during the design phase*
OccupancyFlexibility	*Description of the requirements for individual choices by the users after building is completed*

6.3.8.7 Flexibility of Technical Systems

- Attribute set which defines the flexibility *Requirements* for the technical systems (Figure 59)
- Direct link to *Design Model:* Building (IfcBuilding)
- Indirect links: All technical systems (IfcSystem – AudioSystem, CirculationSystem, ElectricalSystem, FireSafetySystem, GasSupplySystem, HvacSystem, ItNetworkSystem, PlumbingSystem, SecuritySystem, TelecomSystem)

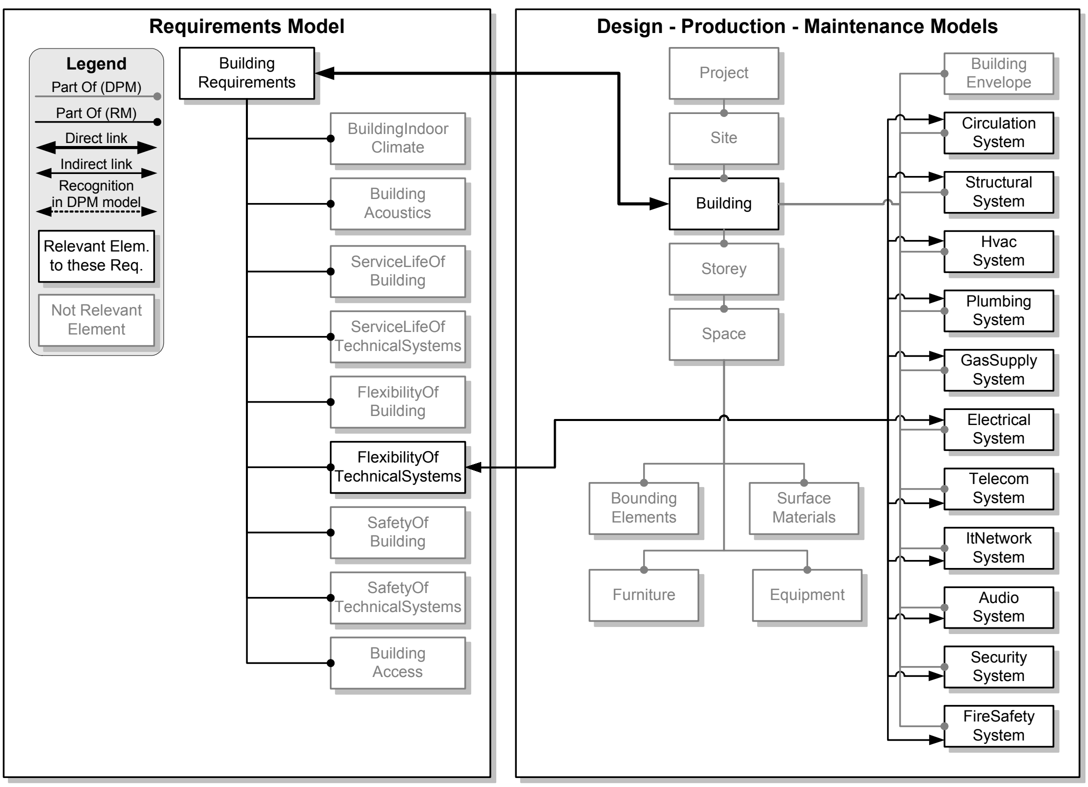

Figure 59: Building *Requirements* 6/9 – Flexibility of technical systems

```
ENTITY NewFlexibilityOfTechnicalSystems;
    SUBTYPE OF (NewRequirement);
        ElevatorFlexibility : OPTIONAL NewRequirementDescription;
        EscalatorFlexibility : OPTIONAL NewRequirementDescription;
        HorizontalFlexibility : OPTIONAL NewRequirementDescription;
        VerticalFlexibility : OPTIONAL NewRequirementDescription;
        BuildingAutomationFlexibility : OPTIONAL NewRequirementDescription;
        HeatingSystemFlexibility : OPTIONAL NewRequirementDescription;
        HvacSystemFlexibility : OPTIONAL NewRequirementDescription;
        SprinklerFlexibility : OPTIONAL NewRequirementDescription;
        WaterSupplyFlexibility : OPTIONAL NewRequirementDescription;
        GasSupplyFlexibility : OPTIONAL NewRequirementDescription;
        ElectricalSystemFlexibility : OPTIONAL NewRequirementDescription;
        ElectricalInstallationFlexibility : OPTIONAL NewRequirementDescription;
        IlluminationFlexibility : OPTIONAL NewRequirementDescription;
        ItNetworkFlexibility : OPTIONAL NewRequirementDescription;
        TelecomSystemFlexibility : OPTIONAL NewRequirementDescription;
        AudioSystemFlexibility : OPTIONAL NewRequirementDescription;
        FireSafetySystemFlexibility : OPTIONAL NewRequirementDescription;
        SecuritySystemFlexibility : OPTIONAL NewRequirementDescription;
END_ENTITY;
```

Name	**Description**
ElevatorFlexibility	*Description of the flexibility requirements for the elevators*
EscalatorFlexibility	*Description of the flexibility requirements for the escalators*
HorizontalFlexibility	*Description of the flexibility requirements for the horizontal installations*
VerticalFlexibility	*Description of the flexibility requirements for the vertical shafts*
BuildingAutomation Flexibility	*Description of the flexibility requirements for the building automation systems*
HeatingSystem Flexibility	*Description of the flexibility requirements for the heating system*
HvacSystem Flexibility	*Description of the flexibility requirements for the ventilation and cooling system*
SprinklerFlexibility	*Description of the flexibility requirements for the sprinkler system*
WaterSupply Flexibility	*Description of the flexibility requirements for the water supply system*
GasSupplyFlexibility	*Description of the flexibility requirements for the gas supply system*
ElectricalSystem Flexibility	*Description of the flexibility requirements for the main electrical distribution system*

ElectricalInstallation Flexibility	*Description of the flexibility requirements for the electrical installations on space level*
IlluminationFlexibility	*Description of the flexibility requirements for the illumination system*
ItNetworkFlexibility	*Description of the flexibility requirements for the IT network*
TelecomSystem Flexibility	*Description of the flexibility requirements for the telecommunications system*
AudioSystem Flexibility	*Description of the flexibility requirements for the audio system*
FireSafetySystem Flexibility	*Description of the flexibility requirements for the fire safety system*
SecuritySystem Flexibility	*Description of the flexibility requirements for the security and access control system*

6.3.8.8 Safety of Building

- Attribute set which defines safety and security *Requirements* for the building (Figure 60)
- Direct link to *Design Model:* Building (IfcBuilding)
- Indirect links: Envelope, structural, HVAC, security and fire safety systems (IfcSystem – StructuralSystem, HvacSystem, SecuritySystem, FireSafetySystem)

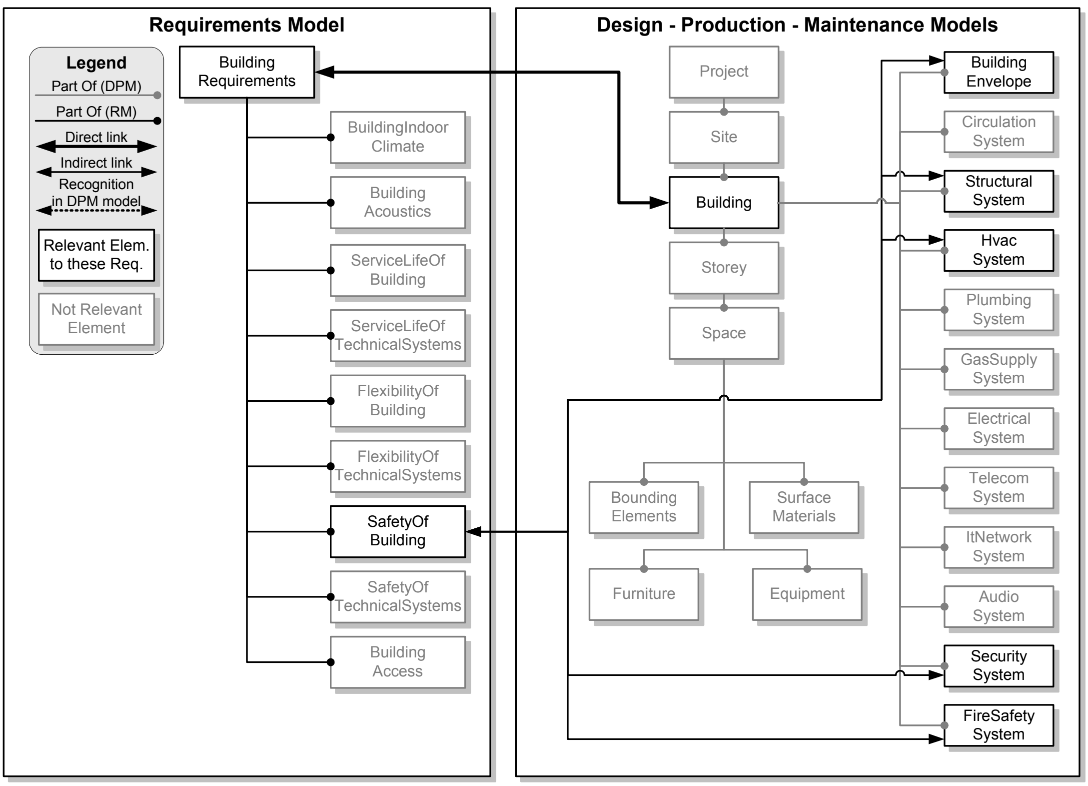

Figure 60: Building *Requirements* 7/9 – Safety of building

```
ENTITY NewSafetyOfBuilding;
	SUBTYPE OF (NewRequirement);
		BuildingAccessControl  :  OPTIONAL NewRequirementDescription;
		SeparationOfZones  :  OPTIONAL NewRequirementDescription;
		LoadCapacity   :   OPTIONAL NewRequirementDescription;
		FireSafetySystem   :   OPTIONAL NewRequirementDescription;
		FireResistanceRating   :   OPTIONAL NewRequirementDescription;
		FireResistanceTime   :   OPTIONAL NewRequirementReal;
		SurfaceFirePropagation   :   OPTIONAL NewRequirementDescription;
		SurfaceInflammability   :   OPTIONAL NewRequirementDescription;
		FireRatingForFittings  :  OPTIONAL NewRequirementDescription;
		BuildingSecurity   :   OPTIONAL NewRequirementDescription;
		AirIntakeLocation  :  OPTIONAL NewRequirementDescription;
END_ENTITY;
```

Name	Description
BuildingAccess Control	*Description of the general access control requirements for the circulation systems in the building*
SeparationOfZones	*Description of the zone separation requirements*
LoadCapacity	*Description of the general load capacity requirements for the building. Space-specific requirements are part of spatial requirements*
FireSafetySystem	*Description of requirements for the fire safety and sprinkler systems*
FireResistanceRating	*Required fire-resistance rating*
FireResistanceTime	*Required fire-resistance time*
SurfaceFire Propagation	*Required surface layer fire-propagation rating*
SurfaceInflammability	*Required surface layer inflammability rating*
FireRatingForFittings	*Required fire-rating for fittings and furniture*
BuildingSecurity	*Description of requirements for the building security systems and other security requirements*
AirIntakeLocation	*Description of the safety requirements for air intake location*

6.3.8.9 Safety of Technical Systems

- Attribute set which defines safety and security *Requirements* for the technical systems (Figure 61)
- Direct link to *Design Model:* Building (IfcBuilding)
- Indirect links: All technical systems (IfcSystem – AudioSystem, CirculationSystem, ElectricalSystem, FireSafetySystem, GasSupplySystem, HvacSystem, ItNetworkSystem, PlumbingSystem, SecuritySystem, TelecomSystem)

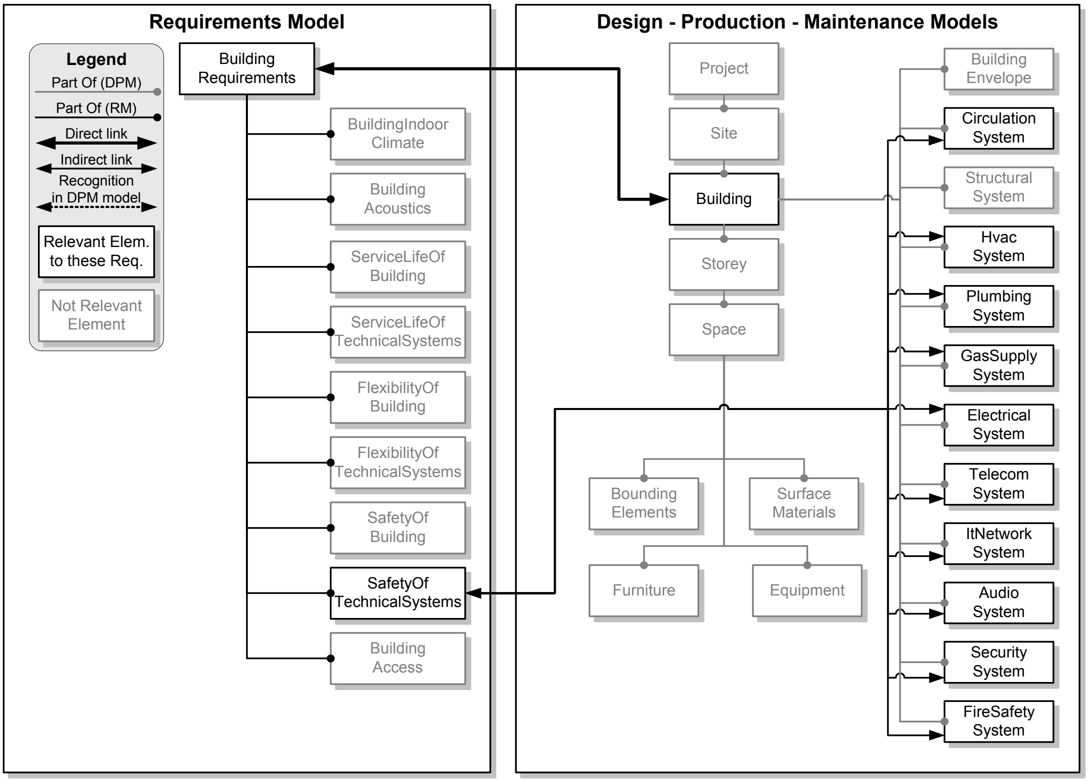

Figure 61: Building *Requirements* 8/9 – Safety of technical systems

ENTITY ***NewSafetyOfTechnicalSystems****;*

SUBTYPE OF (*NewRequirement*);

ElevatorReliability : OPTIONAL NewRequirementRatio;

EscalatorReliability : OPTIONAL NewRequirementRatio;

HvacReliability : OPTIONAL NewRequirementRatio;

SewerFloodingPrevention : OPTIONAL NewRequirementDescription;

GasSupplyReliability : OPTIONAL NewRequirementRatio;

ElectricalReliability : OPTIONAL NewRequirementRatio;

ElectricalBackupSystem : OPTIONAL NewRequirementDescription;

TelecomReliability : OPTIONAL NewRequirementRatio;

TelecomBackupTime : OPTIONAL NewRequirementReal;

ItNetworkReliability : OPTIONAL NewRequirementRatio;

ItNetworkBackupTime : OPTIONAL NewRequirementReal;

ItNetworkSecurity : OPTIONAL NewRequirementDescription;

AudioSystemReliability : OPTIONAL NewRequirementRatio;

SecuritySystemReliability : OPTIONAL NewRequirementRatio;

FireSafetySystemReliability : OPTIONAL NewRequirementRatio;

END_ENTITY;

Name	Description
ElevatorReliability	*Reliability/availability requirements for the elevator system, typically % of the capacity*
EscalatorReliability	*Reliability/availability requirements for the escalator system, typically % of the time*
HvacReliability	*Reliability/availability requirements for the HVAC systems, typically % of the capacity*
SewerFlooding Prevention	*Description of the required sewer flooding prevention system*
GasSupplyReliability	*Reliability/availability requirements for the gas supply systems, typically % of the time*
ElectricalReliability	*Reliability/availability requirements for the electrical systems, typically % of the time*
ElectricalBackup System	*Description of the required electricity backup system*
TelecomReliability	*Reliability/availability requirements for the telecommunication systems, typically % of the time*
TelecomBackupTime	*Minimum required backup time for telecommunication systems in electricity failure situations*
ItNetworkReliability	*Reliability/availability requirements for the IT network, % of the time*
ItNetwork BackupTime	*Minimum required backup time for IT network in electricity failure situations*
ItNetworkSecurity	*Description of the security requirements for the IT network*
AudioSystem Reliability	*Reliability/availability requirements for the audio systems, % of the time*
SecuritySystem Reliability	*Reliability/availability requirements for the security systems, % of the time*
FireSafetySystem Reliability	*Reliability/availability requirements for the fire safety systems, % of the time*

6.3.8.10 Building Accessibility

- Attribute set which defines accessibility *Requirements* for the building (Figure 62)
- Direct link to *Design Model:* Building (IfcBuilding)
- Indirect links: Circulation and audio systems (IfcSystem – CirculationSystem, AudioSystem)

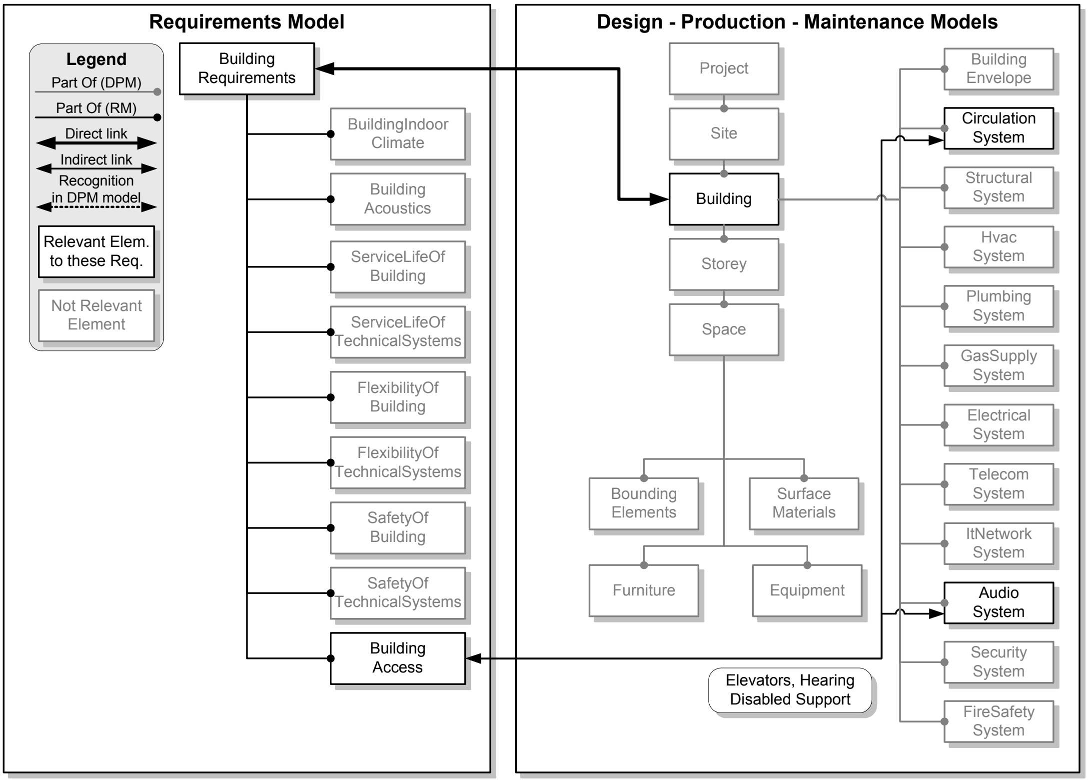

Figure 62: Building *Requirements* 9/9 – Building accessibility

ENTITY ***NewBuildingAccessibility****;*

SUBTYPE OF (*NewRequirement*);

ElevatorRequirements : OPTIONAL NewRequirementDescription;

AccessibilityForHandicapped : OPTIONAL NewRequirementDescriptionList;

AccessibilityForHearingImpared : OPTIONAL NewRequirementDescriptionList;

AccessibilityForSightDisabled : OPTIONAL NewRequirementDescriptionList;

END_ENTITY;

Name	**Description**
Elevator Requirements	*Description of the elevator requirements*
AccessibilityFor Handicapped	*Description of the accessibility requirements for handicapped people; a list which can contain an unlimited number of IfcTexts*
AccessibilityFor HearingImpaired	*Description of the accessibility requirements for hearing impaired people; a list which can contain an unlimited number of IfcTexts*
AccessibilityFor SightDisabled	*Description of the accessibility requirements for sight disabled people; a list which can contain an unlimited number of IfcTexts*

6.3.9 Building Story Requirements

6.3.9.1 Building Story Requirements

- Main object to which the other building story *Requirements* are linked (Figure 63)
- Attribute set which defines *Requirements* for a building story
- Direct link to *Design Model:* Building story (IfcBuildingStorey)
- Indirect links: Circulation system (IfcSystem – CirculationSystem)

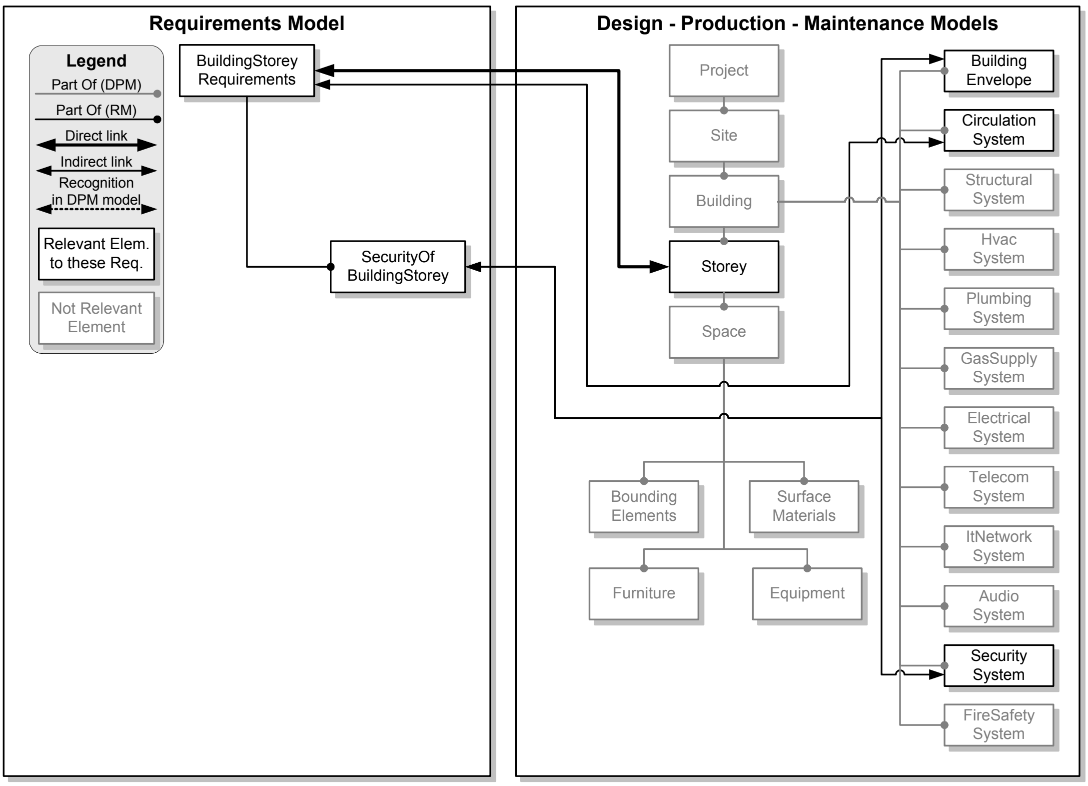

Figure 63: Story *Requirements*

ENTITY ***NewBuildingStoreyRequirements****;*

SUBTYPE OF (*NewRequirement*);

StoreyAccess : OPTIONAL NewRequirementDescription;

SecurityOfBuildingStorey : OPTIONAL ***NewSecurityOfBuildingStorey****;*

END_ENTITY;

Name	**Description**
StoreyAccess	*Description of the access requirements to a building story*

6.3.9.2 Safety of Building Story

- Attribute set which defines *Requirements* for security of a building story (Figure 64)
- Direct link to *Design Model:* Building story (IfcBuildingStorey)
- Indirect links: Building Envelope and security system (IfcSystem – BuildingEnvelope, SecuritySystem)

ENTITY ***NewSecurityOfBuildingStorey****;*

SUBTYPE OF (*NewRequirement*);

StoreyEnvelopeSecurity : OPTIONAL NewRequirementDescription;

StoreyDoorSecurity : OPTIONAL NewRequirementDescription;

StoreyWindowSecurity : OPTIONAL NewRequirementDescription;

END_ENTITY;

Name	**Description**
StoreyEnvelope Security	*Description of the security requirements for the envelope of a building storey*
StoreyDoorSecurity	*Description of the security requirements for the doors of a building storey*
StoreyWindow Security	*Description of the security requirements for the windows of a building storey*

6.3.10 Space Requirements

6.3.10.1 Space Program Instance

- Main object to which the *Space Program Type* is linked (Figure 64).
- Attribute set which defines *Requirements* for a *Space Program Instance* in the *Requirements Model*. One *Space Program Instance* can be linked to several *Space Instances* in the *Design Model* (Sections 5.1.1 and 6.1.7)
- Direct link to *Design Model: Space* (IfcSpace)
- Indirect links: None

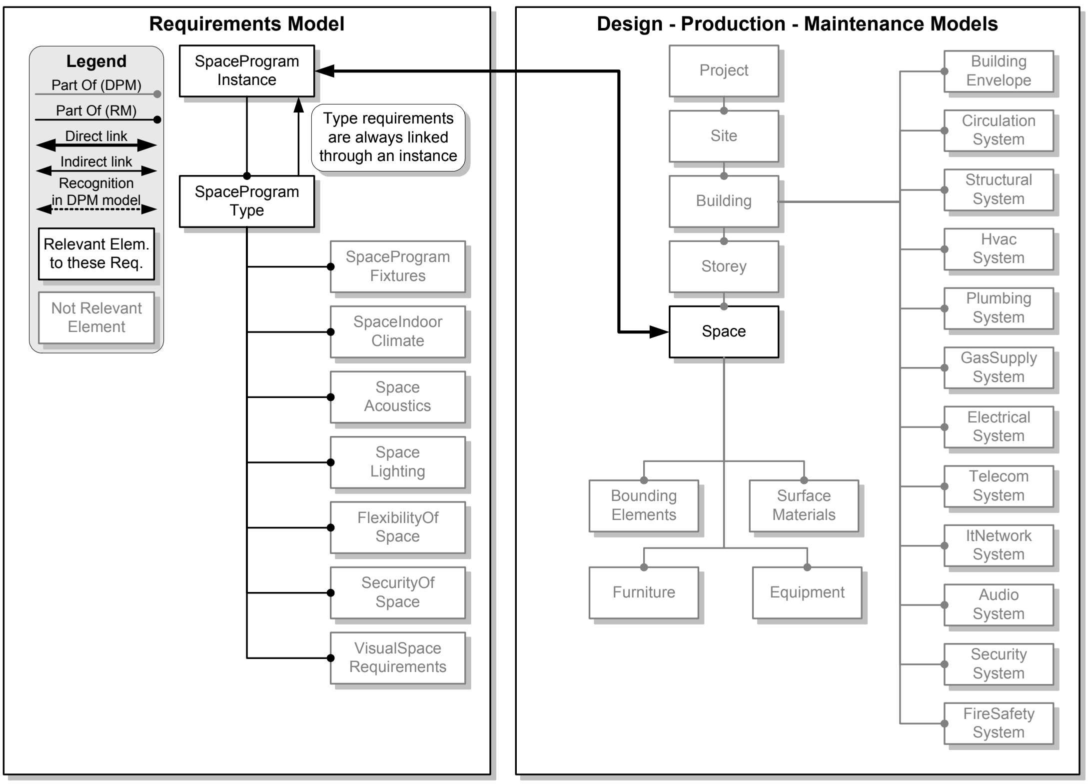

Figure 64: *Space* and *Space Type Requirements* 1/8 – *Instance* and Type

*ENTITY **IfcSpaceProgramInstance**;*

SUBTYPE OF (*NewRequirement*);

StandardRequiredArea : *NewRequirementArea;*

MaxRequiredArea : OPTIONAL *NewRequirementArea;*

MinRequiredArea : OPTIONAL *NewRequirementArea;*

RequestedLocation : OPTIONAL *NewRequirementDescription;*

OccupancyType : OPTIONAL NewRequirementDescription;

EmployeeType : OPTIONAL NewRequirementDescription;

MaxOccupancyNumber : NewRequirementInteger;

NumberOfSpaceUnits : NewRequirementInteger;

Department : OPTIONAL NewRequirementDescription;

AdjacentSpaces : OPTIONAL NewRequirementDescriptionList;

NormalStartTime : NewRequirementAttribute;

NormalEndTime : NewRequirementAttribute;

UseHoursPerDay : NewRequirementInteger;

UseDaysPerWeek : NewRequirementInteger;

*SpaceTypeRequirements : OPTIONAL **NewSpaceProgramType**;*

INVERSE

HasInteractionReqsFrom : SET OF IfcRelInteractionRequirements FOR RelatedSpaceProgram;

HasInteractionReqsTo : SET OF IfcRelInteractionRequirements FOR RelatingSpaceProgram;

END_ENTITY;

Name	**Description**
StandardRequired Area	The floor area programmed for the space; included in the current IfcSpaceProgram object.
MaxRequiredArea	The maximum floor area programmed for the space; included in the current IfcSpaceProgram object.
MinRequiredArea	The minimum floor area programmed for the space; included in the current IfcSpaceProgram object.
RequestedLocation	General description of the required location for the space (e.g., "third floor south"); included in the current IfcSpaceProgram object.
OccupancyType	*Occupancy type for the space. It is defined according to the applicable building code.*
EmployeeType	*General description of the employee type that will occupy the space (e.g. manager, programmer, secretary, etc.). The type classification depends on the company based terms for employee types.*
MaxOccupancy Number	*Maximum number of occupants for the designed usage of the space.*
NumberOf SpaceUnits	*Number of the space units in the building program; the physical instances in the Design Model having identical requirements are linked to one requirements instance. For example, 10 office rooms in the Accounting Department, 12m^2 each, occupied by one person doing normal office work.*
Department	*The department or other unit to which the space belongs*
AdjacentSpaces	*List of spaces which should be located near to the space; an alternative method for "HasInteractionReqsFrom" & "HasInteractionReqsTo" to store information of related spaces*
NormalStartTime	*Time when the use of the space normally starts, for example, 8:00*
NormalEndTime	*Time when the use of the space normally ends, for example, 17:00*

UseHoursPerDay	*Frequency of normal use, how many hours the space is normally used per day. For example, the meeting room will be occupied 4 hours per day*
UseDaysPerWeek	*Frequency of normal use, days per week. For example, the meeting room will be used on 5 days per week*

6.3.10.2 Space Program Type

- Main object which is linked to the *Space Program Instance* and to which all the shared *Space Requirements* are linked (Figure 64)
- Attribute set which defines *Requirements* for a *Space Type* in the *Requirements Model*. Several *Space Instances* in the *Requirements Model* can share these *Requirements*.
- Direct link to *Design Model: Space* (IfcSpace)
- Indirect links: Structural system (IfcSystem – StructuralSystem)

ENTITY ***NewSpaceProgramType****;*

SUBTYPE OF (*NewRequirement*);

Activities : OPTIONAL NewRequirementDescriptionList;

FunctionRequirements : OPTIONAL NewRequirementDescription;

SpecialLoadRequirements : OPTIONAL NewRequirementDescription;

VibrationControl : OPTIONAL NewRequirementDescription;

SpaceProgramFixtures : OPTIONAL ***NewSpaceProgramFixtures****;*

SpaceIndoorClimate : OPTIONAL ***NewSpaceIndoorClimate;***

SpaceAcoustics : OPTIONAL ***NewSpaceAcoustics****;*

SpaceLighting : OPTIONAL ***NewSpaceLighting****;*

FlexibilityOfSpace : OPTIONAL ***NewFlexibilityOfSpace****;*

SafetyOfSpace : OPTIONAL ***NewSecurityOfSpace****;*

ComfortOfSpace : OPTIONAL ***NewComfortOfSpace****;*

END_ENTITY;

Name	**Description**
Activities	*Description of main activities in the space; a list which can contain an unlimited number of IfcTexts*
Function Requirements	*General description of the functional requirements for the space (in addition to the space name)*
SpecialLoad Requirements	*Description of special load requirements, such as heavy equipments and archive shelves*
VibrationControl	*Description of vibration control requirements, for example, caused by sensitive measurement equipment*

6.3.10.3 Space Program Fixtures

- Attribute set which defines *Requirements* for fixtures, furniture, equipment and finishes of a *Space Type* (Figure 65)
- Direct link to *Design Model: Space* (IfcSpace)
- Indirect links: None, but recognition of *Space*-related elements in the *DPM Models* required; *Bounding Elements*, furniture, equipment, and surface materials

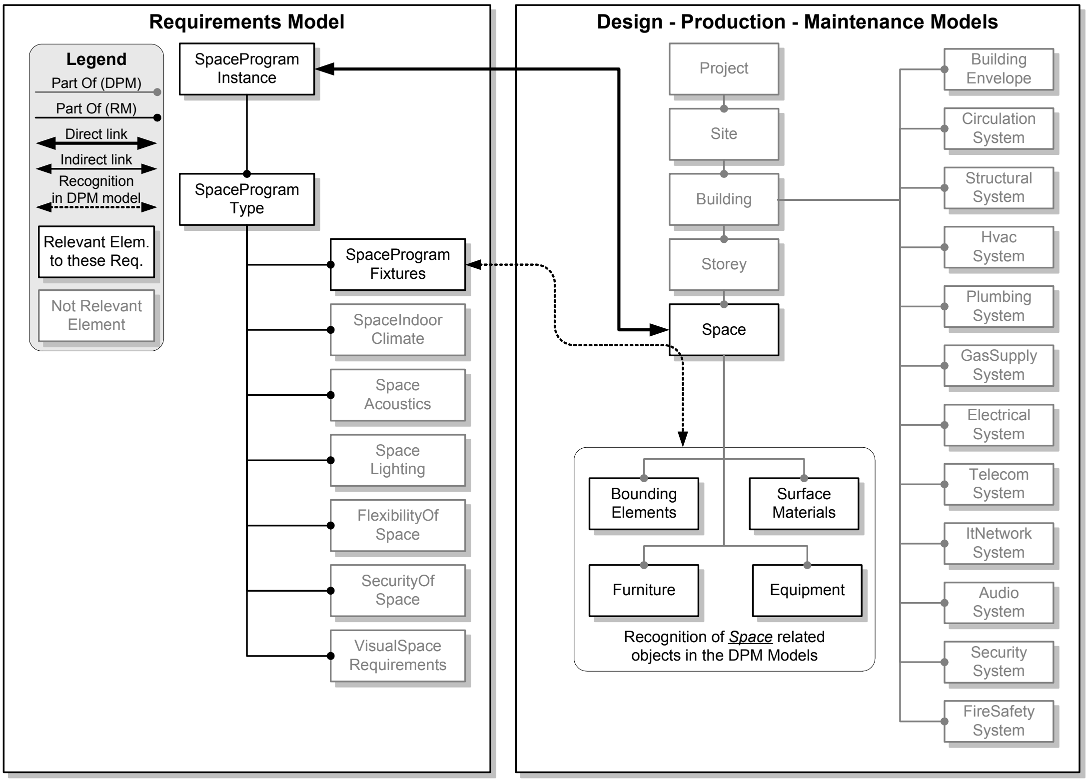

Figure 65: *Space Type Requirements* 2/8 – Doors, windows, furniture, equipment, finishes and fixtures

*ENTITY **NewSpaceProgramFixtures**;*

SUBTYPE OF (*NewRequirement*);

AccessFloor : OPTIONAL NewRequirementDescription;

FloorSurface : OPTIONAL NewRequirementDescription;

Doors : OPTIONAL NewRequirementDescriptionList;

Windows : OPTIONAL NewRequirementDescriptionList;

Fixtures : OPTIONAL NewRequirementDescriptionList;

Furniture : OPTIONAL NewRequirementDescriptionList;

Equipment : OPTIONAL NewRequirementDescriptionList;

AvEquipment : OPTIONAL NewRequirementDescriptionList;

WallFinishes : OPTIONAL NewRequirementDescription;

CeilingFinishes : OPTIONAL NewRequirementDescription;

CeilingHeight : OPTIONAL NewRequirementDistance;

END_ENTITY;

Name	**Description**
AccessFloor	*Description of the access floor requirements*
FloorSurface	*Description of the floor surface requirements*
Doors	*Description of the door requirements, such as size, material, and sound insulation; a list which can contain an unlimited number of IfcTexts*
Windows	*Description of the window requirements, such as size, material, and sound insulation; a list which can contain an unlimited number of IfcTexts*
Fixtures	*Description of the fixture requirements; a list which can contain an unlimited number of IfcTexts*
Furniture	*Description of the furniture requirements; a list which can contain an unlimited number of IfcTexts*
Equipment	*Description of the equipment requirements; a list which can contain an unlimited number of IfcTexts*
AvEquipment	*Description of the audio-visual equipment requirements; a list which can contain an unlimited number of IfcTexts*
WallFinishes	*Description of the wall surface requirements*
CeilingFinishes	*Description of the ceiling surface requirements*
CeilingHeight	*Definition of the required free ceiling height*

6.3.10.4 Space Indoor Climate

- Attribute set which defines *Requirements* for the indoor air quality and other condition *Requirements* of a *Space Type* (Figure 66)
- Direct link to *Design Model: Space* (IfcSpace)
- Indirect links: HVAC system (IfcSystem – HvacSystem)

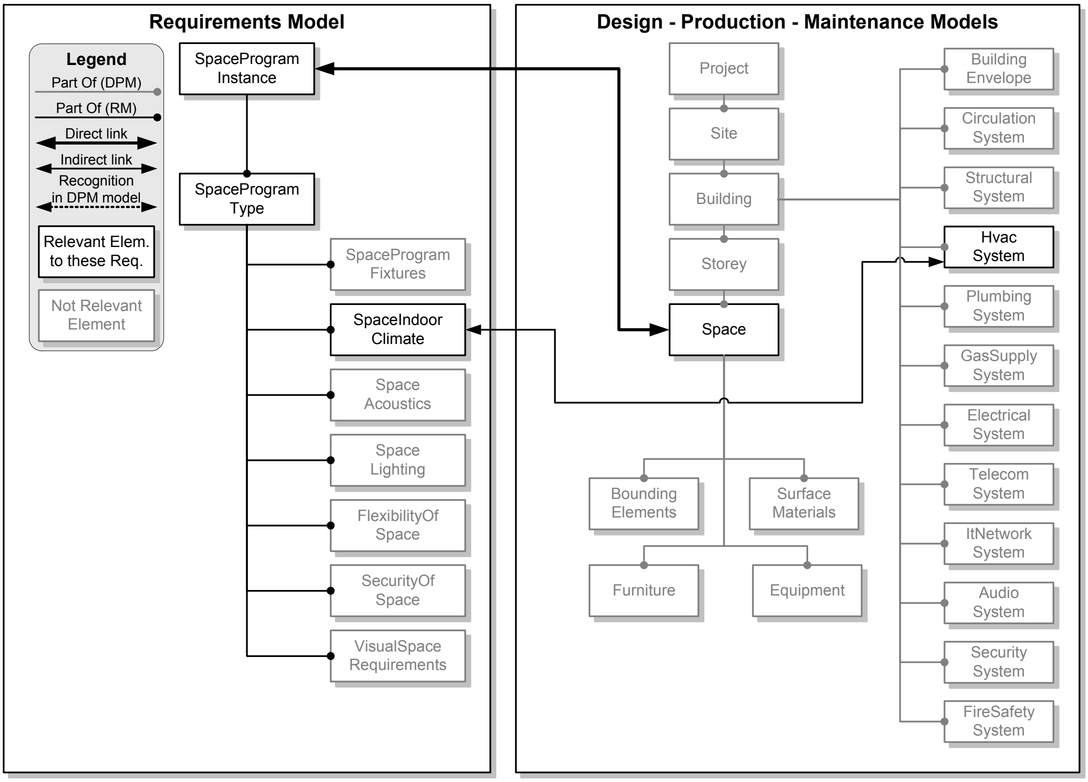

Figure 66: *Space Type Requirements* 3/8 – Indoor climate

ENTITY ***NewSpaceIndoorClimate;***

SUBTYPE OF (*NewRequirement*);

MaxHvacNoiseLevel : OPTIONAL NewRequirementSound;

MaxTemperature : OPTIONAL NewRequirementTemperature;

MinTemperature : OPTIONAL NewRequirementTemperature;

IndividualRoomTemperatureControl : OPTIONAL NewRequirementTemperature;

MaxHumidity : OPTIONAL NewRequirementRatio;

MinHumidity : OPTIONAL NewRequirementRatio;

MaxAirVelocity : OPTIONAL NewRequirementAttribute;

MinAirflowPerPerson : OPTIONAL NewRequirementAttribute;

MinNoOccupancyAirChangeRate : OPTIONAL NewRequirementRatio;

MaxFloorTemperature : OPTIONAL NewRequirementTemperature;

MinFloorTemperature : OPTIONAL NewRequirementTemperature;

TemporarilyVentilationControl : OPTIONAL NewRequirementRatio;

AllowedTemporaryDeviation : OPTIONAL NewRequirementTemperature;

MaxVerticalTemperatureDifference : OPTIONAL NewRequirementTemperature;

END_ENTITY;

Name	Description
MaxHvacNoiseLevel	*Maximum allowed noise level caused by building services systems*
MaxTemperature	*Maximum temperature, typically ºC or ºF*
MinTemperature	*Minimum temperature, typically º C or ºF*
IndividualRoom TemperatureControl	*Control range for individual settings for the space bypassing system settings, typically ±x º C or ºF*
MaxHumidity	*Maximum relative humidity*
MinHumidity	*Minimum relative humidity*
MaxAirVelocity	*Maximum air velocity in the space*
MinAirflow PerPerson	*Minimum airflow per person (Maximum number of people in the space is defined by the MaxOccupancyNumber attribute in Pset_SpaceProgramInstance)*
MinNoOccupancy AirChangeRate	*Minimum air change rate in the space when not occupied*
MaxFloor Temperature	*Maximum temperature of the floor surface*
MinFloor Temperature	*Minimum temperature of the floor surface*
Temporarily VentilationControl	*Temporary individual adjustments of the ventilation*
Allowed Temporary Deviation	*Allowed temporary deviation from the defined minimum and maximum temperatures in exceptional weather conditions*
MaxVerticalTem-peratureDifference	*Maximum vertical temperature difference in the occupied zone*

6.3.10.5 Space Acoustics

- Attribute set which defines acoustical *Requirements* of a *Space Type* (Figure 67)
- Direct link to *Design Model: Space* (IfcSpace)
- Indirect links: Audio system and Building Envelope (IfcSystem – AudioSystem, BuildingEnvelope), also recognition of *Bounding Elements* and surface materials of the *Space* in *DPM Models* required

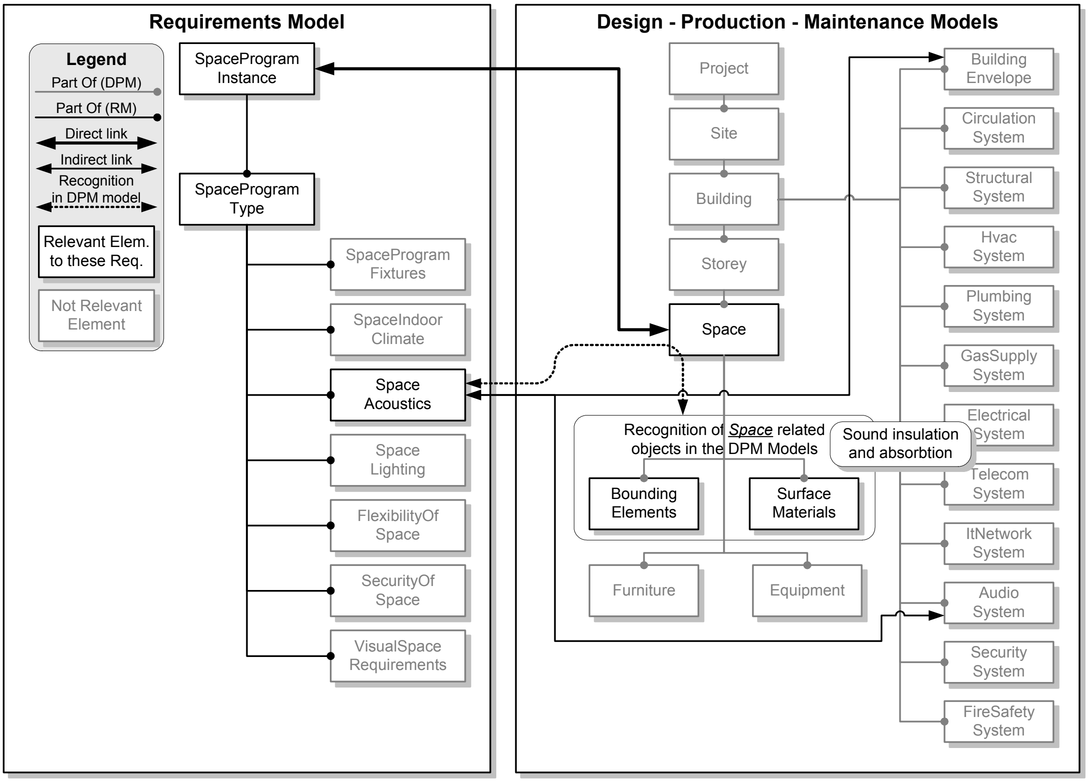

Figure 67: *Space Type Requirements* 4/8 – Acoustics

ENTITY ***NewSpaceAcoustics****;*

SUBTYPE OF (*NewRequirement*);

BackGroundSound : OPTIONAL NewRequirementDescription;

MaxReverberationTime : OPTIONAL NewRequirementReal;

MinReverberationTime : OPTIONAL NewRequirementReal;

MinSoundInsulation : OPTIONAL NewRequirementSound;

MaxTrafficNoiseLevel : OPTIONAL NewRequirementSound;

END_ENTITY;

Name	**Description**
BackGroundSound	*Description of the background sound system and/or level in the space*
MaxReverberation Time	*Maximum reverberation time; typically required for lecture halls, staircases, hallways, etc.*
MinReverberation Time	*Minimum reverberation time; typically required for concert halls and other music facilities*
MinSoundInsulation	*Required insulation between spaces*
MaxTrafficNoise Level	*Allowable maximum traffic noise level in the space*

6.3.10.6 Space Lighting

- Attribute set which defines lighting *Requirements* of a *Space Type* (Figure 68)
- Direct link to *Design Model: Space* (IfcSpace)
- Indirect links: Electrical systems (IfcSystem – ElectricalSystem), also recognition of *Space*-related elements in the *DPM Models* required; *Bounding Elements* (windows) and colors of furniture, equipment and surface materials

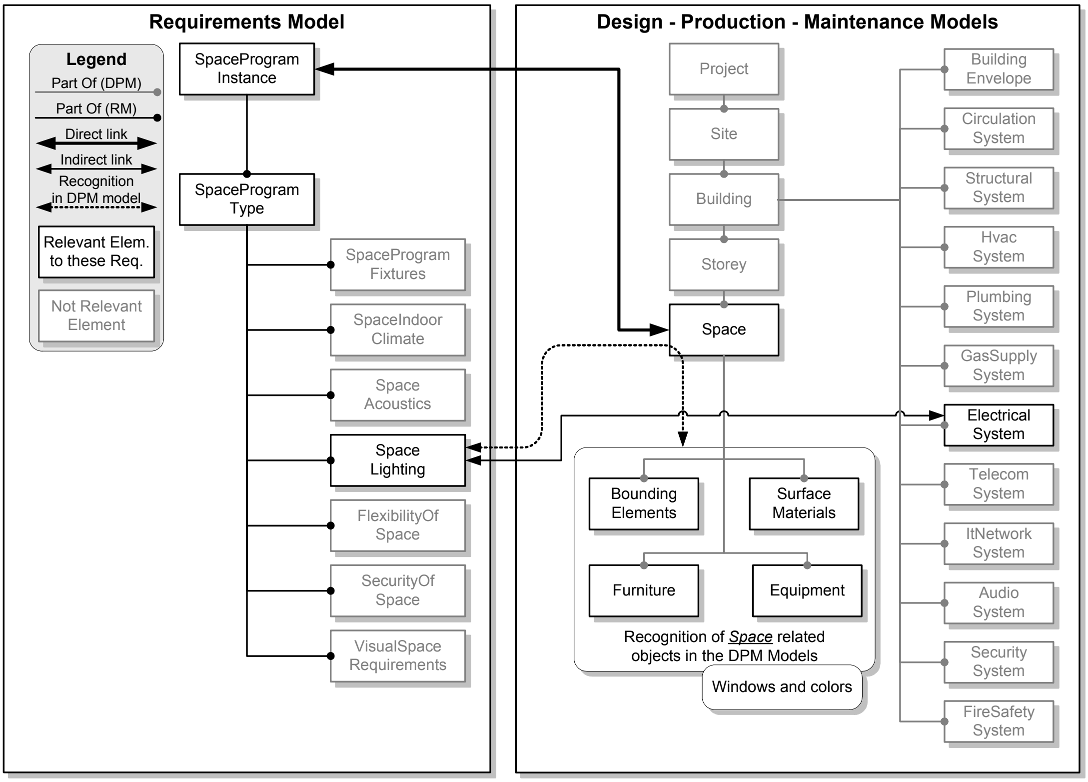

Figure 68: *Space Type Requirements* 5/8 – Lighting

*ENTITY **NewSpaceLighting**;*

SUBTYPE OF (*NewRequirement*);

Daylight : OPTIONAL NewRequirementAttribute;

NoDaylight : OPTIONAL NewRequirementAttribute;

Darkenable : OPTIONAL NewRequirementDescription;

MinLampEnergyEfficiency : OPTIONAL NewRequirementRatio;

MaxLuminance : OPTIONAL NewRequirementAttribute;

MinLuminance : OPTIONAL NewRequirementAttribute;

LuminanceDistribution : OPTIONAL NewRequirementDescription;

LightingAdjustability : OPTIONAL NewRequirementDescription;

LusterReflection : OPTIONAL NewRequirementDescription;
ColorRenderingIndex : OPTIONAL NewRequirementRatio;
MaxColorTemperature : OPTIONAL NewRequirementTemperature;
MinColorTemperature : OPTIONAL NewRequirementTemperature;
DirectionalLighting : OPTIONAL NewRequirementDescription;
GlareIndex : OPTIONAL NewRequirementRatio;
ShadowFormation : OPTIONAL NewRequirementDescription;
ContrastReproduction : OPTIONAL NewRequirementRatio;
LightingUniformity : OPTIONAL NewRequirementDescription;
TaskLighting : OPTIONAL NewRequirementDescription;
END_ENTITY;

Name	**Description**
Daylight	*Daylight required in the space*
NoDaylight	*Daylight not allowed in the space*
Darkenable	*Description of shade requirements for windows and other openings*
MinLampEnergy Efficiency	*Minimum required energy efficiency of the light sources*
MaxLuminance	*Maximum illuminance in the working area*
MinLuminance	*Minimum illuminance in the working area*
Luminance Distribution	*The luminance distribution on different surfaces in the field of view determined by the reflectance and the illuminance on the surfaces*
LightingAdjustability	*Description of the required level of individual lighting control in the space*
LusterReflection	*Allowable level of luster reflection in the space*
ColorRenderingIndex	*Required minimum color rendering index for light sources in the space*
MaxColor Temperature	*Maximum color temperature of the light sources in the space*
MinColor Temperature	*Minimum color temperature of the light sources in the space*
DirectionalLighting	*Description of the required level of directional lighting*
GlareIndex	*Required maximum glare index for light sources in the space*
ShadowFormation	*Description of the balance between diffuse and directional light in the space*
Contrast Reproduction	*Minimum contrast reproduction index value (CRF) for the space*
LightingUniformity	*Description of the lighting uniformity requirements in the space*
TaskLighting	*Description of task lighting requirements in the space*

6.3.10.7 Flexibility of Space

- Attribute set which defines the flexibility *Requirements* of a *Space Type* (Figure 69).
- Direct link to *Design Model: Space* (IfcSpace)
- Indirect links: None, but recognition of the *Bounding Elements* in the *DPM Models* required

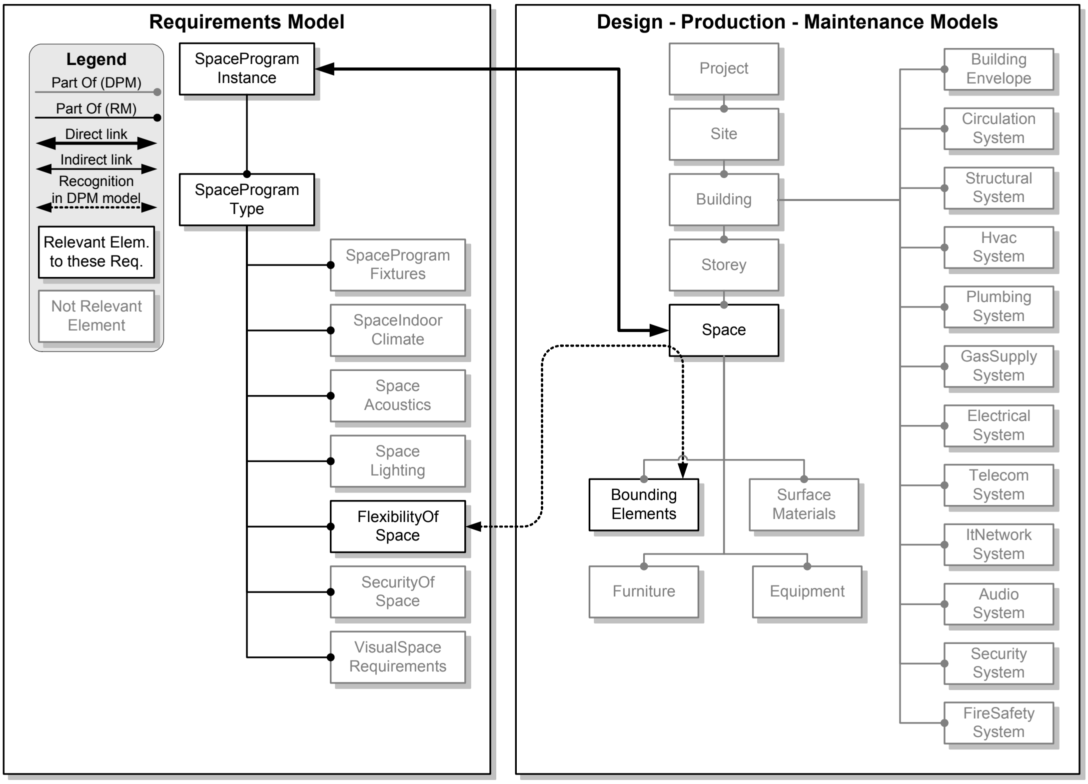

Figure 69: *Space Type Requirements* 6/8 – Flexibility

ENTITY ***NewFlexibilityOfSpace****;*

SUBTYPE OF (*NewRequirement*);

AlternativeFurnishing : OPTIONAL NewRequirementDescriptionList;

AlternativeUse : OPTIONAL NewRequirementDescriptionList;

DivisionAndCombination : OPTIONAL NewRequirementDescriptionList;

END_ENTITY;

Name	**Description**
AlternativeFurnishing	*Description of the alternative furnishing requirements; a list which can contain an unlimited number of IfcTexts*
AlternativeUse	*Description of the alternative usage requirements; a list which can contain an unlimited number of IfcTexts*
DivisionAnd Combination	*Requirements for the division and/or combination flexibility for the space; a list which can contain an unlimited number of IfcTexts*

6.3.10.8 Security of Space

- Attribute set which defines the security *Requirements* of a *Space Type* (Figure 70).
- Direct link to *Design Model: Space* (IfcSpace)
- Indirect links: Security and Fire Safety systems (IfcSystem – SecuritySystem, FireSafety), also recognition of *Bounding Elements* in the *DPM Models* required.

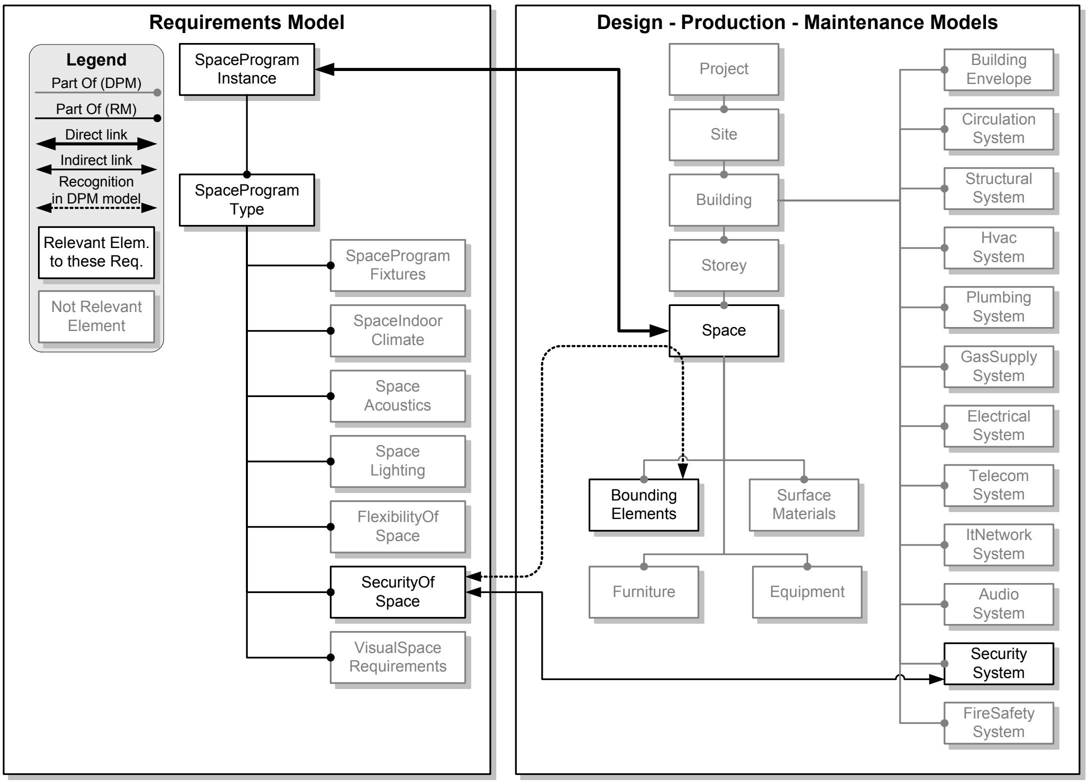

Figure 70: *Space Type Requirements* 7/8 – Security

ENTITY ***NewSecurityOfSpace****;*

SUBTYPE OF (*NewRequirement*);

AccessZone : OPTIONAL NewRequirementDescription;

AccessControl : OPTIONAL NewRequirementDescriptionList;

END_ENTITY;

Name	Description
AccessZone	*Description of the access zone to which the space belongs*
AccessControl	*Description of the access control of the space; key, electric lock, card reader, RFID, etc; a list which can contain an unlimited number of IfcTexts*

6.3.10.9 Functionality and Visual Contacts of Space

- Attribute set which defines the visual contact *Requirements* for a *Space Type* (Figure 71)
- Direct link to *Design Model: Space* (IfcSpace)
- Indirect links: Building envelope (IfcSystem – BuildingEnvelope), also recognition of *Bounding Elements* of the *Space* required

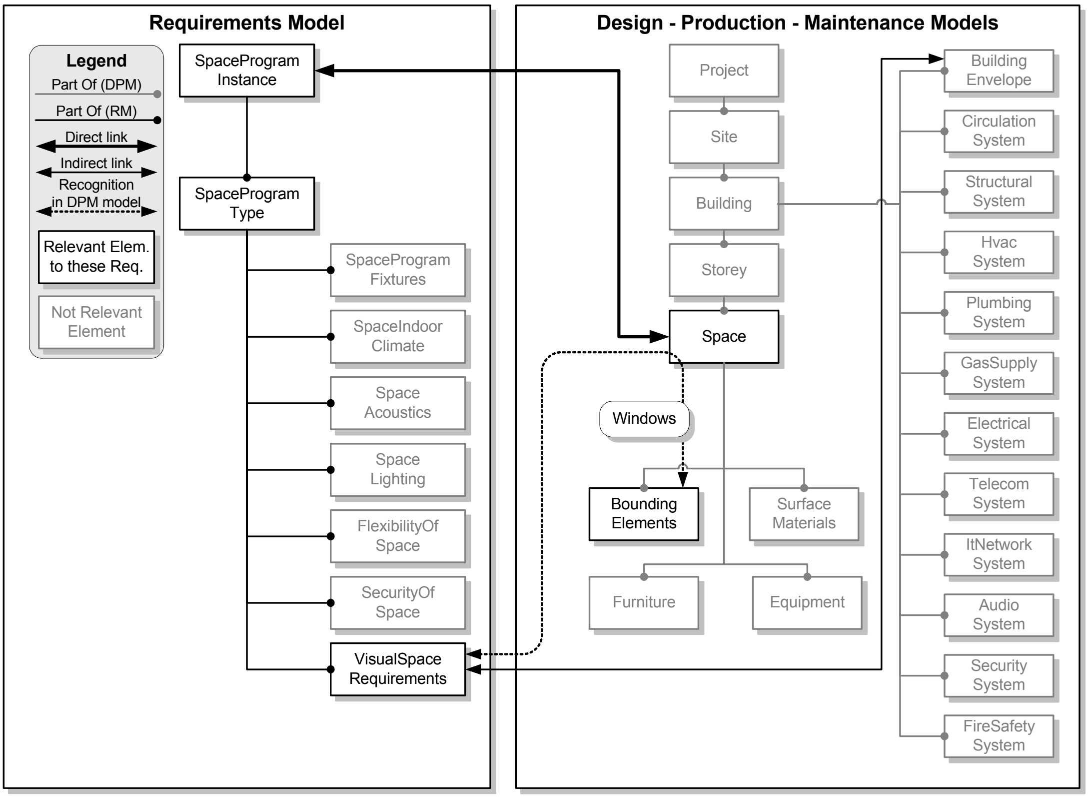

Figure 71: *Space Type Requirements* 8/8 – Comfort

*ENTITY **NewVisualRequirementsForSpace**;*

SUBTYPE OF (*NewRequirement*);

InteriorDesignAndFunctionality : OPTIONAL NewRequirementDescription;

ExternalVisualContacts : OPTIONAL NewRequirementDescriptionList;

InternalVisualContacts : OPTIONAL NewRequirementDescriptionList;

END_ENTITY

Name	**Description**
InteriorDesignAnd Functionality	*Description of general design requirements for the space*
ExternalVisual Contacts	*Description of contact or privacy requirements outside of the building; a list which can contain an unlimited number of IfcTexts*
InternalVisual Contacts	*Description of contact or privacy requirements inside of the building; a list which can contain an unlimited number of IfcTexts*

6.3.11 Building Envelope Requirements

6.3.11.1 Building Envelope Requirements

- Attribute set which defines *Requirements* for *Building Envelope* (Figure 72)
- Direct link to *Design Model:* Building envelope (IfcSystem – BuildingEnvelope)
- Indirect links: None

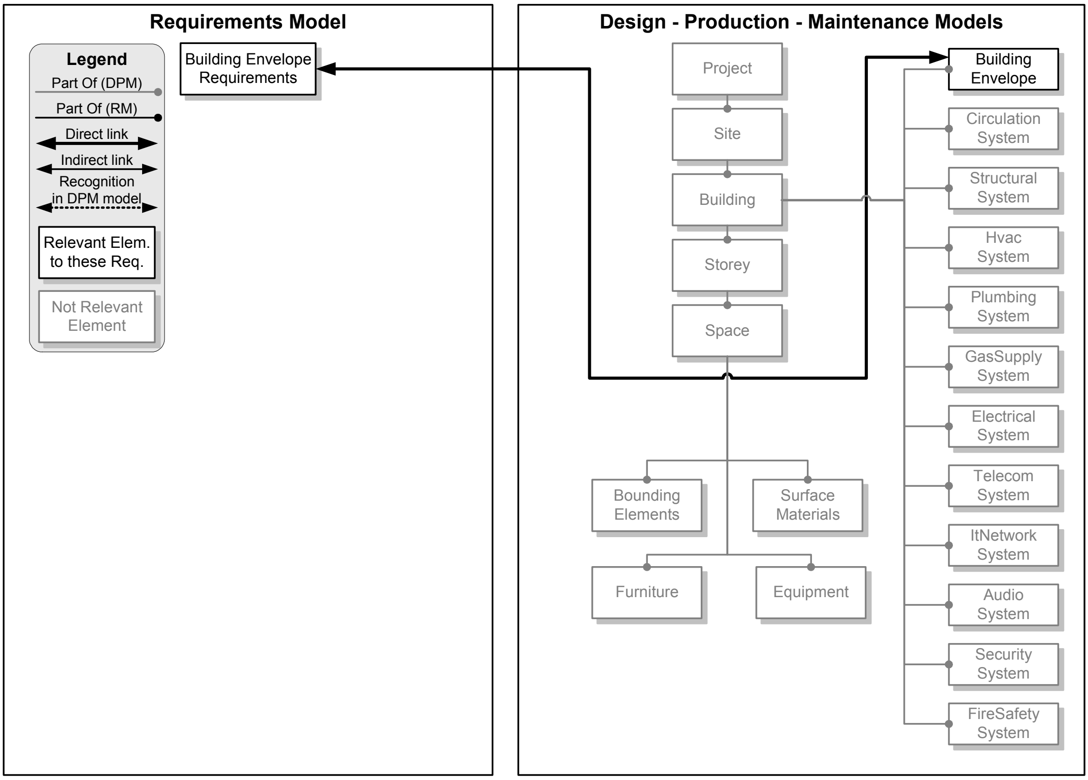

Figure 72: *Building Envelope Requirements*

ENTITY ***NewBuildingEnvelopeRequirements****;*

SUBTYPE OF (*NewRequirement*);

AestheticEnvelopeRequirements : OPTIONAL NewRequirementDescriptionList;

EnvelopeVentilation : OPTIONAL NewRequirementDescription;

MaxEnvelopeAirLeakage : OPTIONAL NewRequirementAttribute;

MinEnvelopeSoundInsulation : OPTIONAL NewRequirementSound;

BaseFloorInsulation : OPTIONAL NewRequirementReal;

ExternalWallInsulation : OPTIONAL NewRequirementReal;

EnergySavingBufferSpaces : OPTIONAL NewRequirementDescription;

ExternalDoorInsulation : OPTIONAL NewRequirementReal;

WindowInsulation : OPTIONAL NewRequirementReal;

WindowShading Coefficient : OPTIONAL NewRequirementReal;

SolarProtection : OPTIONAL NewRequirementDescription;

RoofInsulation : OPTIONAL NewRequirementReal;

END_ENTITY;

Name	Description
AestheticEnvelope Requirements	*Description of aesthetic requirements for the building envelope; a list which can contain an unlimited number of IfcTexts*
EnvelopeVentilation	*Description of ventilation requirements for the building envelope*
MaxEnvelope AirLeakage	*Allowable maximum air leakage value of the building envelope*
MinEnvelopeSound Insulation	*Required minimum sound insulation of the building envelope*
BaseFloorInsulation	*Required minimum insulation of base floor structures*
ExternalWall Insulation	*Required minimum insulation of external walls*
EnergySaving BufferSpaces	*Description of requirements to use 'buffer spaces' for energy saving (zone between the outdoor and heated and/or cooled spaces)*
ExternalDoor Insulation	*Required minimum insulation of external doors*
WindowInsulation	*Required minimum insulation of external windows*
WindowShading Coefficient	*Required minimum shading coefficient value for windows*
SolarProtection	*Description of requirements for solar protection and shading devices*
RoofInsulation	*Required minimum insulation of roof structures*

6.3.12 Circulation System Requirements

6.3.12.1 Circulation System Requirements

- Attribute set which defines the circulation system *Requirements* for the building (Figure 72)
- Direct link to *Design Model:* Circulation system (IfcSystem – CirculationSystem)
- Indirect links: None

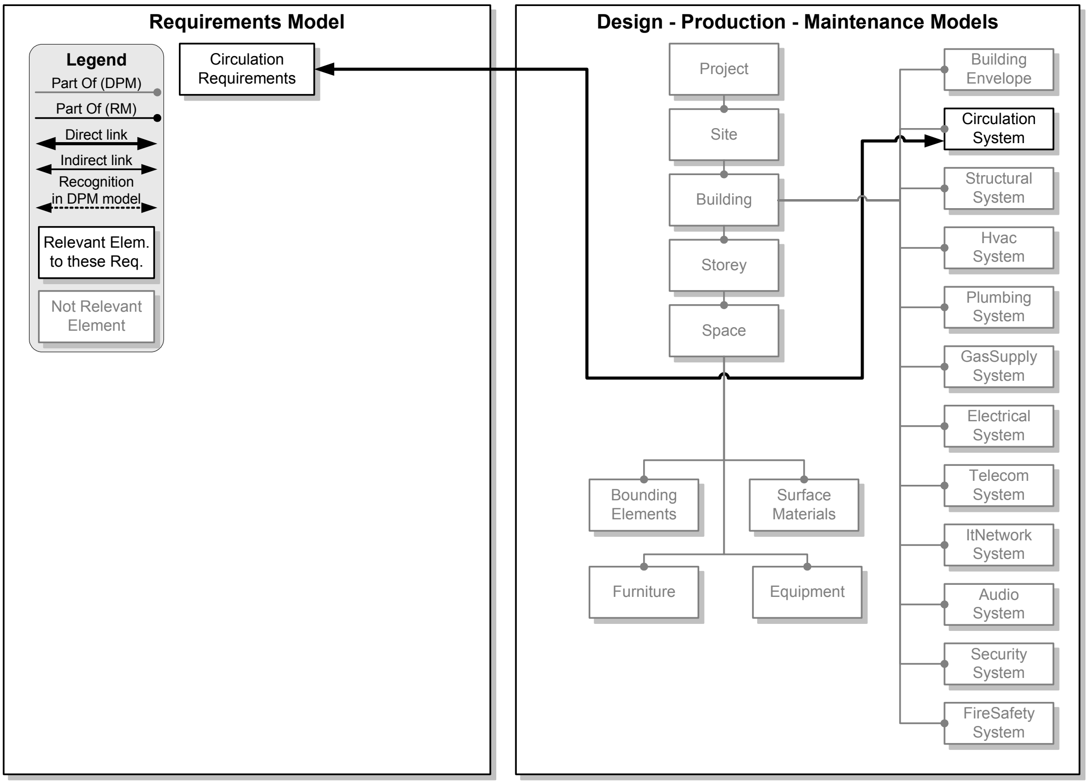

Figure 73: *Circulation System Requirements*

*ENTITY **NewCirculationSystemRequirements**;*

SUBTYPE OF (*NewRequirement*);

MaxCirculationAreaRatio : OPTIONAL NewRequirementRatio;

LobbyRequirements : OPTIONAL NewRequirementDescriptionList;

CorridorRequirements : OPTIONAL NewRequirementDescriptionList;

StairRequirements : OPTIONAL NewRequirementDescriptionList;

ElevatorRequirements : OPTIONAL NewRequirementDescriptionList;

EscalatorRequirements : OPTIONAL NewRequirementDescriptionList;

LoadingDockRequirements : OPTIONAL NewRequirementDescriptionList;

END_ENTITY;

Name	Description
MaxCirculation AreaRatio	*Maximum ratio of circulation area in the building compared to the Space Program area*
LobbyRequirements	*Description of the requirements for the lobby; a list which can contain an unlimited number of IfcTexts*
Corridor Requirements	*Description of the requirements for the corridors; a list which can contain an unlimited number of IfcTexts*
StairRequirements	*Description of the requirements for the stairs; a list which can contain an unlimited number of IfcTexts*
Elevator Requirements	*Description of the requirements for the elevators; a list which can contain an unlimited number of IfcTexts*
Escalator Requirements	*Description of the requirements for the escalators; a list which can contain an unlimited number of IfcTexts*
LoadingDock Requirements	*Description of the requirements for the loading dock; a list which can contain an unlimited number of IfcTexts*

6.4 Implementation View

The full *Requirements Model Specification* consists of a large number of *Requirements*, 300 in total. They cover most architectural design *Requirements* for the buildings which are in the scope of my research; office and laboratory buildings (Section 1.3). The question is: are there too many *Requirements* rather than is something missing from the *Specification*. However, the purpose of my research was to create a theoretical framework and sufficient foundation for practical implementation and it is easier to make implementation agreements by leaving something out from the supported view than adding something new. Thus, I argue that an inclusive approach, which leads to a large number of *Requirements,* is the best possible solution at this point.

The implementation of the *Requirements Model Specification* is not in the scope of my research. However, documenting several issues which have come up during the rapid prototyping and the development of the *Specification* are useful for further research and also for the practical implementation of the *Specification*. Thus, they are recorded in this Section.

6.4.1 Basic Guideline

As documented in Section 3.5, BLIS has developed several *Implementation Views* for the *IFC Specifications*. The basic idea of these views is to define an exact subset of the *Specification* for implementation. On the BLIS website Jiri Hietanen has defined the *Implementation View* for ‘Client Brief / Space Layout ⇨ Architectural Design’ which is documented in Section 3.5 [*BLIS 2004* [92]].

My *Requirements Model* is changing this definition and adding several new possibilities to use the *Requirements*. Thus, my proposal for the new ‘Requirements ⇨ Architectural Design’ implementation guideline is the following:

‘Requirements ⇨ Architectural Design’ consists of two views, ‘Space Requirements View ⇨ Architectural Design’ and ‘Project Requirements View ⇨ Architectural Design.’ These views define two subsets of the *Requirements Model Specification* to link a *Requirements Model* with an architectural *Design Model*.

The *Requirements Management* application can be anything from a simple spreadsheet to a dedicated *Model Server* application. The crucial demand is that it can create links between the objects in the *Requirements Model* and in the architectural *Design Model*. This requires that the information in the *Requirements Management* application is structured according to the *Requirements Model Specification*, and that the design application has (1) the logical structure defined in the *IFC Specifications*, and (2) each entity type includes a usable identifier for the link.

The recommended technical solution is a *Model Server* including both *Requirements and Design Models*. To enable the linkage between the *Requirements* and *Spaces* in the *Design Model* I recommend using an application which can create the ‘skeleton’ for *Spaces*, e.g., the right number of *Spaces* of the right types and areas, and automatically link the *Spaces* in the *Design Model* with their *Requirements* in the *Requirements Model*.

The first level, the ‘Space Requirements View ⇨ Architectural Design,’ enables the above-mentioned generation of ‘space skeleton.’ The second level, ‘Project Requirements View ⇨ Architectural Design,’ enables the creation of active links between all levels in the *Requirements Model* and *Design Model* and the checking of *Requirements* from the design application.

6.4.2 Linkage of Requirements to the DPM Models

The current 'Client Brief / Space Layout ⇨ Architectural Design' *Implementation View* is based on file exchange. On that level the main functionality is to generate a 'space skeleton' because the view does not include *Requirements* other than required area, name and type of the *Spaces*.

My *Requirements Model Specification* is based on a different approach. I store the *Requirements* in their own *Instantiated Model*, and they are linked to the *Design Model*, (Figure 74). Section 6.3.2 documents the technical solution for the link is in detail.

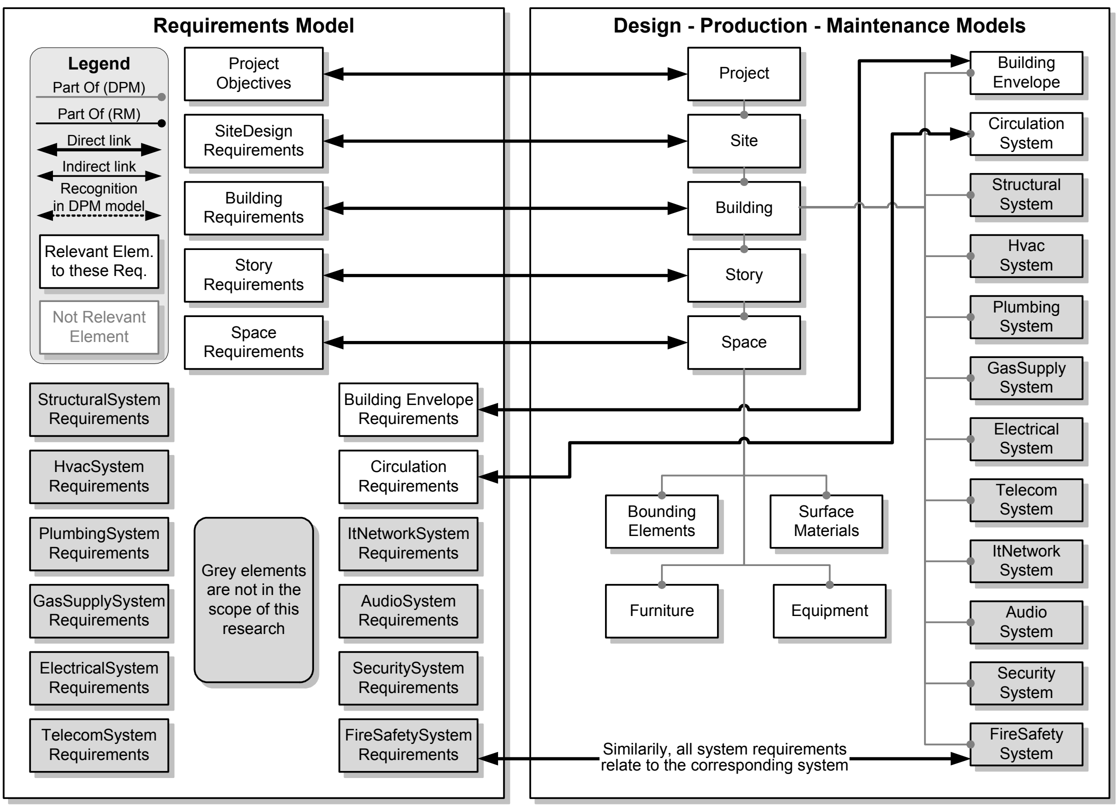

Figure 74: Connection types between *Requirements* and *DPM Models*

Spaces are the most complicated objects from the *Requirements Management* viewpoint. The number of *Spaces* can be very large. Thus, manual linkage can be a time-consuming and error-prone process. Automated linkage between an existing *Design Model* and *Requirements Model* would demand that the *Space* identifiers are exactly the same in both *Models*. Thus, linking *Space Requirements* after the *Design Model* is created is not a recommendable method.

The preferred method is to create the "spatial skeleton" (Sections 3.5, 7.1.1 and Appendix C, C1) of the *Design Model* automatically from the *Requirements Model*. At the same time, it is naturally possible to create the "skeleton" for the whole building, and create automatically the links between all *Requirements Objects* and objects in the *Design Model*. Section 7.1.1 documents this method and its advantages in more detail.

6.4.3 Contents in the Requirements Model Applications

In Section 6.4.1, I propose two subsets of *Requirements* for two *Implementation Views:* (1) 'Space Requirements View ⇨ Architectural Design' and (2) 'Project Requirements View ⇨ Architectural Design.' The content of 'Space Requirements View ⇨ Architectural Design' should consist of the *Requirements Objects* specified in Section 6.3.10, and 'Building Project Requirements View ⇨ Architectural Design' contains all *Requirements* (Sections 6.3.6–6.3.12).

The *Requirements Objects* included in the *Implementation Views* can be agreed on in detail within the implementation group when software vendors implement the *Requirements Model Specification* into practical products.

7 Model Validation

The validation criteria for the *Requirements Model Specification* are:

1. **Usefulness**: Does the *Requirements Model Specification* address relevant factors of the identified problem and could its implementation into a tool improve the current process?
2. **Generality**: Does the *Requirements Model Specification* cover a reasonable part of the identified problem?
3. **Implementability**: Is the *Requirements Model Specification* possible to implement?

As mentioned in Section 2.4, there is no objective method to validate a *Model Specification*. Thus the validation must be based on:

- Comparison of the potential *Model* features and problems in real projects: Are the identified problems related to the *Requirements Management*, and could the implementation of my *Requirements Model Specification* help to solve these problems?
- Comparison of the *Specification* content and the *Requirements* in real projects: Does my *Requirements Model Specification* include elements for the *Requirements* related to the identified problems, and is the *Specification* general enough to cover a reasonable number of the *Requirements* in a typical project and discipline which are in the scope of my research?
- Implementability of other *Specifications* based on similar methods: Are there any existing examples of implementation of a similar idea and similar *Specifications*, and how will the experts in that field evaluate my *Specification* from the implementation viewpoint?

7.1 Usefulness of the Requirements Model Specification

In the AEC industry a *Project Team* usually works together in one project only, and each case has different challenges and problems. *Requirements Management* is one of the many sub-processes in the design and construction process. Because of the unique nature of building projects it is difficult, if not impossible, to quantify the benefits of one factor in the process. However, it is possible to identify successful processes, and also clear mistakes in the projects. The examples of the following five projects illustrate some examples and their relation to my *Requirements Model Specification*.

7.1.1 ICL Headquarters

The ICL Headquarters project built in 1994–1996 in Helsinki was the first project using some of the concepts in my *Requirements Model Specification* (Section 1.2.1). However, the tools used in the ICL project were not based on a formally defined *Specification*, and in this project only the area information of the *Space Program* was linked to the *Design Model*. However, it demonstrates the potential of the link between the *Requirements* and design solutions and it is also an example of the implementability of the idea.

The design schedule was extremely tight. The design process started in April 1994 and the construction work began in September 1994. The five-month design period included not only design but also the building permit and cost estimation processes. The key objective was the total cost of the building, but also quality *Requirements* were relatively high. Because the volume of a building is the most important single cost factor, it was important to keep the size of the building as small as possible so that the design still met the *Space Program Requirements*.

The design process started with automatic creation of the "spatial skeleton" (Sections 3.5 and Appendix C, C1) from the *Space Program* which the Project Manager created in MS Excel. This "spatial skeleton" was generated using my software application, KIVI, and it was based on the extended data possibilities of

AutoCAD blocks and polyline objects. In schematic design these *Space Objects* were simple scalable rectangular blocks. The application included a set of tools which enabled modification of the dimensions of these primitive *Space Objects* maintaining the required area. This system allowed rapid testing of different layouts by moving *Spaces* and departments around to find an optimal building shape for a relatively difficult site (Figure 75).

Figure 75: ICL Headquarters, ground floor

One additional benefit from this “spatial skeleton” was that the *Space* blocks provided an excellent method to prevent some of the 800 *Spaces* from being forgotten. Later in design, when the building started to take its final shape, these blocks were automatically transformed to more flexible objects consisting of polylines so that any *Space* shape was possible. AutoCAD's extended data were used to link the objects together in the drawings. These data also enabled automated calculation of areas and area information linkage back to the MS Excel spreadsheets (Figure 9).

During the entire design process, the *Target Values* were compared to the design solutions almost in real-time, at least once a week, by exporting the actual areas from the drawings into MS Excel and comparing them to the *Target Values* (Table 12). The net area in the *Building Program* was about 20,000 m^2, consisting of about 800 *Spaces*. The process would have been impossible in the required schedule without a system which could automatically calculate all details of the program areas for each business unit as well as the gross area.

The area information was used also in cost estimation by combining the *Room* information with the *Room* specifications to calculate the amount of different materials and finishes. This improved both speed and accuracy of the process.

Table 12: ICL Headquarters area table, total areas compared to the targets

Office Building	Program Area	Rentable Area	Circulation Area	Technical Area	Other Areas	Building Permit Area	Gross Area	Volume
Lower Basement	350	402	50	61	461	0	461	1,200
Upper Basement	3,710	4,271	679	390	3,302	1,640	4,942	24,200
Ground Floor	4,211	4,445	433	73	0	4,753	4,753	23,200
1. Floor	2,065	2,510	517	89	0	2,867	2,867	9,700
2. Floor	2,132	2,510	422	89	0	2,871	2,871	9,700
3. Floor	2,028	2,510	524	89	0	2,871	2,871	9,700
4. Floor	2,143	2,510	483	89	0	2,871	2,871	9,700
5. Floor	1,932	2,510	621	89	0	2,871	2,871	9,700
6. Floor	1,613	2,510	412	344	297	2,259	2,556	8,600
7. Floor	0	0	39	923	1,039	0	1,039	4,600
Total	**20,181**	**24,175**	**4,178**	**2,233**	**5,098**	**23,000**	**28,098**	**110,300**
				7.9%				
Difference to Target	**963**						**748**	**8800**
Change from previous	334	105	-153	81	217	0	217	900

Program Area	**19,218**					**23,000**	**27,350**	**101,500**
Design/Target	105.0%					100.0%	102.7%	108.7%

Gross/Program Ratio	1.39
Gross/Rentable Ratio	1.16

Garage	Program Area	Gross Area
Lower Basement	4,097	4,396
Upper Basement	4,088	4,414
Total	**8,184**	**8,810**

The project was a success story in Finland. According to the Owner's Project Manager: "*Still today, over 9 years later, ICL Headquarters is the only project where I have got practically real-time information comparing actual areas to the building program on a detailed level, and was able to follow constantly that the project design stayed within the allocated limits.*"

This example demonstrates the efficiency of an automated link between the *Space Program* and *Design Model*, and the potential of a *Requirements Model* for project management to help design to meet the spatial and cost goals. Even a simple implementation of the area *Requirements* linked to the *Design Model* provided a concrete improvement compared to the traditional methods.

7.1.2 Clark Center

The Clark Center is a new landmark building at Stanford University. The basic idea was to create synergy by creating a laboratory building based on an open concept for several disciplines. The building was designed by Sir Norman Foster in association with MBT Architects, and it has received widely praising comments in the public [*Clark Center 2004* [93]]. It is obvious that in many respects the building is very well designed and built.

However, even this remarkable building demonstrates the problems of managing detailed *Client Requirements* in the process. The interior of laboratory *Spaces* have several details which are not satisfying the functional needs of the end-users of the *Spaces*. The following examples are based on Dr. Alfred M. Spormann's [*Spormann, 2004* [94]] interview in November 2004, about one year after the completion of the building. Dr. Spormann represented his laboratory in the design and construction phase.

The approach in the interior design was top-down; the end-users were not consulted in the beginning of the process. Instead, they got three basic laboratory concepts to select from, and only some small adjustments to the basic concepts were allowed. The main constraint was the area per research group, and also the number of laboratory benches was predefined. They were not asked how they work; rather they were asked, "will this work?"

Flexibility was the main goal; everything in the laboratory is on wheels. According to Dr. Spormann, this was "*an expensive, but probably good solution*." My observation in the building was that the sinks are not movable (which of course would have been technically very difficult to solve), and thus, they limit the possibilities to move furniture. In addition, this end-user's use of the form "*probably*" indicates that the flexibility has not been utilized, at least not during the first year. This raises the question of the investment priorities: Would the end-user have rather used the available money to correct some of the short-comings described in the problem examples (Section 7.1.2.1) than have the movable furniture? According to Dr. Spormann, this question was never discussed during the design and construction process.

Another expensive building solution was the vibration control in the basement intended for high-accuracy laser equipment. This large investment was not utilized because the intended user did not move into the building. Dr. Spormann's comments related to this issue were that the assessment of the design was very difficult in the early phases of the project, and this lead to the lack of commitment and long uncertainty about the neighbors, which could have totally different needs. All this led to the situation, where some of the available funding was used pointlessly, and in other places budget constraints cut some necessary details from the design. All this could have been improved if the end-users had been given a better understanding of the design, its relation to their *Requirements* and the *Requirements* of others and the related tradeoffs in all phases. The end-users should be able to participate in the priorization throughout the process from the beginning.

The detailed end-user *Requirements* were not followed. Dr. Spormann's estimation was that only half of the end-user *Requirements* were actually implemented, and that it was totally impossible to check the design accurately enough to understand what was left out during the process. The *Requirements* were not recorded in the design documents, and many solution details were "*hidden*" in the complexity of drawings. In addition, the promised end-user budget never came true, which was supposed to help fix possible shortcomings. The end result of the process was, in Dr. Spormann's words, "*We felt that we were betrayed.*"

"Value engineering" was a big problem; the priorities were defined by the facility management people, not by end-users, and the cost cuts affected crucial functionalities. The end users should be able to participate in the decision making when the choices and trade-offs are made. In most cases end-users had no clue why, when or who made the changes compared to their *Requirements*. There was a lack of a distinct organization role advocating the end-user needs in the process.

7.1.2.1 Problem examples and their relation to the Requirements Model

The next 5 problem examples, CC1–CC5, are documented from Dr. Spormann's interview and the following comments illustrate how my *Requirements Model* could improve the process if implemented to a *Requirements Management* application. The problem examples are real, but the solution examples are hypothetical.

Example CC1: The interior architect designed black furniture despite the end-users' opposite *Requirement*. The black furniture is not suitable from a functional viewpoint. There is not enough light to work comfortably in the laboratory and the high contrast is stressful for the eyes (Figure 76). The problem is even worse because also all the task lights were "value engineered," i.e., removed, during the process. Only half of the intended lighting is in place according to the lighting expert who was defining the original lighting *Requirements*. In practice the end-users try to solve the problem by covering the tables (Figure 77), but this does not solve the problem of black selves.

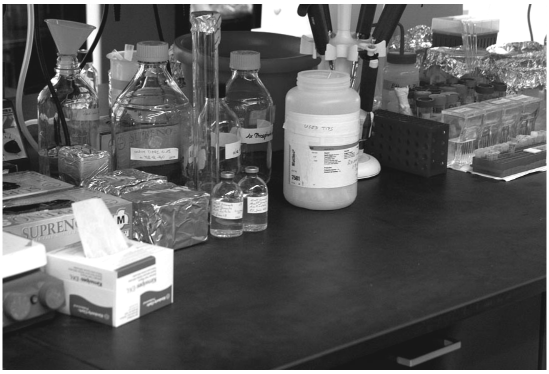

Figure 76: Example CC1: Black table top

Figure 77: Example CC1: Table top covered with white material by the end-users

CC1, Relation to the Requirements Model:

If the color *Requirements* would be visible in the *Requirements Management* interface of the design software, the designer would immediately see them. If he then disagrees with the *Requirement*, he can record the design intent in the *Requirements Model*, and thus make the conflicting issue visible to the other participants, including the end-users, who could also use the *Requirements Management* application (Figure 78). If the end-users will not accept the conflict between *Requirements* intent and design intent, the conflict must be decided by the project manager based on the project's priorities; does the color of the furniture have functional effects, and if it does, are architectural issues more important than the usability of the laboratory?

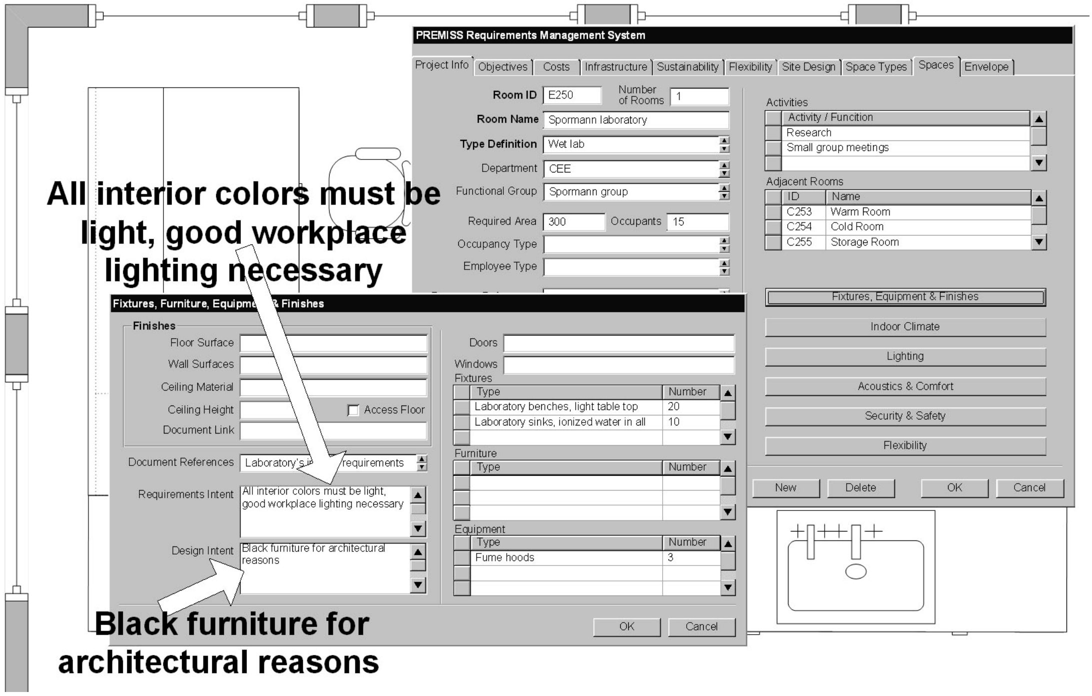

Figure 78: Example CC1: Visible conflict between *Requirements* intent and design intent

Example CC2: The laboratory made a definite *Requirement* to have ionized water in every sink because it is essential to most of the laboratory's intended research functions. When they moved in, they found out that only 1/3 of the sinks had the required taps. The change was never communicated to them. However, this affects the everyday processes in the laboratory. Figure 79 represents the expected situation, and Figure 80 the current situation in 2/3 of the sinks. The necessary ionized water is in large bottles which occupy valuable desk space and have to be constantly filled. Both are distractions from the core activities in the laboratory. Leaving out the required taps saved some costs, but because the laboratory already has the ionized water system, the savings are hardly significant, and not justified compared to the difficulties the lack of required equipment now causes to the end-users.

Figure 79: Example CC2: Sink with ionized water

Figure 80: Example CC2: Sink without ionized water

CC2, Relation to the Requirements Model:

The basic logic of making the *Requirement* visible to the designer is similar with CC1 (Figure 81), but in this case, instead of making a design decision against the end-users' explicit functional need, the situation should be negotiated, and, if the change is acceptable, the *Requirement* updated instead of just recording a different design intent.

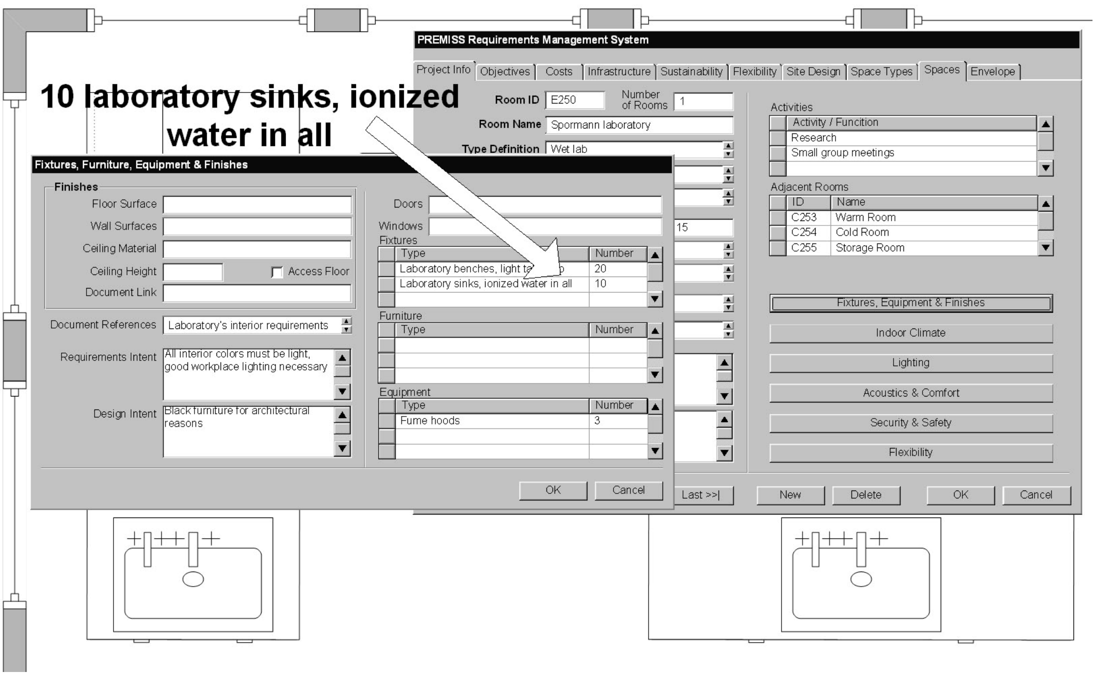

Figure 81: Example CC2: Ionized water *Requirement*

Example CC3: In the gas storage room the rack system is bolted to the floor so that it is impossible to use trolleys to move the gas containers (Figure 82). Placing the clamps behind the rack would have solved the problem easily and without any additional costs if the designer had been aware of the *Requirement*, (Figure 83, made by image processing).

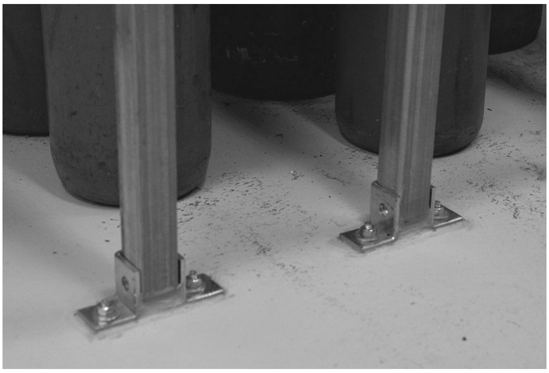

Figure 82: Example CC3: Attachments of gas container racks

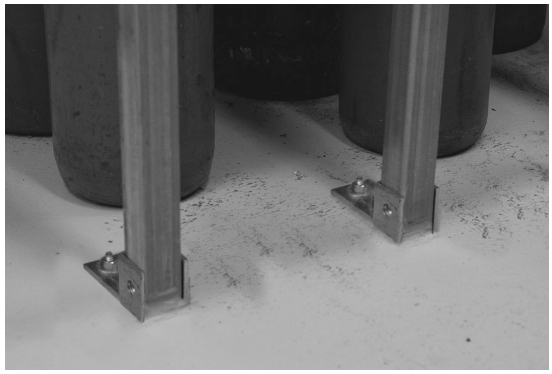

Figure 83: Example CC3: How the attachment should have been

CC3, Relation to the Requirements Model:

This type of problem is very common. One project can contain thousands of small detailed *Requirements*, which are very difficult to find or remember during the detailed design. If the *Requirements* would be linked to the *Design Model* so that the designer could see them without need to go through the documentation, it would significantly improve the chances of finding the relevant information (Figure 84). In this case there could not have been any financial or architectural reasons to ignore the *Requirement*, which are possible reasons in examples CC1 and CC2.

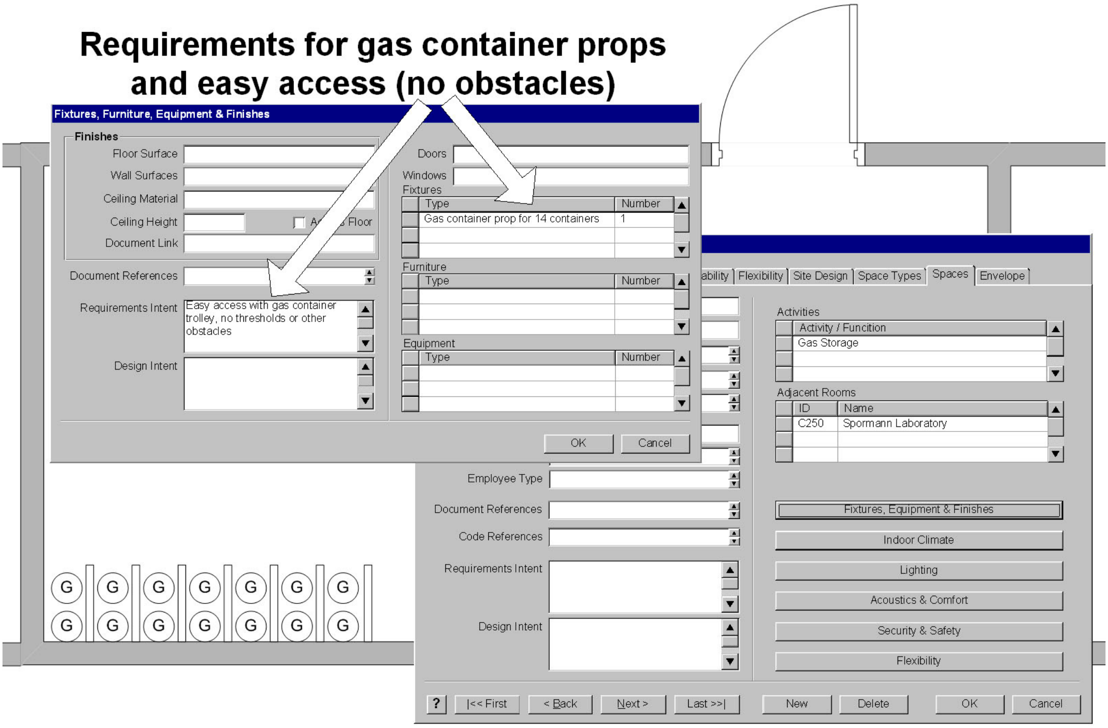

Figure 84: Example CC3: Gas container racks

Example CC4: The temperature *Requirement* for the warm room, +30±0.5 °C, was not followed. Now the temperature fluctuates ±1.5 °C, which disrupts some experiments. In this case, it is difficult to say if this problem was caused by the incorrect implementation or by a missed *Requirement*.

CC4, Relation to the Requirements Model:

The temperature *Requirements* for the warm room were exceptional; both the temperature and the tolerances were unusual. In this case the *Requirements Model* can serve as a reminder of the unusual *Requirements* in the design and construction phases (Figure 85), but if the problem was caused by quality problems in the construction work, the system could only help to verify that the construction and MEP contractors had the correct information, and possibly force them to correct the situation afterwards.

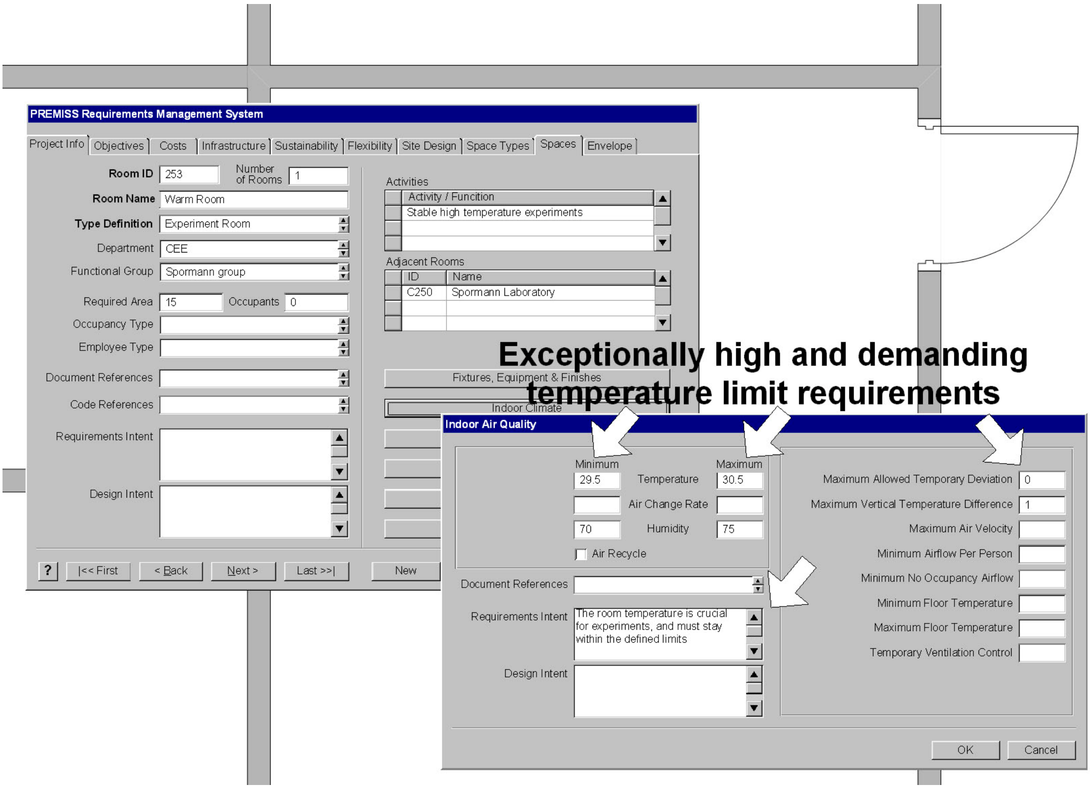

Figure 85: Example CC4: Warm room *Requirements*

Example CC5: A standard door without any threshold or sealing was used in the cold room in which the temperature is about 4 °C. The door mistake caused the cold air to flow to the neighbor room below the door and the door had to be corrected afterwards.

CC5, Relation to the Requirements Model:

This example shows a clear difference compared to traditional *Requirements Documentation*. By selecting a door and asking for its *Requirements*, the system can search the *Requirements* which are defined in the *Requirements Model Specification* to have effect on the *Bounding Elements* from both *Spaces* which the door connects. This search process is based on the *Building Product Model* capability to recognize relationships (Section 6.2.6). Then the system can show these *Requirements* to the end-user who can see if there is a need for a special solution (Figure 86). In this case the temperature differences on both sides of the door are so high that it is obvious that the door must have either a threshold or good sealing; a standard door was in this case a clear design mistake.

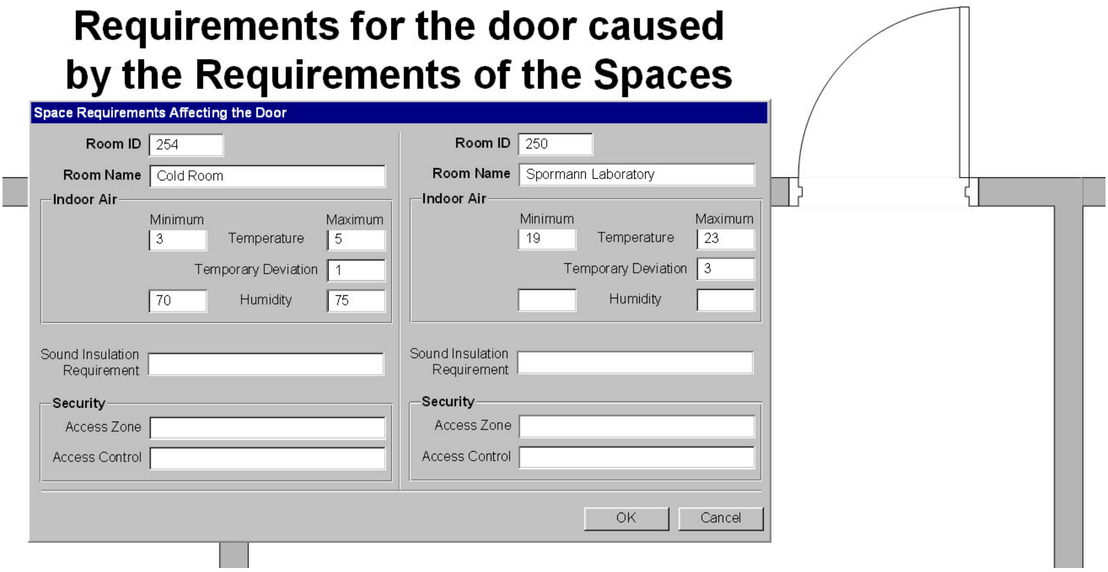

Figure 86: Example CC5: Cold room door

7.1.2.2 Conclusions from the Clark Center case:

Dr. Spormann's final comments were: "*You learn to live with what you have. The building concept and architecture are great, and the collaboration thanks to the openness has already improved our research quality. However, the mistakes and shortcomings in details are annoying, and in many places would have been avoidable, if the process would have been better managed. We should have been able to participate in the trade-off decisions and known what we will really get.*"

This case demonstrates (1) the shortcomings in the current process and (2) how important it is to have a clear documentation of *Requirements* and the possibility of comparing *Requirements* and design solutions in a way that is easily readable for the end users.

7.1.3 Laboratory Facility

In this case study a research team collaborated with the *Project Team* to construct a three-dimensional *Building Product Model* of a $100 million research laboratory facility [*Kam and Fischer, 2004* [95]]. They identified several design and integration problems, of which some are related to *Requirements Management*. Also in this case, the problem examples are real, but the solution examples are hypothetical.

Example LF1: The *Client* had a vertical proximity *Requirement* for teleconference rooms on different floors of the building. During the design process the spatial arrangements went through several iterations. In this process the teleconference rooms on different floors were moved to different places, and the end result ignored the vertical proximity *Requirement*. There were three main reasons for the mistake. (1) Designers work on each floor with separate drawings, and any connection to other floors is difficult to keep in mind if the connection is not obvious, such as the vertical connection of columns, shafts, elevators and staircases. (2) The vertical proximity *Requirement* was not recorded in the design documentation. (3) The *Project Team* worked under high schedule pressures and did not have enough resources for design coordination and to check the *Requirements* at every stage. Thus, the vertical proximity *Requirement* disappeared from the process.

LF1, Relation to the Requirements Model:

The connection of this problem to my *Requirements Management* system is similar to several Clark Center examples. If the *Requirement* would have been connected to the *Spaces*, it would have been visible to the *Project Team* (Figure 87). Thus, the likelihood of finding the problem in design coordination would have been higher. The case also emphasizes the importance of efficient *Requirements Management* tools in the current process where designers struggle with the time and resource problems.

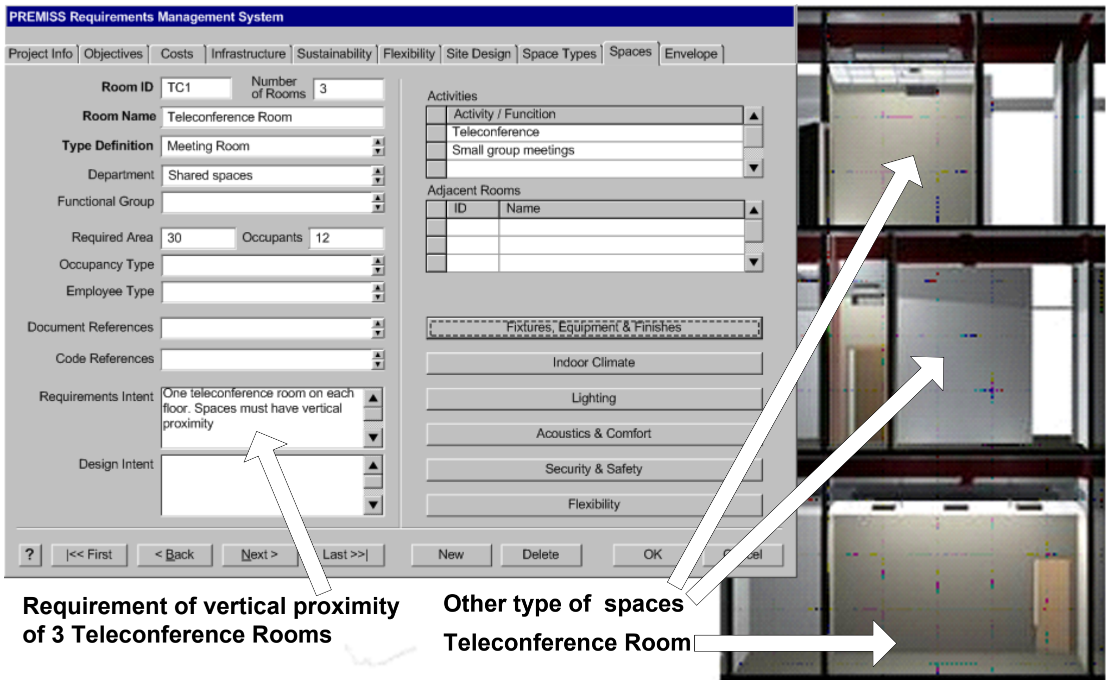

Figure 87: Example LF1: Vertical proximity of teleconference rooms

Example LF2: The *Client* added minimum distance *Requirements* for the telecommunication cable trays relative to the ceiling grid and structural elements during the design process. However, there was miscommunication within the *Client* organization, so the design team did not get the final set of *Requirements* from the telecommunication team until late in the process. When the *Requirements* finally reached the design team, the project could not afford the time and cost of the changes which would have demanded changing floor-to-floor height and structural member sizes to meet the *Requirements*. Thus the *Requirement* had to be ignored in several places. As in example 1, in this case the schedule and resource problems also affected the situation.

LF1, Relation to the Requirements Model:

In this case the connection to *Requirements Management* is related to poor communication within the *Client* organization. This example emphasizes the need to record all *Requirements* in a *Requirements Management* system instead of in different documents scattered in the organization. If the *Requirement* would have been recorded in a shared *Requirements Management System*, it would have been visible to the whole *Project Team* immediately when it was created, and there would have been time to adjust the design to meet the *Requirement*.

7.1.4 Two Facility Development Project Examples

7.1.4.1 Case FD1: Oak Grove

A facility developer agreed to preserve an oak grove at one corner of a property as one of the development approval terms with the city council. Several months down the facility development process, his building permit was rejected because his mechanical engineer submitted a building plan that routed the water supply piping system through this oak grove. The mechanical engineer, who was not aware of the preservation *Requirement*, located the piping route in that corner, because it was the *Location* for all major water intake points to the site. This mistake caused six months delay for the project [*Ibrahim and Paulson, 2004* [96]].

7.1.4.2 Case FD2: Play Structure

In another project, facility developers lost valuable operating revenues for 'forgetting' to deliver an agreed item. In this case, the funding program required a play structure in an affordable housing project. As the design progressed, the play structure was replaced by a flat playground area. A few years after the project completion, the funding agency fined the developer for not providing the required play structure. It also requested the property developer to build a new play structure or return the funds to the agency [*Ibrahim and Paulson, 2004* [97]].

7.1.4.3 Relation to the Requirements Model

Both examples show typical design mistakes; one of the members of the *Project Team* does not find a specific *Requirement* from the documentation, maybe does not even know that he should look for such information. Because of this missed information he makes a wrong decision which causes problems to the project. In case 1 the missed information is vegetation which is required to be preserved. In case 2 the missed *Requirement* is a site accessory which is part of the funding clauses. There are many reasons for these mistakes; amount and quality of information, lack of designer's resources and time, etc.

A *Requirements Management* application linked to the design tools could easily show the *Requirements* related to an object, which in both these cases is the site. Although these building types are not in the scope of my research, my *Requirements Model Specification* includes elements needed for both cases, as the hypothetical solution example of the case 1 problem in Figure 88 illustrates. In case 2 the example would be basically similar; the only difference would be that the *Requirement* would be in the “Site Amenities” category, and the source for the *Requirement* would have been the funding agency instead of the city council.

As described in Sections 6.3.2 and 6.4.2, such links from the *Requirements Model* to the *Design Model* can be fully automatic, which means that there is no additional work compared to the recording of *Requirements* in a normal document or database. An important feature in the *Requirements Model Specification* is that all *Requirements* have both the owner and source (Figure 88).

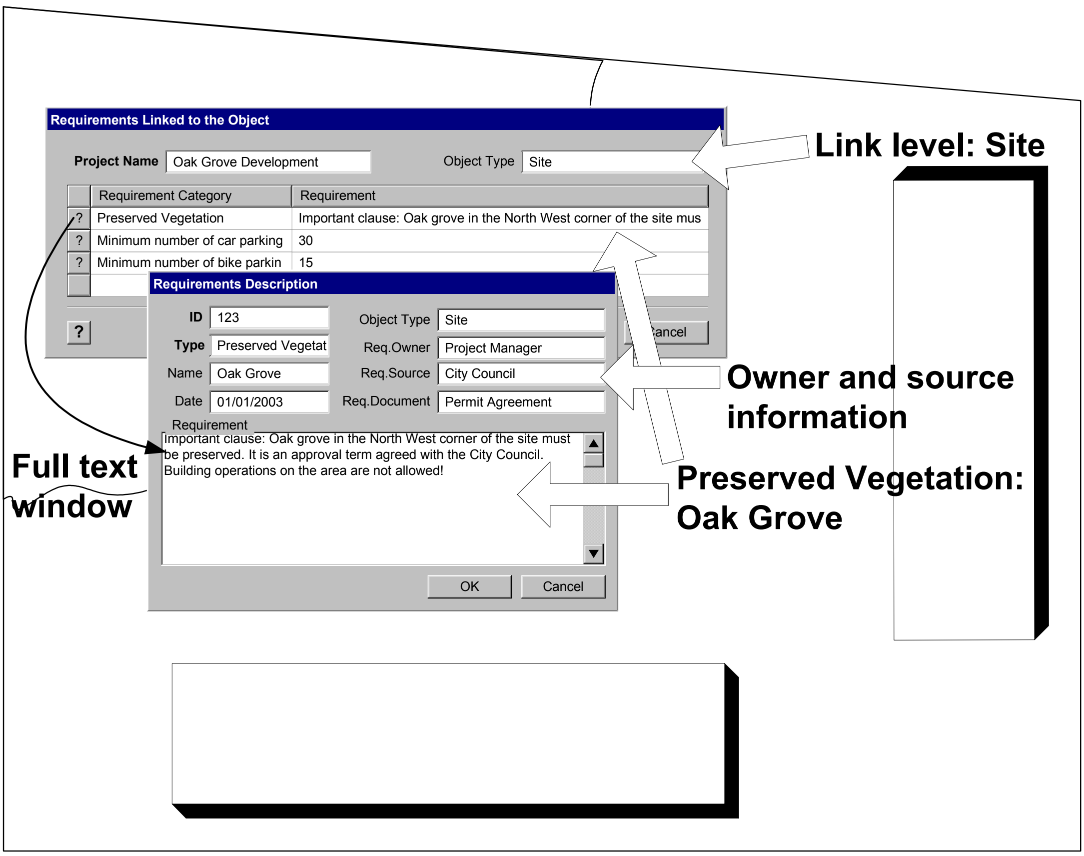

Figure 88: Information of the Oak Grove which must be preserved

7.1.5 Conclusions about Usefulness

Based on the case studies, interviews of several AEC industry experts [*Discussions and Interviews 2003* [98]] and my own design experience (Section 1.1), I argue that the main problems in the current *Requirements Management* relate to two main issues: (1) keeping the project within the total cost limits, e.g., managing the size and quality of the building(s), and (2) keeping the detailed *Requirements* in the minds of the many *Project Team* members during the design and construction process.

The ICL Headquarters project offers concrete evidence of the usefulness of the link between *Requirements* and *Design Models*. (1) The automated generation of *Space Objects* from the *Space Program*, (2) automatic calculation of areas in the drawings and (3) area information linkage back to the MS Excel spreadsheets were crucial factors in the successful project. Compared to a manual process to calculate the areas and collect the data to spreadsheets, the system saved several working days every week and improved the accuracy significantly. In this fast-track project, where the design was changing daily, even more important was the possibility of following the development of design in real-time. In a manual process, different areas and the total volume of the building, important control information, would have been available several days later, when the design had already changed.

In the detailed *Client Requirements*, the problem relates to the amount of data; there can be thousands of detailed *Requirements* in a large project, and they easily disappear in the process, as the examples from the Clark Center demonstrate. The design missed several details including some crucial *Client Requirements* or functional *Properties*. They provide concrete evidence of the problem identified in Section 1.1.3. The solution examples are based on the content of my *Requirements Model Specification*, and illustrate how a practical implementation of the *Requirements Model Specification* and link to design applications could improve the process.

The first example in the Laboratory Facility represents similar problems as the Clark Center examples, but it demonstrates problems with *Location Requirements*; especially that the vertical proximity *Requirements* are difficult to manage because of the usual working methods where the building is divided into different floors which are not often compared to each other. The second example, LF2, emphasizes the difficulty to record and communicate the *Requirements Changes* during the process, a problem which was identified also in the LCE project (Section 1.2.2).

The facility development examples FD1 and FD2 show a different aspect of the *Requirements Management* problem. In these cases the missed *Requirements* were not small details, but crucial conditions which caused significant costs to the project when they were missed. Examples like these are not unusual; they can easily happen, especially when the project participants change. As documented in Sections 1.2.2 and 1.1.3.2, in the current process a significant part of the information is tacit knowledge, and even if the information is explicitly recorded, the knowledge of its existence, origin and location is mostly tacit. The power of a *Requirements Management* system which would link the *Requirements* to the *Design Models* is largely in making this explicit, but "hidden," information visible to the whole *Project Team*.

7.2 Generality of the Requirements Model Specification

The examples in Sections 1.2 and 7.1 demonstrate some of the identified problems (Section 1.1.3), and the *Requirements Model Specification* includes potential solutions for each of them. It also includes all elements which were necessary for the ICL Headquarters project's successful project management. The *Specification* covers 300 *Requirements* in 14 main and 35 sub-categories (Appendix B3, Table 15). It is based on a synthesis of two large, widely used *Requirements Hierarchies* (Section 3.2.2), analysis of *Requirements* in five *Building Programs* (Chapter 4) and *Spatial Requirements* in the current *IFC Specifications* (Section 6.2).

These *Requirements* are organized in the *Specification* into 7 main-level and 30 sub-level *Requirements Objects* which have direct links to 5 levels of detail and 2 systems in the *Building Product Model* plus indirect links to 4 levels of detail and 12 systems (Section 6.3, Appendix B2, Table 14). In addition, each *Requirements Object* can be extended with project-specific attributes using the *Property Set* mechanism, which is part of the *IFC Specifications*.

The *Requirements Model Specification* is formally defined as an extension of the current *IFC Specifications*, because the *IFC Specifications* are both official and de-facto standards for *Building Product Models* (Section 3.4). However, the principles of the *Requirements Model* are not limited to the IFC environment. A similar linkage between the *Requirements and Design Models* could be implemented in any application which is able to identify the targets for the defined *Requirements*; project, site, building, story, *Space* and different systems.

As documented in Section 6.1.5, the content of a *Requirements Model Specification* is an issue which can be discussed indefinitely. There is no "correct" answer, because the needs in different projects are inevitably different. Thus, it is impossible to claim that the *Specification* covers every possible *Requirement* for buildings in my research scope (Section 1.3). However, based on the analysis, I argue that it covers a reasonable part of the identified problem.

7.3 Implementability of the Requirements Model Specification

Implementation of the *Requirements Model Specification* is not in the scope of my Ph.D. research. Thus, the implementability of the *Specification* must be based on indirect evidence, such as previous implementation of the link between *Requirements* and a *Design Model*, rapid prototyping, implementation of the other *Specifications* using similar of definitions, and expert evaluation of my *Specification*.

7.3.1 Software Application for ICL Headquarters

The first practical implementation of a link between a *Requirements Model* and *Design Model* known to me was used in the ICL Headquarters project (Sections 1.2.1, 7.1.1 and Appendix C, C1). The implementation had many differences compared to my *Requirements Model Specification:* (1) the application was not based on a formally defined *Specification*, (2) the *Requirements Model* was a simple MS Excel Spreadsheet, and (3) the only *Requirements* linked to the *Design Model* were the areas of the *Spaces*. However, the application demonstrated that such a link is implementable and has many benefits compared to the manual processes (Section 7.1.1).

7.3.2 Space Layout Editor

Jiri Hietanen implemented the same functionality which was used in the ICL Headquarters project in a product called Space Layout Editor. This product was based on MS Visio and use of IFC data exchange. In this application the *Space Program* was also a MS Excel file [Figure 89, *Hietanen, 2000* [99]].

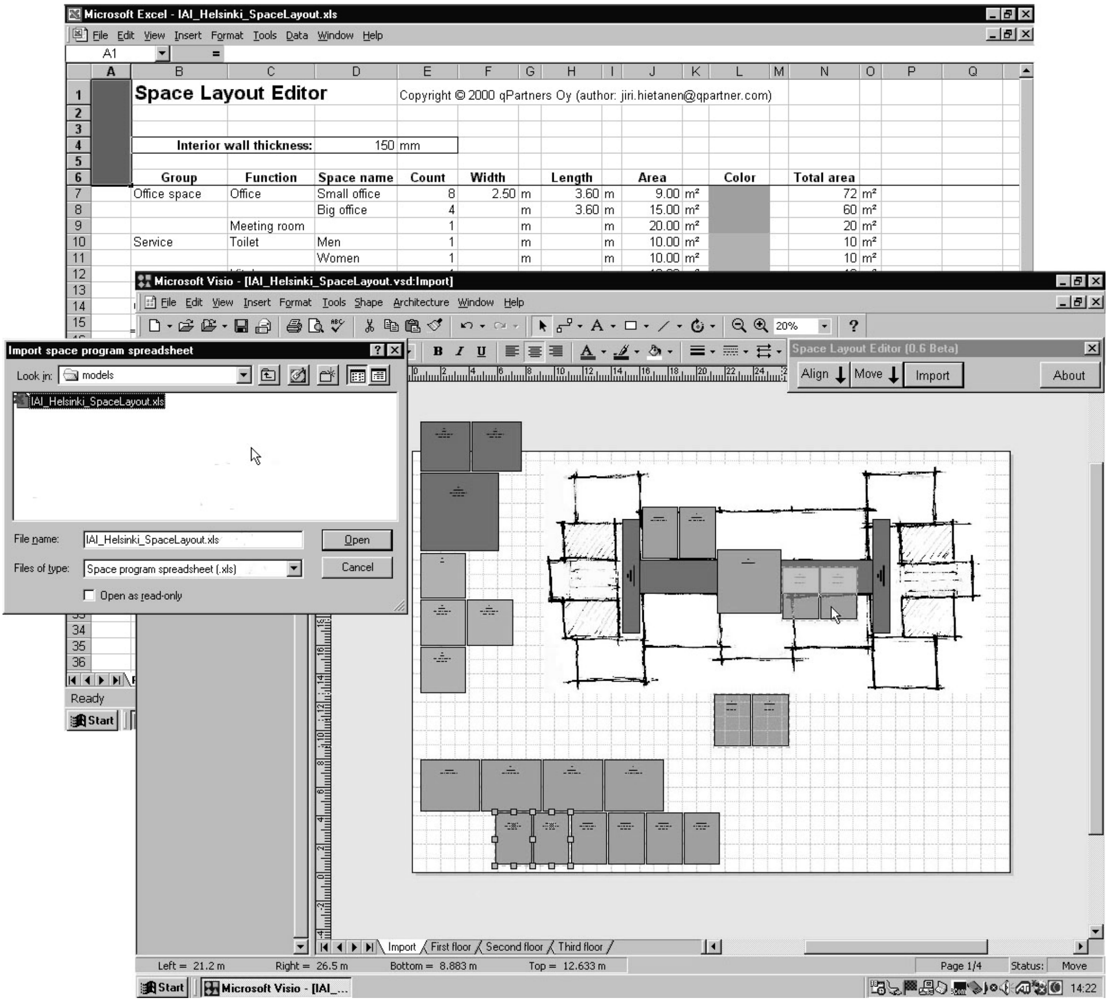

Figure 89: Space Layout Editor, [© *Hietanen, 2000*]

The basic concept was the same as that used in the ICL project, but this application demonstrated the implementability of the *Requirements Model* based on *IFC Specifications* and in an object-oriented software environment. BLIS used the application as a part of presentations in many seminars around the world [*BLIS 2000* [100]], and it is an excellent demonstration of the possibilities of inter-operable software tools (Figure 90).

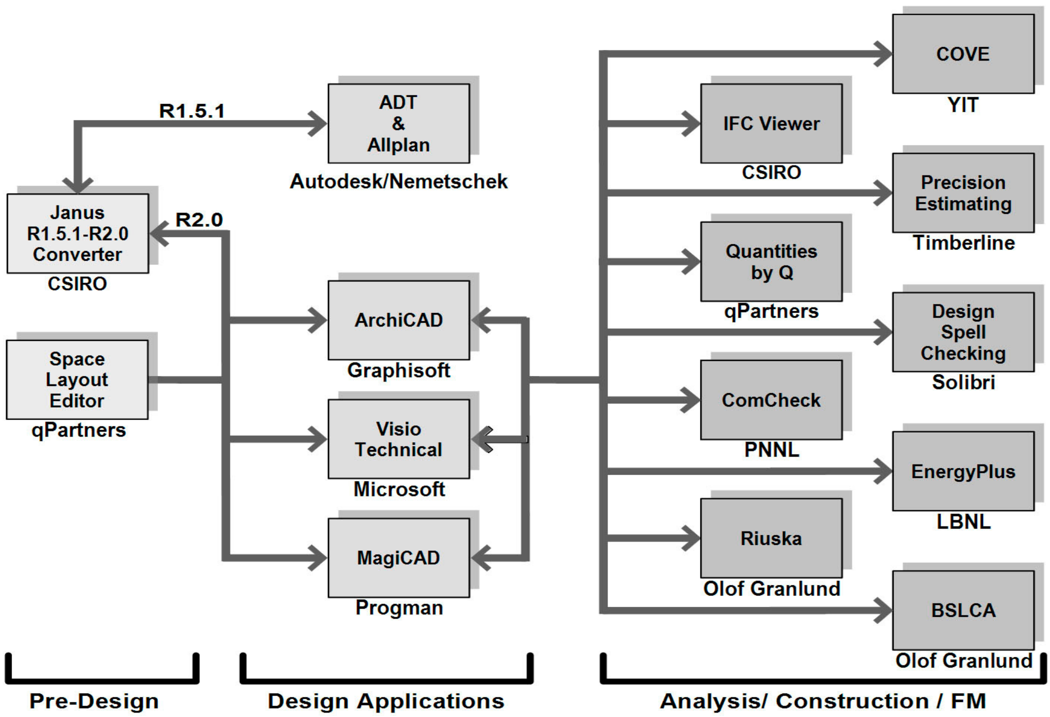

Figure 90: BLIS presentation overview [© *BLIS 2000*]

7.3.3 Rapid Prototyping

The rapid prototyping phase of my research demonstrated that the *Requirements Model* for *Space*-related *Client Requirements* is implementable also for *Requirements* other than area *Requirements* (Chapter 5).

7.3.4 Implementation of Other Specifications Using Similar Definitions

Depending on the source, there are slight differences in the numbers of IFC-compliant software products as of January 2005: According to the IAI Implementation web site [*IAI ISG 2004* [101]], there are 36 certified software products, 9 implementation toolboxes and 19 demonstrators, pre-releases or prototypes. The BLIS web site [*BLIS 2004* [102]] lists 49 end-user software products, 7 development component products and 4 developer application platforms. In addition, there are numerous demonstration and research implementations of the *IFC Specifications* not listed on these sites. In any case, there are many IFC-compliant software products on the market (December 2004) proving, that *IFC Specifications* are commonly implemented technology.

My *Requirements Model Specification* is based on the same formal language, EXPRESS, and the *Requirements Objects* are subclasses of existing IFC objects (Section 6.3). My *Requirements Objects* use (1) existing IFC elements, such as IfcDocumentReference and IfcApprovalStatus, and (2) existing data types, such as IfcLabel, IfcText and IfcAreaMeasure, defined in the *IFC Specifications*.

7.3.5 Expert Evaluation of the Requirements Model Specification

One of the Advising Committee members in my Ph.D. research, Dr. Vladimir Bazjanac is one of the leading experts in *Building Product Modeling*. Dr. Bazjanac has been one of the key persons in the IAI and the Chairman of IAI Technical Advisory Committee since IAI's foundation 1994. Dr. Bazjanac works at Lawrence Berkeley National Laboratory as a Staff Scientist testing and using IFC implementations both in research and in large building projects.

In addition, a group of world leading experts of *Building Product Models* kindly accepted the task to review the structure principles on my *Requirements Model Specification.* I asked them to specifically check my *Specification* (Chapter 6) and write their statement about its implementability. The group includes the following people (in the chronological order of their statements): Jiri Hietanen, Patrick Houbaux, Kari Karstila, Robin Drogemuller, and Richard See. Their statements and short curriculums are in Appendix D.

7.3.6 Conclusions about Implementability

Based on the arguments in Sections 7.3.1–7.3.5, my conclusion is that my *Requirements Model Specification* meets the implementability criterion.

7.4 Conclusions of the Validation

My conclusion from Sections 7.1, 7.2 and 7.3 is that the *Requirements Model Specification* presented in this thesis meets all three validation criteria, and that this expansion of the existing *IFC Specification* is a valid scientific and practical contribution. Chapter 8 summarizes these contributions in detail.

8 Summary of the Scientific Contributions, Practical Implications and Suggested Future Research

The goal of my Ph.D. research was to develop and validate a method to create an active link between *Requirements* and *Building-Product-Model*-based design applications. The purpose of this link is to improve the *Requirements Management* in the design process. The scope was limited to architectural design of office and laboratory buildings. However, I believe that many of the principles are also apply to other design domains and other building types.

The main scientific contributions of my research are a *Requirements Model Specification* and division of an instantiated *Building Product Model*, i.e., the data set of a project, into four main *Models.* In addition, my research documents the problem of *Requirements Management* of detailed *Client Requirements* in building projects and defines a *Requirements Hierarchy* for the basis of the *Requirements Model Specification*. Section 8.1 documents the scientific contributions in detail.

The major implications on a practical level are that (1) the *Requirements Model Specification* enables implementation of *Requirements Management* applications linked to *Building Product Models*, and that (2) the use of such applications can improve the *Requirements Management* in the design process. I also propose some improvements in the current *IFC Specifications*. Section 8.2 documents the practical implications in detail.

My research opens a wide range of future research issues. We need future research of *Requirements* for other AEC domains and building types. The methods to utilize the *Requirements History* are also an area for future research. In addition, my *Requirements Model Specification* can provide a basis for research on other topics, such as automated verification of design or semi-automated design applications. Section 8.3 documents the proposed future research topics in detail.

8.1 Scientific Contributions

The scientific contributions of my research are:

- Documentation of the *Requirements Management* problem related to detailed *Client Requirements* on building projects (Sections, 1.2, 7.1, and 8.1.1)
- Documentation and analysis of the different *Requirements* types based on five case studies and two major *Requirements Hierarchies* (Chapter 4 and Section 8.1.2)
- Conceptual division of an instantiated *Building Product Model*, i.e., the data set of a project, into four *Models; Requirements, Design, Production, and Maintenance Models* (Sections 5.1.1 and 8.1.3)
- Concept of *Requirements* related to the different levels of detail in *Building Product Models* (Sections 6.3.1 and 8.1.4)
- Identification of the special needs of *Space Requirements* (Sections 4.3 and 8.1.5)
- Concept of *Direct and Indirect Requirements* in *Building Product Models* (Sections, 5.1.1, 6.1.6, and 8.1.6)
- *Requirements Model Specification* based on the *Requirements* analysis and the concepts listed above (Sections 6.3 and 8.1.7). This *Requirements Model Specification* connects the abstract concepts to a concrete system of *Requirements* for building design.

8.1.1 Documentation of the Requirements Management Problem

A commonly known and recognized problem in the AEC industry is that there are *Requirements Management* problems throughout the design and construction process. However, the problem is not well documented. It is difficult to find projects where the *Requirements*, design decisions and changes during the process were systematically documented. In addition, the people who have been involved in the process are often unwilling to speak about the mistakes in the project. This is natural human behavior; we want to forget, definitely not emphasize, our mistakes. The end-users of buildings, who know what is missing, do often not know the reasons for the shortcomings; were the missing *Properties* never asked for, or did the *Requirements* disappear in the process?

My research documents three aspects of the *Requirements Management* problem. The first case (Section 1.2.2) documents some of the problems in current *Requirements Documentation*. The second case (Section 7.1.2) documents the end results of the current *Requirements Management*; how many of the detailed *Client Requirements* disappear during the process. The facility development cases (Section 7.1.4) document that not only details but sometimes also *Requirements* related to crucial approval or funding conditions can disappear, causing significant time and financial losses in the project.

8.1.2 Requirements Hierarchy

Section 6.1.2 documents three possible solutions for *Requirements Objects* which are the requisite for a *Requirements Model Specification*. The conclusion from the alternatives is that only a structured, reasonably large, predefined set of *Requirements* will enable a usable link between *Requirements and Design Models* (Section 6.1.5). A generic *Requirements Object* would be too difficult to use in practice (Section 6.1.3), and attaching *Requirements* directly to the *Design Objects* is not a feasible solution, for two reasons: (1) It would lead to extensive multiplication of the same *Requirements* in the *Design Model*, and (2) the *Design Objects* do not exist when the *Requirements Capturing* process starts (Section 6.1.4).

My *Requirements Hierarchy* is based on analysis of two existing *Requirements Hierarchies* and five *Building Programs* (Chapter 4). This *Requirements Hierarchy* can be organized based on the "traditional" functional categories, such as safety, lighting, and acoustical *Requirements* (Appendix B3, Table 15). Another way to organize the *Requirements Hierarchy* is according to the level of detail in the *Building Product Model*, such as project, site, building, story, *Space* and systems (Sections 6.3.1, 8.1.4 and Appendix B2, Table 14).

These two ways to classify *Requirements* enable several differently organized views of the *Requirements*, and this provides the basis for useful user-interfaces for *Requirements Management* (Section 8.2.2.2)

8.1.3 Requirements Design, Production, and Maintenance Models

The first main concept is the division of the *Integrated Project Information Model* into four related *Models: Requirements, Design, Production, and Maintenance* (Figure 23). However, it is important to emphasize the difference between an *Instantiated Model* and *Model Specification*. The *Requirements, Design, Production, and Maintenance Model Specifications* can be based on one *Specification*. The division is needed only in *Instantiated Models*, the data sets of projects.

Some previous research projects have recognized the problem of one integrated *Model* [*Kam and Fischer, 2002* [103]*, Haymaker et al., 2003* [104]], but, to my knowledge, the division has not been formalized earlier. This division is crucial for *Requirements Model* development for several reasons. The full documentation of the reasons is in Section 5.1.1, but the main reasons are:

- The data content and structure of these *Models* differ. For example, one *Space Program Instance* (Figure 24) can relate to a number of separate *Instances* with identical *Requirements* in the *Design, Production, and Maintenance Models*.
- Typically, a *Project Team* produces several alternative design proposals which all should meet the defined *Requirements*. Thus, having one *Requirements Model* linked to the alternative *Design Models* is a logical structure instead of multiplying the same *Requirements* to different design alternatives, which could easily lead to *Requirements Management* problems.
- The flexibility of the *Requirements Model Specification* is greater if the *Models* are separated and connected with a “thin” link, e.g., there is only one identifier in both *Models* connecting the *Requirements* and *Design Objects* (Section 6.3.2). Adding or removing *Requirements* in the *Requirements Model Specification* does not change the design applications.

- Another reason for the separation is to make the distinction between *Requirements* and *Properties* clear; for example sound insulation is a *Requirement* for a *Space* in the *Requirements Model* and a *Property* of the *Bounding Elements* in the *Design Model*.

Because of the different information content of different design and contractor domains, the *Design and Production Models* will need a further division into several *Models* (Section 5.1.1), but this issue is not in the scope of my research. I propose it as one of the future research topics (Section 8.3.1.5).

8.1.4 Requirements Related to Different Levels of Detail in Building Product Models

The second main concept for the *Requirements Model Specification* is the categorization of *Requirements* by the link level to correspond to the structure of *Building Product Models*. This concept is necessary to create a systematic way to connect *Requirements* and building objects.

The levels are the same as in the *IFC Specifications:* Project, Site, Building, Building Story, *Space* and Systems (Section 6.3.1). The *IFC Specifications* do not specify Systems; it defines them as an aggregation of their parts (Section 6.2.6.2): "*Organized combination of related parts within an AEC product, composed for a common purpose or function or to provide a service. System is essentially a functionally related aggregation of products*" [*IFC 2004k* [105]]. To be able to build the links to the systems, I have defined 12 systems (Section 6.2.6.3). Two of these systems relate to the architectural design: building envelope and circulation system, and my *Requirements Model Specification* includes *Direct Requirements* for them. The other 10 systems relate to the structural and technical systems that are not in the scope of my research. My *Requirements Model Specification* includes only *Indirect Requirements* for those systems.

As one of the practical implications, I propose that IAI standardizes the names for systems in buildings in the *IFC Specifications* (Section 8.2.4). *Direct Requirements* to other design domains are among the future research topics (Section 8.3.1.2)

8.1.5 Requirements for Spaces: Type and Instance

As identified in Chapter 4, *Requirements* for *Spaces* are the most commonly defined *Requirements* in *Building Programs*. This is quite obvious; *Spaces* are the reason for buildings. Efficient management of area *Requirements* is crucial for the management of the size of building, and because size is the most important single cost factor in a project, management of *Space Requirements* is a crucial success factor for projects (Section 7.1.1).

In addition, most detailed *Client Requirements* are related to the *Spaces*, and the number of these *Requirements* can be very high. Thus, these *Requirements* are one of the main problems for the *Requirements Management* in building projects (Section 7.1.2).

These issues were the reason to concentrate on the *Space Requirements* in the rapid prototyping phase of this research. The prototyping and development of the *Requirements Model Specification* highlighted one important difference compared to the other *Requirements* in the *Requirements Model Specification:*

The *Space Requirements* in typical office and laboratory buildings are *Cascading;* e.g., there is a number of *Requirements* which are shared by several *Spaces* (Section 6.1.7). However, these *Spaces* also have individual *Requirements* which are not shared. A practical example of this is that all office *Spaces* can share the indoor air quality, lighting and acoustical *Requirements*, but they do not share area *Requirements* and they are not related to the same department. Repeating the shared *Requirements* in all office *Spaces* is a problem for *Requirements Management*. In the *Requirements Capturing* phase, defining the same *Requirements* for all *Spaces* is laborious and error-prone, and later, if some shared *Requirements* change, the changes must be updated in many places. To manage this I have defined *Space Program Type* and *Space Program Instance* objects in the *Requirements Model Specification* (Sections 5.1.1 and 6.3.10). This enables more efficient management of these *Cascading Requirements*.

This finding has also practical implications (Section 8.2.2) and it also creates some future research topics (Section 8.3.2.4).

8.1.6 Direct and Indirect Requirements

The third main concept for the *Requirements Model Specification* is the identification of *Direct and Indirect Requirements* (Sections 5.1.1 and 6.1.6). This is a critical issue for *Requirements Management* in the AEC industry. Many *Requirements* are defined for an object but they also affect other objects in the building, for example, temperature *Requirements* for a *Room* affect the HVAC systems and *Bounding Elements* (Section 6.1.2, Appendix B2: Table 14 and Appendix B3: Table 15). This concept and its implications are of course known in design practice, but, as far as I know, are not formally documented in any *Requirements Hierarchy* or *Requirements Management* system for the AEC industry.

The notion of *Indirect Requirements* is critical for my *Requirements Model Specification*. It connects the *Requirements* to several levels in the *DPM Models*, and enables different structured views of the *Requirements* (Section 8.2.2.2 and Appendix B, Table 14 and Table 15).

8.1.7 Requirements Model Specification

The main contribution of my Ph.D. research is the *Requirements Model Specification* (Section 6.3), which is a synthesis of the analysis and concepts documented in Sections 8.1.1–8.1.6. This *Requirements Model Specification* (1) connects the abstract concepts to a concrete system of *Requirements* for building design, and (2) enables the implementation of these concepts in a functional *Requirements Management* system that connects the *Requirements* to the *Design Model* and can improve the *Requirements Management* in the process.

However, as documented in Section 6.1.5, the content of a *Requirements Model Specification* is an issue which can be discussed indefinitely. There is no “correct” answer, because the needs in different projects inevitably differ. In spite of this fact, I believe that my *Requirements Model Specification* is a useful framework for practical implementations; the content of the *Model* is sufficient for most building projects within the scope of my research (Section 7.2).

8.2 Practical Implications

The main practical implication of my Ph.D. research is that the *Requirements Model Specification* enables the development of *Requirements Management* software which can link the *Requirements* and design solutions and improve *Requirements Management* during the design and construction process. The practical implications can be divided into four groups:

- Process implications
- *Requirements Management* software development issues
- Implications for related software products
- Improvements in the *IFC Specifications*

8.2.1 Process Implications

Although the focus of my research is clearly technical, and its goal was to develop a *Requirements Model Specification*, the problems of *Requirements Management* are not just technical issues. The Clark Center case study (Section 7.1.2) pointed out many issues which are related to the project management and involvement of end-users of buildings in the design and decision-making process. If end-users cannot participate in the evaluation of alternatives and if they cannot decide on priorities when there is need to make trade-off decisions, the help of a *Requirements Management* system is limited. This issue is related to the difficulty predefining the importance of different *Requirements* in a way that could help the *Project Team* to know the priorities of the users. This is one of the proposed future research issues (Section 8.3.1.9).

However, the difficulty of weighting alternatives does not eliminate the value of a system that links *Requirements* to the design solutions. On the contrary, it emphasizes the need to record and manage the *Requirements* and compare them to the design solutions in a way which the end-users of buildings can understand. When contradictions between *Requirements* and solutions arise, the decisions of the necessary changes should be made in collaboration with the shareholders, and implementation of my *Requirements Model Specification* into practical tools can help designers and project managers to visualize and manage these problems.

The practical impacts of the *Requirements Model* and *Building Product Model* based AEC processes are discussed in Section 8.4.

8.2.2 Requirements Management Software Development Issues

Implementation into practical software tools is the mandatory step to have practical outcomes from my research. The *Requirements Model Specification* can serve as the basis for development of *Requirements Management* applications. These applications can implement the whole *Specification* or some part of it, such as *Space Requirements*. There are also several technologies which software developers can use for such applications; the connection to the design application can be based on, for example, IFC file exchange, IFC-XML, or *Model Server* technologies (Appendix C, C2). This Section documents some implementation issues which came up during my research, especially in the rapid prototyping phase.

8.2.2.1 Space Program Type and Instance in Data Base Structures

Sections 5.1.1 and 6.1.7 document the concept of *Space Program Type* (*SPT*) and *Space Program Instance (SPI)*. In rapid prototyping, I based the database structure on this concept, but the way I implemented it was not optimal (Chapter 5). During the development of the *Requirements Model Specification* the implementation of this concept became clear, and the database structure is documented in Section 5.5. An important improvement was the concept of a *Virtual SPT*, an individual *SPT* for each *SPI* which is not based on some actual *SPT* definition. *Virtual SPT* simplifies the database structure significantly (Figure 31).

As the LCE case study demonstrated, one of the problems in the *Requirements Documentation* is the incoherent way to describe the same *Requirements* (Section 1.2.2.3). The use of *Space Program Types* and *Cascading Requirements* decreases this risk, because the same *Requirements* are not repeated as often as if they would be defined for each *Space Program Instance*. However, some *Requirements* are still used repeatedly. Thus, in the rapid prototyping database I used enumerations instead of free text fields (Section 5.2). Based on my analysis, I recommend this method for *Requirements Management* software implementation. One of its implications is, however, that end-users must be able to create new enumerations easily when they populate a project's *Requirements Model*.

8.2.2.2 Some User-interface Issues

In practice a well designed user-interface (UI) is important for managing large numbers of different *Requirements* and to focus on a set of *Requirements* that is meaningful from the viewpoint of a particular task.

The structure in my *Requirements Model Specification* enables a meaningful connection between *Requirements and Design Models*. The same structure can be useful also for some tasks in the design process, where the information needed is related to the object hierarchy in the *Design Model*. Thus, the different sets of relevant information can be the same as the groups in the *Requirements Model Specification*. *Space Requirements* are one example of this; their UI can follow the logical structure of the *Specification* (Section 7.1.2.1). For *Space Requirements* the structure also supports the *Requirements Capturing* process.

However, even in the design process, the *Requirements* needed for a task do not always follow this structure. Illustrations of such cases are detailed design of doors or decisions on partition wall types, in which the necessary data are aggregated from the *Requirements Model*. The information shown to the user must include the relevant data from the *Spaces* on both sides of the *Bounding Element*. In these examples the relevant set of information would consist of temperature, sound insulation and security *Requirements* of both *Spaces* (Figure 86).

In the same way, software developers can build UI to show any relevant aggregation or subset of the *Requirements Model* for *Requirements Capturing*, *Requirements Management*, design, or quality control tasks. There is no need to change the *Requirements Model Specification* to enable different views. The data structure connects the *Requirements* to the *DPM Models*.

One possible approach to manage a large set of *Requirements* is a *Requirements Template*, where users could define a meaningful subset of *Requirements* for their different project types and the UI would show only the defined subset. An example of this could be a *Requirements Template* for standard office buildings where many of the indoor air quality and lighting *Requirements* might not be relevant.

Another possible approach for the *Requirements Management* UI is LBNL's Design Intent Tool (Section 3.3.2). However, UI development is not in the scope of my research, and it is not just an implementation issue. It is one of the proposals for future research (Section 8.3.2.3).

8.2.3 Implications to Related Software Products

8.2.3.1 Requirements Capturing and Management Software

My *Requirements Model Specification* can be implemented in a *Requirements Management* application without the connection to design applications. The structured content enables different views of the *Requirements* and can help find *Requirements* to some specific issue and to document the evolution of *Requirements*. In that case the solution would be similar in principle to LBNL's Design Intent Tool (Section 3.3.2). However, the main benefits come with the connection to the design application (Section 8.2.3.2).

Requirements Capturing applications are used in the beginning of the *Requirements Management* process. The point of departure for my research was that the *Requirements Capturing* process has already produced a documented set of *Requirements*. In practice, the *Requirements Capturing and Management* applications should be connected, preferably as one product. Some *Requirements Capturing* applications could be developed to connect to my *Requirements Model Specification*, as the following examples illustrate. The integration of some *Requirements Capturing and Management* application to my *Requirements Model Specification* is one of the proposed future research topics (Section 8.3.2.5).

Connection to EcoProp

Because I based the *Requirements Hierarchy* of my *Requirements Model Specification* on the hierarchy in EcoProp, EcoProp could relatively easily be modified to use the structure of my *Requirements Model Specification*. For most *Requirements* the only needed change would be to add the correct link level to the *Design Model*. This change would not affect the use of the system, because the user does not have to make decisions, or even to know, on which level the *Requirements* are linked to the *Design Model*. *Space Requirements* are an exception. As documented in Section 3.2.2.2, many *Requirements* in EcoProp are defined now on the building level, although they are in fact *Space* specific.

There are two possible scenarios for the development of *Space Requirements* in EcoProp. The first scenario would include only the possibility of defining *Space Program Type Requirements* (Section 6.3.10.2) in EcoProp. After this phase the *Requirements Capturing* would continue to define a detailed *Space Program* using some other *Requirements Management* application which would add detailed *Requirements*, such as areas and number of *Spaces*, utilizing the *Space Type* definitions made in EcoProp. The other scenario also would include the tools to develop detailed *Space Programs* in the EcoProp system.

Connection to CRPM, Client Requirements Processing Model

As documented in Section 3.2.1, the CRPM system did not provide a useful *Requirements Hierarchy* as the point of departure for my research, but its Tertiary level *Requirements* could connect to my *Requirements Model Specification*. Possibly the *Requirements* on CRPM's primary and secondary level also could be connected to the project and building levels in the *Requirements Model Specification*.

Connection to the Serviceability Tools of ICF

The main purpose of the Whole Building Functionality and Serviceability (WBFS) system of ICF is to evaluate and rate existing buildings (Section 3.2.2.1). Thus, it might not even be relevant to discuss whether or not my *Requirements Model Specification* is applicable to the WBFS system. However, I believe that the

WBFS system could use the *Specification* by adding the Occupant Requirement Scale information to the appropriate RequirementsIntent fields in the *Requirements Model*. This information could then be utilized as a basis to break the verbal descriptions down to more detailed and specific *Requirements* in some other *Requirements Management* system. This issue is also one of the proposed future research topics (Section 8.3.1.7).

8.2.3.2 Design Software

Requirements Management applications based on my *Requirements Model Specification* are not “islands”; the main benefits of the system can be achieved only in connection with some design applications.

The first level of such a connection can be an application creating the “spatial skeleton” (Sections 7.1.1 and 7.3.2) and linking area information, and possibly also other *Space Requirements*, between the *Requirements Management* and design applications.

The second level would require a more sophisticated connection to which the *Model Server* technology provides the most promising platform, although other technical solutions are also possible (Section 3.4 and Appendix C, C2). The main issue on the second level is that design applications should be able to show the *Requirements* directly in their UI. All UI examples in Sections 5.6 and 7.1 are based on the second-level functionality.

8.2.4 Improvements in the IFC Specifications

My research addresses the need for some minor corrections and two additions in the current *IFC Specifications*. Integrating my *Requirements Model Specification* into the *IFC Specifications* would be of course a major change.

8.2.4.1 Proposed Corrections in the IFC Specifications

The current *Property Sets* in the IfcSpace and IfcSpaceProgram objects are incoherent and include several overlapping *Requirements* (Section 6.2.2.4).

As a short-term correction, I propose that in the next version of the *IFC Specifications*

- The overlapping definitions be removed, and
- All *Space*-related *Requirements* in the *IFC Specification* be placed into the IfcSpaceProgram entity, grouped into four categories: Common, Thermal, Lighting, and Fire Safety *Requirements*.

In the long-term, I believe that the correct solution is to have a systematic set of *Requirements Objects* in the *IFC Specifications*. I propose my *Requirements Model Specification* (Section 6.3) as the basis for this future IFC development.

8.2.4.2 Proposed Addition in the IFC Specifications

The methods to link objects in one *Model* cannot be used to link objects in different *Models*. The current *IFC Specifications* do not address this problem adequately. However, the need to separate the *Models* is crucial for future IFC development (Section 5.1.1). The need is addressed not only for *Requirements* in this research, but also for the link between the *Design and Production Models*. At least one previous research project [*Kam and Fischer, 2002* [106]] addressed the problem of one integrated *Model* in data exchange by pointing out the different content and structure of different design domains, although they did not propose a solution for the problem.

I have defined a new method for the link between objects in different *Models*. The link is based on two new subtypes. The new subtype of IfcExternalReference, IfcExternalObjectReference, would include the information of the type and location of the referenced *Model* and the link to a specific object in that *Model*. The new subtype of IfcRelAssociates, IfcRelAssociatesExternalObject, would enable the association of this reference to the objects inside a *Model* (Section 6.3.2).

John Haymaker addressed this problem in his Ph.D. research and proposed a solution based on "perspectors" [*Haymaker et al., 2003* [107]]. I propose the use of "perspectors" with my linking mechanism as one of the future research topics (Section 8.3.1.6).

All *Requirements Objects* in my *Requirements Model Specification* build on a new subtype of IfcControl, i.e., NewRequirement. It also includes a new subtype of IfcRoot, NewRequirementElement (Sections 6.3.3 and 6.3.4), which enables the recording of the owner and source of each individual *Requirement*.

Related to this proposed addition, the naming of different *Models* (6.3.2) and building systems (6.2.6.3) also should be standardized in the *IFC Specification*.

8.3 Suggested Future Research

My research opens several future research topics. In general they fall into two categories:

- Research which expands the *Requirements Model Specification*; for example, the relationship between high-level strategic owner *Requirements* and detailed end-user *Requirements*, *Requirements* for other design domains, other parts of the process, different building types, etc.
- Research which relates to the use of the *Requirements Model*; for example, user-interface issues, use of *Requirements History*, automated verification of design, semi-automated design software, etc.

The prioritization of the future research topics depends naturally on each reader's own area of interests, but in my opinion the most important topics are (1) implementation of a multi-model environment, *Requirements and Design Models*, using *Model Server* technology and (2) the use of the *Requirements Management* tools on real projects (Section 8.3.2.1).

8.3.1 Expansions of the Requirements Model

8.3.1.1 Other Requirements for Building Projects

My main research focus was on *Client Requirements*, but I also discussed *External Requirements* briefly. My *Requirements Model Specification* has links (Sections 6.2.5 and 6.3.7) to building codes, site regulations and community *Requirements* (Section 3.2.2.3), but all these topics need also further research.

In addition there are many different types of *Requirements*, for example, process and organizational *Requirements* for building projects, such as schedules and workflow management of the design and construction process. What is the relation of these *Requirements* to *Building Product Models*? Is there a need for a *Requirements Model Specification* and/or link to the *Building Product Model* in these *Requirements*? Or do the current 4D tools and/or the IfcConstraint already cover these issues sufficiently?

8.3.1.2 Other Design Domains

As described in Section 3.3, the designers' role in defining detailed *Requirements* for technical and structural systems is more dominant than in architectural design. Research in this area would provide another view of *Requirements Management* in the AEC industry.

My *Requirements Model Specification* links a wide variety of *Requirements* for architectural design to the *Design Model* and identifies several connections to systems in the building (Sections 6.2.6.2 and 6.3). These *Requirements* are often defined in the *Building Program* in connection to the *Space Requirements*, and thus they are documented as part of my *Specification*. However, some of these *Requirements* may fit better to HVAC, electrical, structural or other engineering domain-specific *Requirements Models*, such as load capacity, service life and flexibility of technical systems, security *Requirements*, etc. This and the more detailed content of the technical *Requirements* and the formal link between these *Requirements* and *Design, Production, and Maintenance Models* is one of the topics for further research.

8.3.1.3 Other Process Phases

As defined in Section 1.3, the scope of my research covers only a short period of the building life cycle process, design. The use of the *Requirements Model* in other parts of the process, such as construction and FM, is not covered in detail, though the same basic concepts apply. However, the *Requirements Hierarchy* and the user-interface implementation for *Requirements Management* would probably need some modifications. These are potential topics for future research.

8.3.1.4 Other Building Types and Infrastructure Construction

As defined in Section 1.3, the scope of my research only covers a few building types: office and laboratory buildings. The same *Conceptual Model* applies to any building, but because of the different *Requirements*, the *Requirements Hierarchy* and the user-interface implementation for *Requirements Management* would probably need some modifications; all are topics for future research.

This topic also can expand to other types of construction work, such as bridges, roads, railroads, and power lines. Is there a need to develop *Requirements Management* in these projects? Would the concepts documented in this research be applicable to these domains?

8.3.1.5 Division of Design and Production Models

As mentioned in the Sections 5.1.1 and 6.3.2, the logical extension of dividing the instantiated *Model* of a project into *Requirements, Design, Production, and Maintenance Models* is that the *Design and Production Models* would be divided into several domain *Models*. This opens up several issues for future research: What is the information content of such *Models?* How they should be linked to each other? Is the linking mechanism in my *Requirements Model Specification* applicable for this purpose, or does it need some further development?

8.3.1.6 Links between Different Models

John Haymaker's Ph.D. research presented the idea of "perspectors," which are a generic reasoning mechanism that analyzes "source perspectives" (different domain-specific representations) to produce one "dependent perspective," a representation which is dependent on the source representations [*Haymaker et al., 2003* [108]]. One example of this in the manual process is how a production drawing is derived from the drawings of different design domains; it is a new representation that depends on the information in the source drawings.

This issue closely relates to the division of the *Instantiated Model* of a project into *Requirements, Design, Production, and Maintenance Models*. How could the concept of "perspectors" apply to the link which is defined in my *Requirements Model Specification* (Section 6.3.2)? Are there needs for a dynamic link between *Models*? For example, how is the change recognized and updated in architectural *Design Model* and *Production Models* when the structures in the structural *Model* change? Could the *Model Server* technology address this problem or is agent technology a better solution for dynamic links (Sections 8.3.2.1 and 8.3.2.2)?

8.3.1.7 Linking Strategic Requirements to Detailed Requirements

Some *Requirements Hierarchies* are based on building owners' strategic *Requirements*, and their viewpoint is focusing on asset portfolios; an example of such a *Requirements Hierarchy* is WBFS (Section 3.2.2.1). This strategic viewpoint is naturally crucial for building projects, and the building design should meet these *Requirements* as well as the detailed end-users' needs. My research has several elements which are on the project level. The project is also the main level for the strategic *Requirements*, but the connection between the high-level view and details is not easy to recognize in many places. The CRPM system is trying to build a systematic chain from strategic *Requirements* to the detailed *Requirement Attributes* (Section 3.2.1). However, I believe that there is still a gap between these two views.

Calvin Kam's on-going Ph.D. research, "Decision Dashboard," may provide an innovative approach, which could connect different levels of *Requirements* into logical chains, help manage the *Requirements* and decision topics, and evaluate different design solutions [Figure 91, *Kam 2005* [109]].

Another possible approach could be a tool using some of the ideas in LBNL's Design Intent Tool (Section 3.3.2); a *"Requirements Intent"* tool which would enable to link different levels of *Requirements* to usable sets, and manage the *Requirements* on many levels. Section 8.2.3.1 describes another simple approach that is based on the idea of recording first high-level *Requirements Intent*, and then defining the details from the top down. Without further research it is difficult to tell how usable this approach would be in practice. Thus, I believe that future research in this area is still needed.

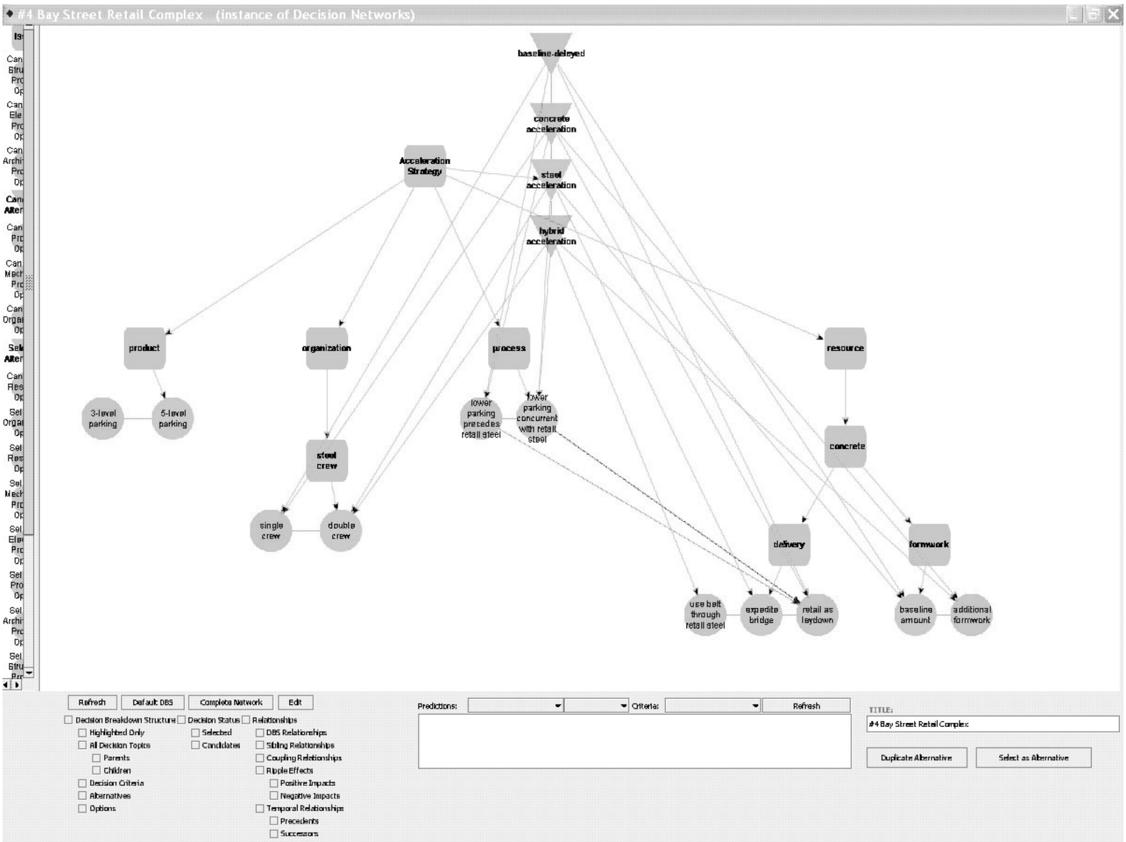

Figure 91: Decision Dashboard [© *Kam, 2005*]

8.3.1.8 Verification and Identification of Conflicts of "Fuzzy" Requirements

Section 8.3.2.7 describes a scenario of automated verification of "exact" *Requirements*, *Requirement Attributes*. However, in building design many *Requirements* are just verbal descriptions without any clear metrics, such as "Operations warrant a prestigious public lobby of the building, with top materials and condition, spacious, and very attractive" [*ICF, 2000* [110]]. In total 157, 52%, of the 300

Requirements in my *Specification* are in this category, which means that they are very common in building projects (Table 10). Verification of these “fuzzy” *Requirement Descriptions* is a totally different problem; verification of these *Requirements* demands human interaction. It is impossible to imagine how any checking system could verify, for example, if the design meets aesthetic *Requirements*, where clear metrics cannot be defined. However, designers or project managers could record in the *Requirements Model* that these *Requirement Descriptions* are met. This record could serve as a formal project history, and my *Requirements Model Specification* includes an element, ApprovalStatus, which can serve as a simple mechanism for recording verification on the *Requirements Object* level. However, it is an open question if a system should handle individual *Requirements*. What would the content and the correct granularity be?

It is possible that some *Requirements* are mutually exclusive. Can a *Requirements Model* provide benefits in identifying and managing such conflicts in *Requirement Descriptions*? My intuition is that different structured views of the *Requirements* can already help identify these problems by bringing them “close to each other” in some view, but this is an open research question. Is the problem similar to or different than verification of design solutions? Is it possible to use the same verification methods for *Requirement Descriptions* and *Requirement Attributes*?

8.3.1.9 Weighting the Requirements

Some *Requirements Capturing and Management* systems propose weighting systems for *Requirements* (Section 3.2.1). I have not included such a feature in my *Requirements Model Specification*. This choice was done based on my experience on real projects. It is difficult, in my opinion nearly impossible, to predefine the importance of different alternatives in a way which would be useful later in trade-off situations. The reason for this is that, based on my experience, the choice is only seldom between two *Requirements*. Usually the choices are between different designs which include or exclude several *Requirements*. The number of such combinations in a typical project is “astronomical” and in advance it is not possible to define which combinations will be relevant. The choices come up during the design process when the real alternatives can be compared. However, this issue is not something which I studied in this research, and it can also be a topic for future research.

8.3.2 *Use of the Requirements Model*

8.3.2.1 *Model Servers*

The "PM4D Final Report" pointed out that IFC-based file exchange is an insufficient solution for real projects [*Kam and Fischer 2002* [111]]. It is even less suitable for a concept based on *Integrated Project Information Models* consisting of four separate *Models* (Figure 23). Based on the current knowledge the best potential solution for these problems is *Model Server* technology (Appendix C, C2). Thus, the use of *Model Servers* to integrate *Requirements, Design, Production, and Maintenance Models* is an important future research area. Figure 40 documents an idea of the objects linkage in such a multi-model environment.

When such an active link between a project's *Requirements and Design Models* is implemented in a reasonably robust way, testing of the potential benefits and problems of *Model* integration will be possible in real construction projects.

8.3.2.2 *Agent Technology*

As mentioned in Section 8.3.1.6, a problem in the multi-model environment is how to recognize and update changes which are caused by changes in another *Model*. For example, if the structures in the structural *Model* change they can affect the architectural *Design Model*. A similar functional issue is related to the recognition of the systems and *Bounding Elements* and their relationship with the indirect *Requirements*. Although *IFC Specifications* include these mechanisms (Section 6.2.6), the use of agents could be an efficient technology to automate the formation of links between the objects in the *Requirements and Design Models*. A "smart" agent could automatically propagate the links to the *Bounding Elements* and systems of a *Space*.

8.3.2.3 User-interface Research

I used a simple database approach in the rapid prototype and created a user-interface (UI) which is based on the traditional *Space* sheets used in the architectural domain; one sheet contains the information of one *Space* (Section 5.4). Another demonstration UI is the mock-up for connections to the *Design Models* (Section 5.6). In addition, the solution examples in Sections 7.1.2–7.1.4 have some crude UI ideas. However, they are far from good UI design. *Requirements Models* contain a large amount of data; structuring the data so that the end-users can easily understand the structure and manipulate the data is a challenging task. This is also an interesting human-computer interaction topic for future research.

8.3.2.4 Space Program Type and Space Program Instance

The *Cascading Requirements* structure, where the *Space Requirements* are divided into *Space Program Type* (*SPT*) and *Space Program Instance* (*SPI*), is crucial for the efficiency of the *Requirements Management* system (Sections 5.1.1 and 6.1.7). I base my *Requirements Model Specification* on this concept (Section 6.3.10). Sections 6.3.10.1–6.3.10.9 document the *Requirements* on the *SPT* and *SPI* levels.

The current structure was functional in the two rapid prototype implementations (Chapter 5). However, further research of the division is needed to determinate whether this is an optimal solution, or whether some *Requirements* should be moved from the *SPT* to the *SPI* level, or perhaps the *Cascading Requirements* structure should have even several levels – including a “super-type” for *SPTs*?

8.3.2.5 Connection to Requirements Capturing and Management Processes

As documented in Section 8.2.3.1, some *Requirements Capturing* tool, such as the EcoProp or CRPM system, could be connected to my *Requirements Model Specification*. Depending on the *Requirements Capturing* system and the level of integration, this can just be an implementation issue, but it also can be a future research topic. How to capture the detailed *Requirements* in a system which is linked to the *Design Model*, and how to maintain and update the information when the *Requirements* evolve during the process?

8.3.2.6 Use of Requirements History

One area for future research is the *Requirements History* – how the *Project Requirements* evolved during the process. My *Requirements Model Specification* provides a conceptual basis to store all the *Requirements Changes* during the process in a "history database" (Section 6.2.3). There are several interesting topics in this area: How to implement such a historic perspective of *Requirements Management* in detail? Which functionalities would the user-interface need?

Other research issues in this area are the GUID problems identified in Section 6.2.3.3. How can we improve GUID mechanisms to overcome the problems?

8.3.2.7 Automated Requirements Verification and Conflict Identification

Many *Client Requirements* are verbal descriptions and only human interpretable *(Requirement Descriptions)*, but some have an exact content *(Requirement Attributes).* In total 143, 48% of the *Requirements* in my *Specification,* are in this category (Table 10). The possibility of using these "exact" *Requirement Attributes* for automated verification, i.e., check with an application how well a design meets the *Requirements*, is a potential usage of the *Requirements Model*. At least one commercial tool for automated *Model* checking already exists, Solibri Model Checker [*Solibri* [112]]. The use of the *Requirements Model* as the reference for verification could widen the use of this and other *Model* checking tools. I believe that this field includes several open research questions, such as, how to utilize simulation results in the verification or how to follow the actual behavior of the building compared to the *Requirements*?

A totally new field could be automated conflict identification between *Requirements*. As documented in Section 8.3.1.8, it is possible that some *Requirements* are mutually exclusive. Could such conflicts in *Requirement Attributes* be identified automatically in the *Requirements Model*? Is the problem similar to verification of design solutions?

8.3.2.8 Semi-automated Design Tools

One of the most impressive demonstrations I have seen of the power of integrated software tools was the BLIS presentation referred to in Section 7.3.2 (Figure 89). It demonstrated how a spatial *Model* created with the Space Layout Editor application in MS Visio was exported to Graphisoft's ArchiCAD, which automatically generated the walls between *Spaces*. A *Requirements Model* could take this automation one step further; the walls could get their *Properties* automatically from the *Space Requirements* in the project's *Requirements Model*. Likewise, design tools could automatically read the *Requirements* of related *Spaces* and select the right door type for the designer when he is positioning the doors in the *Model*. It is possible to develop several scenarios, where the *Requirements Model* could generate correct solutions on the detail level or assist the designers to use correct solutions. Identifying and testing relevant scenarios in this area provides a variety of interesting future research topics.

8.4 Conclusions

My first observation of the *Requirement Management* problem was my own experience of a research laboratory project in the mid 1980s. I was responsible of the schematic design but another architect in our office took over the project for detailed design and construction documentation. When I visited the building after its completion I realized how much of the initial *Requirements* had been lost because of inadequate *Requirements Documentation*. However, in the paper-based process this problem was very difficult to fix. In 1994, almost 10 years later, I realized some of the potential of information technology in linking area *Requirements* with the design tools and, as documented in Section 7.1.1, this linkage provided significant benefits in the ICL headquarters project. However, the area *Requirements* are just one aspect of the problem. As the other case studies in Section 7.1 illustrate, construction projects have many problems in *Requirements Management*, and thus the solution must cover more than just area *Requirements*.

The *Requirements Management* problems are mainly related to the amount of information on construction projects and the lack of time of the *Project Team* members. In most cases, the problems are not caused by the lack of skills or goodwill of the designers; they just do not have sufficient resources for *Requirements Management*. Thus, efficiency of the tools is a crucial issue if we want to improve *Requirements Management* in the AEC industry. One solution is to provide a shared *Requirements Model* and to link the *Requirements* to the design tools as documented in this research.

However, the tools alone cannot solve this problem. Someone has to populate the initial *Requirements Model*, update it when the *Requirements* evolve, and also verify the design solutions compared to the *Requirements*. This role can be an extension of the project manager's or designers' roles, but ideally a Requirements Manager would be a new task in AEC projects. The person responsible for *Requirements Management* should be an expert of the specific building type and also have adequate skills and resources to communicate with all project stakeholders, bring up the contradictions and facilitate the trade-offs.

Current "value-engineering" is often not what the name suggests; instead of creating value it focuses on cutting costs, and often these cuts are made in the functionalities which are crucial for the end-users as, for example, the Clark Center case indicates (Section 7.1.2). AEC projects need someone who is actively involved in the design and construction process and acts as the spokesperson for the end-users of the building managing the trade-offs and changes. Explicit documentation and updating of the *Requirements* can help to facilitate the process by making the changes visible and traceable. My *Requirements Model Specification* includes the elements needed to store and track changes of individual *Requirements*, but the utilization of this functionality needs future research; which features would a good tracking tool need (Section 8.3.2.6)?

A systematic *Requirements Management* tool can also help improve the quality of the *Requirements;* the initial *Requirements* are often not well-formulated or even correct. However, in the current process the *Requirements Changes* are often scattered in different documents and difficult to find afterwards if we want to verify the design against *Requirements*.

A systematic *Requirements Management* tool could improve the quality by creating (1) a formal framework for *Requirements*, (2) *Requirements Templates* for different building types, and (3) a data storage which can be compared, not only to the design solutions, but also to the maintenance information throughout the life of the building. Depending on the business model of the companies participating in a project, the *Requirements Templates* could be managed by the *Clients*, designers or construction companies, and they would provide an easy method to set up the initial goals for a project and update both project-specific *Requirements* and *Requirements Templates*, thus creating a useful knowledge base for the users. Likewise, a systematic method to follow *Requirements* related to the specific building elements compared to the maintenance of the building would provide knowledge for the *Requirements* setting for new projects; which *Requirements* have led to good or inadequate design solutions.

The existing simulation software provide possibilities to make virtual prototypes of buildings, which can solve one of the fundamental problems in the AEC industry: production of unique buildings without testing their functionality before their construction. An explicit *Requirements Model* provides the benchmark values for the simulations as well as for the design “spell-checking” (Section 8.3.2.7).

The current mainstream use of information technology just automated the drafting process, and although it improved designers’ productivity significantly compared to manual drafting, it did not change the process or documentation much. The information is still fragmented and repeated in different documents and design changes must be updated in several documents.

The use of *Building Product Models* will change the tasks and processes in the AEC industry fundamentally. The use of *Models* provides the opportunity to manage information instead of documents and link information in ways that are not possible in a document-based environment. My *Requirements Model* is one example of this potential of the *Model*-based approach.

Appendix A: Terminology

This appendix defines the key terms and abbreviations used in this research. When used in the defined meaning, these terms are formatted in *Italic*.

Bounding Elements: Physical building elements bounding a *Room*, including walls, slabs, doors, windows, etc.

Building Product Model: A computer-interpretable description of a building structured according to some *Model Specification*, such as the *IFC Specification*.

Building Program: The documented *Requirements* for a building project.

Cascading Requirements: This research uses the term *Cascading Requirements* when, for example, *Space Program Instances* (*SPIs*) "inherit" the *Requirements* from a *Space Program Type* (*SPT*). *SPI* is not a sub-class of the *SPT*, but all *Requirements* defined in a *SPT* are included in the *Requirements* for all *SPIs* assigned to the *SPT*.

Clients: Building owner(s) and end-user(s) of the building who participate in the *Requirements Capturing* and/or *Requirements Management* by defining *Requirements*. Other project stakeholders, such as the community, are assumed to provide input to the project through the *Client(s)* and *Project Team*.

Client Requirements (CR): Detailed *Requirements* which define some *Client* need, provide useful information for design decisions, and can link to object(s) in the *Design, Production, and Maintenance Models* on some level, e.g., project, site, building, *Space*, building envelope, etc. *CRs* can be either *Requirement Attributes or Requirement Descriptions.* The rapid prototype implementation of this research discussed in Chapter 5 focused on *CRs* which relate to *Spaces*.

Conceptual Model: This research uses the term "*Conceptual Model*" is used for *Model* structures which are illustrations of a principle rather than actual *Specifications*. C.f. *Model* and *Specification*.

Design Object: An object which is stored in *Building Product Models* (*DPM Models*), for example, project, site, building, *Space*, column, door, wall, etc.

Design Model: An instantiated *Building Product Model* representing a design solution. Several *Design Models* can be linked to one *Requirements Model*. C.f. *Maintenance, Production and Requirements Models*, and Figure 23.

DPM models: Design, Production, and Maintenance models.

Direct Requirement (DR): A *Requirement* defined and managed by the *Client* or his appointed representative in the *Project Team;* for example, required area, needed equipment or allowed minimum and maximum temperatures for a *Space*. *DRs* can be either *Requirement Attributes or Requirement Descriptions.*

External Requirement (ER): A *Requirement* defined for a building project by external sources, such as building codes, local regulations, permitting authorities, and neighbors. *ERs* can be either *Requirement Attributes or Requirement Descriptions.*

Generic Requirements Object: A *Requirements Object* which has no specified content or specified connection to any level in the *DPM Models*. The end-user of a *Requirements Management* application defines both content and connections of *Generic Requirements Objects*.

Geometrical Location: The *Location* of a building element in the *DPM Models.* These *Locations* can be defined in different coordinate systems either as an absolute *Location* or as a *Location* relative to another element. They specify the exact place of the element in the *Model*. C.f. *Required Location.*

IFC Specification(s): Industry Foundation Classes (IFC), the *Building Product Model Specifications* defined by the International Alliance for Interoperability. There are several versions of the *IFC Specifications*, for example, IFC 2.0, IFC 2x, IFC 2x2, and IFC 2x2 Addendum 1. The singular format *IFC Specification* refers in this research to IFC 2x2 Addendum 1 if an other version is not specified [*IFC 2004 Add1* [113]].

Implementation View: *Implementation Views* are a concept developed by BLIS to support IFC-based information exchange (Section 3.5). The views consist of concepts which define a specific subset of objects for implementation for a specific use case and how to implement those objects for IFC data exchange.

Indirect Requirement (IR): A *Requirement* for objects on the same or lower level in the *Design Model* derived from or related to some *Direct Requirement* (Section 6.3.1). For example, *Requirements* defined for a *Room* can cause *Indirect Requirements* for the walls bounding the *Room*, such as sound or thermal insulation *Requirements*. *IRs* can be *Requirement Attributes* or *Requirement Descriptions*.

Instance: Instance is a specific object in the *Model*, e.g., an individual occurrence in a populated data set. For example, IfcSpace is an object definition in the *IFC Specifications*, and a specific *Space* in the *Model* for a project is an *Instance*.

Instance-Specific Property (ISP): A *Requirement* or *Project Attribute* which relates to the *Space Program Instances (SPIs)* in the *Requirements Model Specification*, c.f. *SPI, SPT* and *TSP*.

Instantiated Model: An instantiated representation of a *Model*, such as the data set of a building, based on some *Model Specification*. For example, the *Requirements Model* contains the *Requirements* of a project structured according to the *Requirements Model Specification*. Likewise the *Design Model* contains a project's *Design Objects* structured according to some *Building Product Model Specification*, for example, the *IFC Specification*.

Integrated Project Information Model: Set of *Models* linked to each other and containing some part of a project's information, such as *Requirements Model*, architectural *Design Model*, or *Maintenance Model*. The *Integrated Project Information Model* can also include other types of project information, but those are not in the scope of this research. C.f. Figure 23.

Location: In this research *Location* has two different meanings: *Required Location* and *Geometrical Location*.

Maintenance Model: An instantiated *Building Product Model* representing the as-built building. It can also include other types of project information, but those are not in the scope of this research. C.f. *Design, Production, Maintenance and Requirements Models*, and Figure 23.

Model: "An abstraction and representation of the relevant characteristics of the target system for a purpose" [*ProIT 2004* [114]].

Model Server: In this research *Model Server* means specifically an IFC compliant *Model Server* which can store IFC *Building Product Model* within a database system and run over the Internet. IFC-compatible applications can communicate with each other via the Internet and utilize functions implemented in the *Model Server*, such as partial import or export of an *Instantiated Model*.

Model Specification: Formal definition of a *Model* structure, such as *Requirements Model Specification* and *Building Product Model Specification*. C.f. *Model*.

Multi-Value Requirement (MVR): A *Requirement* which can have several different values or references for one *Space Program Instance* (*SPI*), such as activities, equipment, and windows. for a *Space*, cf. *Single-Value Requirements*.

Production Model: An instantiated *Building Product Model* representing a production solution. Several *Production Models* can link to one *Requirements Model* and/or *Design Model*. C.f. *Design, Maintenance and Requirements Models* and Figure 23.

Project Attribute (PA): In the *Requirements Model Specification* the *Project Attributes* are attributes which do not define actual *Requirements*, but serve as identifiers, names or other information of the *Requirements Objects*, such as ID and name of a *Space*. C.f. *Requirement Attribute* and *Requirement Component.*

Project Requirements: Requirements for a specific project; usually created in *Requirements Capturing* and updated in *Requirements Management* processes.

Project Team: Group of people actively producing, managing and using information in the design and construction process, including, typically, project managers, architects, and engineers.

Property: Attribute or feature of an object in *Design, Production, and Maintenance Models*, such as area of a *Space*, thermal insulation of a window, and color of a wall. A single *Property* or a group of *Properties* can meet one or more *Requirements* in the *Requirements Model*.

Property Set: *Property Sets* are a method in the *IFC Specifications* which enables adding properties to objects without changing the *Specifications*. Definition by IAI: "*The IfcPropertySet defines all dynamically extensible properties. The property set is a container class that holds properties within a property tree. These properties are interpreted according to their name attribute.*"

Required Location: Defines *Client Requirements* for a *Location* of a *Space* or group of *Spaces*, usually in relation to other adjacent *Spaces* or a specific story, part of the *Requirements Model*. C.f. *Geometrical Location*.

Requirement: A statement of quality or desired *Property* of the building or its parts. The possible *Requirements* depend on building type and *Client* needs, and, as documented in this research, the list cannot be standardized. Thus, the *Requirements Model Specification* must be a flexible framework which also enables additional project-specific *Requirements*.

Requirement Attribute: Requirement which has a numeric *Target Value* and can be verified from the *Design Model*, not only by human interpretation, but also by calculations, simulation results or other computational methods, such as required area, minimum or maximum temperature, ceiling height, connections to other *Spaces*, and maximum noise level. C.f. *Requirement Description* and *Project Attribute.*

Requirement Description: Requirement defined by a verbal description, and thus needing human interpretation, c.f. *Requirement Attribute.*

Requirements Capturing: The process defining original *Project Requirements* before the design process, c.f. *Requirements Management*.

Requirements Changes: Changes made to the *Project Requirements* in the *Requirements Management* process during the design, construction or maintenance process after the *Requirements Capturing* phase.

Requirement Component: My *Requirements Model Specification* includes also elements which are not actual *Requirements*, such as the *Project Attributes,* for example, ID, purpose and name of a *Space*. To avoid repetitive use of the combination *"Project Attributes* and *Requirements"* I use the term *"Requirement*

Components" covering both element types in the cases where the difference between these elements is not meaningful, for example, in Section 4.

Requirements Database: *Requirements* organized into a database structure. In this *research* the formatted term *Requirements Database* refers specifically to the rapid prototypes.

Requirements Documentation: All documents containing any portion of the *Project Requirements*, such as *Building Program*, environmental goals, and meeting minutes.

Requirements Hierarchy: A systematic organization of *Requirements* based on some ontology.

Requirements History: *Requirements History* consists of the previous versions of *Requirements Objects* stored in a *Requirements Management* system.

Requirements Information: The information content of *Requirements Documentation*.

Requirements Knowledge: The explicit information in the *Requirements Documentation* and the implicit and tacit knowledge of *Project Requirements* in the *Project Team*.

Requirements Management: The process to maintain and update project *Requirements* after the *Requirements Capturing* process.

Requirements Model: An *Instantiated Model* representing the *Requirements* of a specific project based on a *Requirements Model Specification*. C.f. *Maintenance, Production and Requirements Models*, and Figure 23.

Requirements Object: An objectified set of *Requirements* in the *Requirements Model*. *Requirements Objects* can link to each other, and one *Requirements Object* can link to several objects in the *Design Model*. The set of *Requirements* in a *Requirements Object* can be expanded using the *Property Set* mechanism. C.f. Section 6.1.4

Requirements Ownership: Requirements Ownership specifies the actor responsible for creating and managing the *Requirement*. This actor can be a member of the *Project Team* or a *Client* representative.

Requirements Template: A predefined subset of *Requirements* which are relevant for a specific project or building type and can be used as the basis for new projects.

Requirements View: A functionality proposed in my research to show the *Requirements* linked to a specific object in the *Design Model* to the user of design software. C.f. Section 5.6.

Room: Room is a special case of *Space*, defined by physical boundaries.

Shared Properties (ShP): Requirements or *Project Attributes* which can relate either to the *Space Program Type* (*SPT*) or to the *Space Program Instance* (*SPI*) in the *Requirements Model*. The idea of *Virtual SPTs* made the *ShPs* obsolete in the final *Requirements Model Specification*, but they were used in the rapid prototyping (Section 5.5). C.f. *ISP*, *TSP*.

Single-Value Requirements (SVR): Requirements which can have only one value or reference for one *Space Program Instance (SPI)*, such as noise level, maximum number of occupants, and maximum temperature for a *Space*, cf. *Multi-Value Requirements*.

Space: A *Space* is an area (2D representation) or volume (3D representation) bounded either physically or virtually for certain functions within a building. According to the *IFC Specifications* a *Space* can also consist of multiple other *Spaces*, c.f. *Room* and Section 6.3.1.5.

Space Program: A documented set of *Space Requirements* for a project, c.f. *Building Program*.

Space Program Instance (SPI): A *Requirements Object* for *Spaces* in the *Requirements Model*. *SPI* defines *Requirements* for the *Space Instances* to which it is linked in the *DPM Models*. One *SPI* can be linked to several *Spaces* in the *DPM Models*, and it can “inherit” *Cascading Requirements* from an *SPT*, cf. *SPT* and Figure 24.

Space Program Type (SPT): A *Requirements* definition for *Space Types* in the *Requirements Model*. *SPTs* do not have direct links to the objects in the *Design Model;* they relate only to the *Space Program Instances* (*SPIs*). One *SPT* can link to several *SPIs*, c.f. Figure 24 and Figure 38.

Space Requirements: A documented set of *Requirements* related to *Spaces*.

Space Type: Similar abstraction for a group of *Spaces* in the real world, as the *Space Program Type* is for the *Space Program Instance*. General concepts, such as kitchen, office or meeting *Room*, are *Space Types*, and their *Instances* are *Spaces*.

Specification: C.f. *Model Specification*.

Sub-Model: Any partial *Model* containing part of the project's information and linked to the other *Models* in the *Integrated Project Information Model* environment, such as *Requirements Model*, architectural *Design Model*, or *Maintenance Model*.

Target Value: “A specific value that defines the solution space for design attributes (e.g. 5,000 m2 for gross floor area or 10% of gross floor area as circulation *Space*)” [*Kamara et al., 2003* [115]]. In my *Requirements Model Specification* all *Requirement Attributes* have *Target Values*. In the rapid prototyping database, the attributes for which the data type is integer or real are *Target Values* (*Table 5*).

Type-specific Property (TSP): A *Requirement* or *Project Attribute* which relates to the *Space Program Types (SPTs)* in the *Requirements Model Specification*. C.f. *ISP*, *SPI* and *SPT*.

Virtual SPT: *Virtual Space Program Type* concept separates the contents of *Space Program Instances* and *Space Program Types*. Each *SPI* which is not based on a defined *SPT* automatically creates its own *Virtual SPT* which is identified based on the ID of the *SPI*. This principle prevents the duplication of the same fields in the *SPT* and *SPI* databases and thus simplifies the database structure and implementation (Section 5.5).

Appendix B: Detailed Requirements Tables

Appendix B1: Requirements Used in the Analyzed Projects

Table 13 is based on the analysis of the *Requirements Documentation* of five building projects [*Programs 2003* [116]]. The structure is based mainly on the EcoProp attribute list (Appendix B3, Table 15 and Appendix B4, Table 16) with additional attributes from the cases, where a *Requirement* was specified in at least one of the projects, but was not included in the original EcoProp list. The number in the "Defined" column indicates how many of the five projects have used this specific *Requirement*. Only one project had several detailed references to specific building codes, which are indicated at the end of Table 13.

Table 13: *Building Program* analysis; number of projects using each *Requirement* type

A CONFORMITY REQUIREMENTS AND LIMITATIONS	Projects
A1 LOCATION	
A1.1 Site requirements	
Geographical location	
Soil type requirements; excavation and foundation	1
Orientation (solar availability)	
Road infrastructure	
Electricity supply distribution infrastructure	
Gas supply infrastructure	1
Water supply infrastructure	1
Sewage infrastructure	1
Waste service infrastructure	1
Size and suitability requirements for the site	1
A1.2 Transportation requirements	
Availability of public transportation	
Frequency of public transportation	
Distance from public transportation	
Distance from airports	
Accessibility for pedestrians	1
Accessibility for bicyclists	1
Vehicular access to site	1
Parking spaces	1
Bike parking	1
A1.3 Site limitations	
Existing buildings which can/must be demolished	
Existing buildings which must be preserved	1
Existing buildings which have related activities	1
Cultural, historical or recreational value of the site	
Noise level on the site (traffic, airplanes, neighbor buildings, etc.)	1
Site contamination	
Storm water	

A1.4 Environmental impact limitations	
Allowed building location	1
Allowed building footprint size	1
Allowed height of the building	
Allowed number of floors	1
Shading effect	
Glare of building surfaces	
Wind effects	1
Noise emissions	
Heat emissions	
Odor emissions	
A2 AVAILABLE SERVICE REQUIREMENTS	
A2.1 Business and commercial services	
Accommodation services	
ATM/ Banking services	
Laundry	
Maintenance services	
Office services	
Police	
Post services	
Security services	
Shoe repairs	
Shopping malls	
Travel agency services	
A2.2 Car services	
Car rental	
Gas station	
Service stations	
A2.3 Children	
Daycare services	
Schools	
A2.4 Cultural services	
Leisure services; movie theaters, theatres, etc.	
Library	
Parks and other recreational services	
Religious services	
A2.5 Food services	
Fast food services	
Grocery store	
Lunch services	
Market place	
Restaurant	
A2.6 Healthcare and welfare services	
Dentist	
Gym or other exercise services	
Hairdresser/barber services	
Health center	
Hospital	
Pharmacy	

A3 SPATIAL SYSTEM: Space-specific descriptions and requirements	
A3.1 Instance-specific descriptions	
Identifier	4
Name	6
Main purpose of the room (description)	3
Room type	3
Department	6
A3.2 Instance-specific requirements	
Adjacency requirements (connections to other rooms)	5
Number of the rooms	6
Minimum area	5
Maximum number of occupants	4
A3.3 Type- or instance-specific requirements	
Activities	4
Access floor	1
Floor finishes	2
Wall finishes	2
Ceiling finishes	2
Ceiling height	1
Furniture	4
Equipment	4
Doors	2
Windows	1
B PERFORMANCE	
B1 INDOOR CONDITIONS	
B1.1 Indoor climate	
Descriptive text, no specified values	2
Minimum room temperature	2
Maximum room temperature	3
Individual control of room temperature (maximum ± difference)	1
Temporary deviation from set values	1
Maximum air velocity	1
Maximum vertical temperature difference	1
Floor temperature	1
Maximum relative humidity	
Minimum relative humidity	1
Minimum airflow per person (normal occupancy, no smoking)	
Individual control for temporarily increased ventilation	
Basic air change rate when no occupancy	
The air leakage value of the building envelope	
Radon	1
Carbon dioxide (CO_2)	1
Carbon monoxide (CO)	1
Ammonia and amines (NH_3)	1
Formaldehyde (H_2CO)	1
Volatile organic compounds(TVOC)	1
Ozone (O_3)	
Odor intensity (intensity scale)	1
Microbes	
Mass concentration of airborne particulate matter	

B1.2 Acoustics	
Descriptive text, no specified values	2
Maximum noise level	1
Maximum equipment noise level	
Maximum traffic noise level in the building	
Maximum traffic noise level on the site	1
Maximum outdoor area noise level	
Sound insulation, building envelope	
Sound insulation between apartments	
Sound insulation between rooms	2
Impact sound insulation	
Maximum reverberation time	
Minimum reverberation time	
B1.3 Illumination	
Descriptive text, no specified values	2
Luminance distribution	1
Maximum luminance at the task area	1
Minimum luminance at the task area	1
Adjustability	1
Glare (IES-IND)	1
Shielding angle for veiling reflections and reflected glare	
Brightness/shine/luster reflection	1
Contrast repetition/reproduction CRF	1
Shadow formation	1
Directional lighting of visual tasks	
Maximum color temperature	1
Minimum color temperature	1
Color rendering	1
Daylight	3
Darkenable	
Average service life for the light source	
Energy efficiency of the light source	
Luminaire luminance limits with downward flux	
B1.4 Vibration conditions	
Descriptive text, no specified values	1
B2 SERVICE LIFE	
B2.1 Service life for building elements	
Expected building service life	1
Expected service life of load bearing structures	1
Expected service life of components which are difficult to replace	1
Expected service life of major, replaceable external elements (e.g. cladding, windows, doors)	1
Expected service life of major internal elements (e.g. partition walls)	1
Expected service life of other int. elements (e.g. surface materials, doors)	1
B2.2 Service life requirements for technical systems	
MEP-metering, safety and control devices	1
Heat yield machinery (heat transfer casing/boilers, accumulators, tanks)	1
Water circulation heat distribution machinery (steel pipes and battery)	1
Easily replaceable piping (visible)	1

	Inconveniently replaceable piping (inside or behind structures)	1
	HVAC equipment/machine heat transfer-element/installment (heating and cooling radiators)	1
	Ventilation/air conditioning ducts	1
	Terminal, control and other devices in ventilation/air conditioning ducts	1
	HVAC pumps, fans	1
	Water plumbing system components (sealing and control valve, mixers)	1
	Sewer system plumbing and components.	1
	Water and sewer fittings (wash basins, WC-seat, bath)	1
	HVAC-EL-automation systems (control room devices, regulation/setting)	1
	HVAC-EL automation cabling	1
	Reliability/availability requirements	
B3 ADAPTABILITY		
B3.1 Adaptability of spatial and structural systems		
	Initial users' possibility of making individual choices	1
	Users' possibilities to make changes later	1
	Possibilities to make changes in the use of the building	1
	Expandability of the building	1
	Alternative furnishing of spaces	1
	Division and combination of spaces	1
	Alternative use of spaces	1
	Flexibility of the frame structure	1
	Flexibility of the floor structures	1
	Flexibility of the building envelope	1
	Flexibility of the partition walls	1
	Flexibility of the vertical shafts	1
	Flexibility of the horizontal installations	1
B3.2 Adaptability of building services systems		
	Flexibility of the heating system	1
	Flexibility of the ventilation and cooling system	1
	Flexibility of the building automation systems	1
	Flexibility of the water supply system	
	Flexibility of the sprinkler system	1
	Flexibility of the waste disposal system	1
	Flexibility of the main electrical distribution system	1
	Flexibility of the electrical systems on space level	1
	Flexibility of the illumination system	1
	Flexibility of the telecommunications and IT networks	1
	Flexibility of the security and access control system	1
	Flexibility of the fire alarm system	1
B4 SAFETY		
B4.1 Structural safety		
	Bearing/load capacity	2
	Stability	2
	Stiffness	2
B4.2 Fire safety		
	Fire-resistance rating	3
	Fire-resistance time	3
	Fire-resistance rating of functional elements and accessories	2
	Surface layer fire-propagation rating	3

	Surface layer inflammability rating	3
	Fire alarm and sprinkler systems	3
B4.3 Safety in use		
	Security of information systems	1
	Electricity backup systems	1
	Radiation safety	
	Other identified safety issues	
B4.4 Intrusion safety		
	Site	
	Building	
	Space groups	
	Space	1
B4.5 Catastrophic safety		
	Radiation accident	
	Toxic substance leak	
	Earthquake	1
	Volcanic (eruption)	
	Flood/Storms	
	Snow	
	Bush fire	
B5 COMFORT AND AESTHETIC REQUIREMENTS		
B5.1 Comfort requirements		
	Way-finding	2
	Visual contact/privacy internally	1
	Visual contact/privacy externally	1
	Functionality and comfort of the spaces	2
	Interior design and furniture	2
	Site amenities	2
	Outdoor area comfort and usability	2
B5.2 Aesthetic requirements		
	General design objectives for the building	2
	Aesthetic appearance of the building	3
B6 ACCESSIBILITY REQUIREMENTS		
B6.1 Site access		
	Vehicular access	1
	Emergency vehicle access	3
B6.2 Building access		
	Building is accessible for disable/handicapped	3
	Building is accessible for sight disabled people	1
	Building is accessible for hearing impaired people	1
B7 USABILITY		
C COST AND ENVIRONMENTAL PROPERTIES		
C1 LIFE CYCLE COSTS		
C1.1 Investment costs		
	Investment/initial costs	
C1.2 Operation costs		
	Operation costs	
	Energy costs	
	Service and maintenance costs	

C1.3 Demolition and disposal costs	
Disposal costs	
Value of recyclable components and materials	
C2 ENVIRONMENTAL PRESSURE	
C2.1 Biodiversity requirements	
Ecological significance and uniqueness of the site	
Green site area compared to the building footprint	
Existing vegetation; quantity, condition, and extent	2
Existing vegetation which must be preserved	2
Possible effects to the fauna	1
C2.2 Use of resources	
Water consumption	1
Total electrical energy consumption	2
Heating/cooling energy consumption	1
Energy consumption, fans	1
Energy consumption, AC	1
Energy consumption, other HVAC equipment	1
Energy consumption, HVAC system in total	1
Energy consumption, office equipment	1
Energy consumption, lighting	2
Site heating system	1
Use of solar and other renewable energy	
Use of solar protection/screens	1
Exploitation of 'half-warm' spaces for energy saving	
Air recycling/energy recovery	
C2.3 Building envelope requirements	
Roof, U-value	3
Base floor, U-value	2
External walls, U-value	3
External doors, U-value	3
Windows, U-value	3
Windows, shading coefficient	1
C2.4 Emission requirements	
CO_2eq	1
SO_2eq	
C_2H_4eq	
Renewable materials	
Non-renewable materials	
Production and distribution efficiency	
Detailed code references	
Site	1
Building	1
Structural systems	1
MEP systems	
Fire systems	1
Egress	1
Building envelope	1
Materials	1
Others	1

Appendix B2: PREMISS Requirements by the Level of Detail

Table 14 documents the PREMISS *Requirements* organized by the level of direct links. The data type is indicated in the “Type” column: A=*Requirement Attribute*, D=*Requirement Description*, and DL=list of *Requirement Descriptions*. Indirect links for each *Requirement* are indicated by “x” in the columns on the right.

Table 14: PREMISS *Requirements Hierarchy* organized by level of detail including indirect links

Project																		
A	**CONFORMITY REQUIREMENTS**		**Indirect links**															
A.1	General Objectives	Type	Site	Build.	Story	Space	Env.	Circ.	Struct.	HVAC	Plumb.	Gas	Elect.	Telec.	IT	Audio	Sec.	Fire
A.1.1	ProjectObjectives																	
A.1.1.1	GeneralObjectives	D																
A.1.1.2	TotalBuildingArea	A		x														
A.1.1.3	TotalBuildingVolume	A		x														
A.1.1.4	TotalProgramArea	A		x														
A.2	Location Requirements	Type	Site	Build.	Story	Space	Env.	Circ.	Struct.	HVAC	Plumb.	Gas	Elect.	Telec.	IT	Audio	Sec.	Fire
A.2.1	SiteRequirements																	
A.2.1.1	GeographicalLocation	D																
A.2.1.2	SiteArea	A	x															
A.2.1.3	SiteImage	D																
A.2.1.4	SoilType	D	x															
A.2.1.5	SolarAvailability	D	x															
A.2.2	InfrastructureRequirements																	
A.2.2.1	CoolingSupplyInfra	D								x								
A.2.2.2	ElectricityNetwork	D											x					
A.2.2.3	GasSupplyInfra	D										x						
A.2.2.4	HeatingSupplyInfra	D								x								
A.2.2.5	ITNetwork	D													x			
A.2.2.6	RoadInfra	D																
A.2.2.7	SewageInfra	D									x							
A.2.2.8	TelecomNetwork	D												x				
A.2.2.9	WaterSupplyInfra	D									x							
A.2.2.10	WasteInfra	D																
A.2.3	TransportationRequirements																	
A.2.3.1	AirportDistance	A																
A.2.3.2	BikeAccess	D																
A.2.3.3	CarAccess	D																
A.2.3.4	PedestrianAccess	D																
A.2.3.5	PublicTransportation	D																
A.2.3.6	PublicTransportationDistance	A																
A.2.3.7	PublicTransportationFrequency	A																
A.3	Service Requirements	Type	Site	Build.	Story	Space	Env.	Circ.	Struct.	HVAC	Plumb.	Gas	Elect.	Telec.	IT	Audio	Sec.	Fire
A.3.1	ServiceRequirements																	
A.3.1.1	BusinessServices	DL																
A.3.1.2	CommercialServices	DL																
A.3.1.3	CulturalServices	DL																
A.3.1.4	DayCareServices	DL																
A.3.1.5	FoodServices	DL																
A.3.1.6	RecreationalServices	DL																
A.3.1.7	SecurityServices	DL																
A.3.1.8	WelfareServices	DL																
B	**PERFORMANCE REQUIREMENTS**		**Indirect links**															
B.4	Safety Requirements	Type	Site	Build.	Story	Space	Env.	Circ.	Struct.	HVAC	Plumb.	Gas	Elect.	Telec.	IT	Audio	Sec.	Fire
B.4.6	AccidentAndCatastropheRisks																	
B.4.6.1	AccidentRisks	DL																
B.4.6.2	CatastropheRisks	DL																
B.4.6.3	OtherRisks	DL																

		Type	Site	Build.	Story	Space	Env.	Circ.	Struct.	HVAC	Plumb.	Gas	Elect.	Telec.	IT	Audio	Sec.	Fire
Project																		
C	**COST REQUIREMENTS**		**Indirect links**															
C.1	Life Cycle Cost Requirements																	
C.1.1	CostRequirements																	
C.1.1.1	ConstructionCosts	A																
C.1.1.2	DesignAndCMCosts	A																
C.1.1.3	InvestmentCosts	A																
C.1.1.4	SiteCosts	A																
C.1.1.5	EnergyCosts	A																
C.1.1.6	MaintenanceCosts	A																
C.1.1.7	OperationCosts	A																
C.1.1.8	DisposalCosts	A																
C.1.1.9	RecycleValue	A																
D	**ENVIRONMENTAL REQUIREMENTS**		**Indirect links**															
D.1	Sustainability Requirements																	
D.1.2	EnergyRequirements																	
D.1.2.1	CoolingEnergyConsumption	A								x								
D.1.2.2	HeatingEnergyConsumption	A								x								
D.1.2.3	HeatingEnergySource	D								x								
D.1.2.4	LightingEnergyConsumption	A											x					
D.1.2.5	RecycledEnergy	A		x														
D.1.2.6	RenewableEnergyRatio	A		x														
D.1.2.7	TotalElectricalEnergyConsumption	A											x					
D.1.2.8	TotalEnergyConsumption	A		x														
D.1.2.9	TotalHvacEnergyConsumption	A								x								
D.1.2.10	WaterConsumption	A									x							
D.1.3	EnvironmentalPressure																	
D.1.3.1	MaxC2H4eqEmissions	A		x														
D.1.3.2	MaxCO2eqEmissions	A		x														
D.1.3.3	MaxNonRenewableMaterials	A		x														
D.1.3.4	MaxSO2eqEmissions	A		x														
D.1.3.5	MinRenewableMaterials	A		x														
D.1.3.6	ProductionEfficiency	A		x														
Site																		
A	**CONFORMITY REQUIREMENTS**		**Indirect links**															
A.2	Location Requirements																	
A.2.4	SiteDesignRequirements																	
A.2.4.1	EmergencyVehicleAccess	D																
A.2.4.2	MinBikeParkingSpaces	A																
A.2.4.3	MinCarParkingSpaces	A																
A.2.4.4	MinGreenSiteArea	A																
A.2.4.5	SiteAmenities	DL																
A.2.4.6	VechicleAccess	D																
A.2.4.7	SiteTrafficRequirements	D																
A.2.5	ExistingSiteLimitations																	
A.2.5.1	CommunityRequirements	DL		x														
A.2.5.2	CulturalValue	D																
A.2.5.3	ExistingBuildings	DL		x														
A.2.5.4	BuildingsToDemolish	DL		x														
A.2.5.5	BuildingsToPreserve	DL		x														
A.2.5.6	RelatedBuildings	DL		x														
A.2.5.7	EcologicalSignificance	D																
A.2.5.8	ExistingVegetation	DL																
A.2.5.9	PreservedVegetation	DL																
A.2.5.10	FaunaEffects	DL																
A.2.5.11	SiteContamination	D																
A.2.5.12	SiteNoiseLevel	A																
A.2.5.13	StormWater	D																

Site

A CONFORMITY REQUIREMENTS — Indirect links

Code	Requirement	Type	Site	Build.	Story	Space	Env.	Circ.	Struct.	HVAC	Plumb.	Gas	Elect.	Telec.	IT	Audio	Sec.	Fire
A.2	Location Requirements																	
A.2.6	SiteRequirementsForBuilding																	
A.2.6.1	MaxHeatEmissions	A		x														
A.2.6.2	MaxNoiseEmissions	A		x														
A.2.6.3	MaxOdorEmissions	D		x														
A.2.6.4	MaxOutdoorNoise	A		x														
A.2.6.5	PermittedBuildingHeight	A		x														
A.2.6.6	PermittedBuildingArea	A		x														
A.2.6.7	PermittedBuildingFootprint	A		x														
A.2.6.8	PermittedBuildingLocation	D		x														
A.2.6.9	PermittedBuildingVolume	A		x														
A.2.6.10	PermittedNumberOfFloors	A		x														
A.2.6.11	ShadingEffects	D		x														
A.2.6.12	SurfaceGlare	D		x														
A.2.6.13	WindEffects	D		x														
A.2.7	SiteRequirementsForSystems																	
A.2.7.1	OutdoorAreaComfort	D																
A.2.7.2	SiteHeating	D								x								
A.2.7.3	SiteLighting	D											x					
A.2.7.4	SiteDrainage	D									x							
B.4.1	SafetyOfSite																	
B.4.1.1	SiteSecurity	D															x	
B.4.1.2	MonitoringOfSite	D															x	
B.4.1.3	PerimeterControl	D															x	
B.4.1.4	ProtectionFromAttack	D															x	
B.4.1.5	ControlOfParking	D															x	
B.4.1.6	ProtectionOfVehicles	D															x	

Building

B PERFORMANCE REQUIREMENTS — Indirect links

Code	Requirement	Type	Site	Build.	Story	Space	Env.	Circ.	Struct.	HVAC	Plumb.	Gas	Elect.	Telec.	IT	Audio	Sec.	Fire
B.1	Indoor Condition Requirements																	
B.1.1	IndoorClimate																	
B.1.1.1	MaxCO	A								x								
B.1.1.2	MaxCO2	A								x								
B.1.1.3	MaxH2CO	A								x								
B.1.1.4	MaxNH3	A								x								
B.1.1.5	MaxO3	A								x								
B.1.1.6	MaxOdorIntensity	A								x								
B.1.1.7	MaxMicrobes	A								x								
B.1.1.8	MaxParticles	A								x								
B.1.1.9	MaxRadon	A								x								
B.1.1.10	MaxTVOC	A								x								
B.1.1.11	NaturallyVentilated	D								x								
B.1.2	Acoustics																	
B.1.2.1	AudioSystem	DL														x		
B.1.2.2	MinImpactSoundInsulation	A							x									
B.1.2.3	MinUnitSoundInsulation	A					x		x									
B.2	Service Life Requirements																	
B.2.1	ServiceLifeOfBuilding																	
B.2.1.1	BuildingServiceLife	A																
B.2.1.2	EnvelopeServiceLife	A					x											
B.2.1.3	StructureServiceLife	A							x									
B.2.2	ServiceLifeOfTechicalSystems																	
B.2.2.1	AudioSystemServiceLife	A														x		
B.2.2.2	AutomationCableServiceLife	A								x								
B.2.2.3	AutomationControlsServiceLife	A								x								
B.2.2.4	DuctServiceLife	A								x								
B.2.2.5	ElectricalCableServiceLife	A											x					
B.2.2.6	ElectricalFittingsServiceLife	A											x					
B.2.2.7	ElevatorServiceLife	A						x										
B.2.2.8	EscalatorServiceLife	A						x										
B.2.2.9	FireSafetySystemServiceLife	A																x

Building

B PERFORMANCE REQUIREMENTS

Requirement		Type	Site	Build.	Story	Space	Env.	Circ.	Struct.	HVAC	Plumb.	Gas	Elect.	Telec.	IT	Audio	Sec.	Fire
			Indirect links															
B.2 Service Life Requirements																		
B.2.2 ServiceLifeOfTechicalSystems																		
B.2.2.10	GasSystemServiceLife	A										x						
B.2.2.11	HeatingDistributionSystemServiceLife	A								x								
B.2.2.12	HeatMachineryServiceLife	A								x								
B.2.2.13	ItCableServiceLife	A													x			
B.2.2.14	LightSourceServiceLife	A											x					
B.2.2.15	NonVisiblePipingServiceLife	A									x							
B.2.2.16	PlumbingSystemServiceLife	A									x							
B.2.2.17	PumpAndFanServiceLife	A								x	x							
B.2.2.18	RadiatorServiceLife	A								x								
B.2.2.19	SecuritySystemServiceLife	A															x	
B.2.2.20	SewerSystemServiceLife	A									x							
B.2.2.21	TelecomCableServiceLife	A												x				
B.2.2.22	VisiblePipingServiceLife	A									x							
B.2.2.23	WaterSystemServiceLife	A									x							
B.3 Adaptability Requirements		Type	Site	Build.	Story	Space	Env.	Circ.	Struct.	HVAC	Plumb.	Gas	Elect.	Telec.	IT	Audio	Sec.	Fire
B.3.1 FlexibilityOfBuilding																		
B.3.1.1	BuildingFlexibility	D																
B.3.1.2	DesignFlexibility	D																
B.3.1.3	EnvelopeFlexibility	D					x											
B.3.1.4	Expandability	D							x									
B.3.1.5	FloorFlexibility	D							x									
B.3.1.6	FrameFlexibility	D							x									
B.3.1.7	OccupancyFlexibility	D																
B.3.1.8	PartitionFlexibility	D																
B.3.2 FlexibilityOfTechnicalSystems																		
B.3.2.1	AudioSystemFlexibility	D														x		
B.3.2.2	BuildingAutomationFlexibility	D								x								
B.3.2.3	ElectricalInstallationFlexibility	D											x					
B.3.2.4	ElectricalSystemFlexibility	D											x					
B.3.2.5	ElevatorFlexibility	D						x										
B.3.2.6	EscalatorFlexibility	D						x										
B.3.2.7	FireSafetySystemFlexibility	D																x
B.3.2.8	GasSupplyFlexibility	D										x						
B.3.2.9	HeatingSystemFlexibility	D								x								
B.3.2.10	HorizontalFlexibility	D							x	x	x	x	x	x	x	x	x	x
B.3.2.11	HvacSystemFlexibility	D								x								
B.3.2.12	IlluminationFlexibility	D											x					
B.3.2.13	ItNetworkFlexibility	D													x			
B.3.2.14	SecuritySystemFlexibility	D															x	
B.3.2.15	SprinklerFlexibility	D																x
B.3.2.16	TelecomSystemFlexibility	D												x				
B.3.2.17	VerticalFlexibility	D							x	x	x	x	x	x	x	x	x	x
B.3.2.18	WaterSupplyFlexibility	D									x							
B.4 Safety Requirements		Type	Site	Build.	Story	Space	Env.	Circ.	Struct.	HVAC	Plumb.	Gas	Elect.	Telec.	IT	Audio	Sec.	Fire
B.4.2 SafetyOfBuilding																		
B.4.2.1	BuildingSecurity	D					x		x								x	
B.4.2.2	BuildingAccessControl	D					x										x	
B.4.2.3	SeparationOfZones	D															x	
B.4.2.4	FireResistanceRating	D																x
B.4.2.5	FireResistanceTime	A																x
B.4.2.6	FireSafetySystem	D																x
B.4.2.7	LoadCapacity	D							x									
B.4.2.8	SurfaceFirePropagation	D																x
B.4.2.9	SurfaceInflammabilityRating	D																x
B.4.2.10	FireRatingForFittings	D																x
B.4.2.11	AirIntakeLocation	D								x								

Building

Code	Requirement	Type	Site	Build.	Story	Space	Env.	Circ.	Struct.	HVAC	Plumb.	Gas	Elect.	Telec.	IT	Audio	Sec.	Fire
B	**PERFORMANCE REQUIREMENTS**		**Indirect links**															
B.4	Safety Requirements																	
B.4.3	SafetyOfTechnicalSystems																	
B.4.3.1	AudioSystemReliability	A														x		
B.4.3.2	ElectricalBackupSystem	D											x					
B.4.3.3	ElectricalReliability	A											x					
B.4.3.4	ElevatorReliability	A						x										
B.4.3.5	EscalatorReliability	A						x										
B.4.3.6	FireSafetySystemReliability	A																x
B.4.3.7	GasSupplyReliability	A										x						
B.4.3.8	HvacReliability	A								x								
B.4.3.9	ItNetworkBackupTime	A													x			
B.4.3.10	ItNetworkReliability	A													x			
B.4.3.11	ItNetworkSecurity	D													x			
B.4.3.12	SecuritySystemReliability	A															x	
B.4.3.13	SewerFloodingPrevention	D									x							
B.4.3.14	TelecomBackupTime	A												x				
B.4.3.15	TelecomReliability	A												x				
B.5	Aesthetic Requirements																	
B.5.1	VisualRequirements	Type	Site	Build.	Story	Space	Env.	Circ.	Struct.	HVAC	Plumb.	Gas	Elect.	Telec.	IT	Audio	Sec.	Fire
B.5.1.1	AestheticAppearance	DL																
B.5.1.2	Wayfinding	DL																
B.6	Accessibility Requirements																	
B.6.1	BuildingAccessibility	Type	Site	Build.	Story	Space	Env.	Circ.	Struct.	HVAC	Plumb.	Gas	Elect.	Telec.	IT	Audio	Sec.	Fire
B.6.1.1	AccessibilityForHandicapped	DL														x		
B.6.1.2	AccessibilityForHearingImpared	DL						x										
B.6.1.3	AccessibilityForSightDisabled	DL																
B.6.1.4	ElevatorRequirements	D						x										

Building Storey

Code	Requirement	Type	Site	Build.	Story	Space	Env.	Circ.	Struct.	HVAC	Plumb.	Gas	Elect.	Telec.	IT	Audio	Sec.	Fire
B	**PERFORMANCE REQUIREMENTS**		**Indirect links**															
B.4	Safety Requirements																	
B.4.4	SafetyOfStorey																	
B.4.4.1	StoreyEnvelopeSecurity	D					x										x	
B.4.4.2	StoreyDoorSecurity	D					x										x	
B.4.4.3	StoreyWindowSecurity	D					x										x	
B.6	Accessibility Requirements																	
B.6.2	StoreyAccessibility	Type	Site	Build.	Story	Space	Env.	Circ.	Struct.	HVAC	Plumb.	Gas	Elect.	Telec.	IT	Audio	Sec.	Fire
B.6.2.1	StoreyAccess	D						x										

Space

Code	Requirement	Type	Site	Build.	Story	Space	Env.	Circ.	Struct.	HVAC	Plumb.	Gas	Elect.	Telec.	IT	Audio	Sec.	Fire
A	**CONFORMITY REQUIREMENTS**		**Indirect links**															
A.4	Space Requirements																	
A.4.1	SpaceProgramInstance																	
A.4.1.1	AdjacentSpaces	DL																
A.4.1.2	Department	D																
A.4.1.3	EmployeeType	D																
A.4.1.4	RequestedLocation	D																
A.4.1.5	MaxOccupancyNumber	A																
A.4.1.6	MaxRequiredArea	A																
A.4.1.7	MinRequiredArea	A																
A.4.1.8	NumberOfSpaceUnits	A																
A.4.1.9	OccupancyType	D																
A.4.1.10	StandardRequiredArea	A																
A.4.1.11	NormalStartTime	A																
A.4.1.12	NormalEndTime	A																
A.4.1.13	UseDaysPerWeek	A																
A.4.1.14	UseHoursPerDay	A																
A.4.2	SpaceProgramType																	
A.4.2.1	Activities	DL																
A.4.2.2	FunctionRequirements	D																
A.4.2.3	SpecialLoadRequirements	D							x									
A.4.2.4	VibrationControl	D							x									

Space																		
A	**CONFORMITY REQUIREMENTS**		**Indirect links**															
A.4	Space Requirements	Type	Site	Build.	Story	Space	Env.	Circ.	Struct.	HVAC	Plumb.	Gas	Elect.	Telec.	IT	Audio	Sec.	Fire
A.4.3	SpaceProgramFixtures																	
A.4.3.1	AccessFloor	D																
A.4.3.2	CeilingFinishes	D																
A.4.3.3	CeilingHeight	A																
A.4.3.4	Doors	DL																
A.4.3.5	Equipment	DL																
A.4.3.6	AvEquipment	DL																
A.4.3.7	Fixtures	DL																
A.4.3.8	FloorSurface	D																
A.4.3.9	Furniture	DL																
A.4.3.10	WallFinishes	D																
A.4.3.11	Windows	DL																
B	**PERFORMANCE REQUIREMENTS**		**Indirect links**															
B.1	Indoor Condition Requirements	Type	Site	Build.	Story	Space	Env.	Circ.	Struct.	HVAC	Plumb.	Gas	Elect.	Telec.	IT	Audio	Sec.	Fire
B.1.1	IndoorClimate																	
B.1.1.12	AllowedTemporaryDeviation	A								x								
B.1.1.13	IndividualRoomTemperatureControl	A								x								
B.1.1.14	MaxAirVelocity	A								x								
B.1.1.15	MaxFloorTemperature	A								x								
B.1.1.16	MaxHumidity	A								x								
B.1.1.17	MaxHvacNoiseLevel	A								x								
B.1.1.18	MaxTemperature	A								x								
B.1.1.19	MaxVerticalTemperatureDifference	A								x								
B.1.1.20	MinAirflowPerPerson	A								x								
B.1.1.21	MinFloorTemperature	A								x								
B.1.1.22	MinHumidity	A								x								
B.1.1.23	MinNoOccupancyAirChangeRate	A								x								
B.1.1.24	MinTemperature	A								x								
B.1.1.25	TemporarilyVentilationControl	A								x								
B.1.2	Acoustics																	
B.1.2.4	BackGroundSound	A														x		
B.1.2.5	MaxReverberationTime	A																
B.1.2.6	MinReverberationTime	A																
B.1.2.7	MinSoundInsulation	A																
B.1.2.8	MaxTrafficNoiseLevel	A					x											
B.1.3	Lighting																	
B.1.3.1	ColorRenderingIndex	A											x					
B.1.3.2	ContrastReproduction	A											x					
B.1.3.3	Darkenable	D					x											
B.1.3.4	Daylight	A					x											
B.1.3.5	DirectionalLighting	D											x					
B.1.3.6	GlareIndex	A											x					
B.1.3.7	LightingAdjustability	D											x					
B.1.3.8	LightingUniformity	A											x					
B.1.3.9	LuminanceDistribution	D											x					
B.1.3.10	LusterReflection	D											x					
B.1.3.11	MaxColorTemperature	A											x					
B.1.3.12	MaxLuminance	A											x					
B.1.3.13	MinColorTemperature	A											x					
B.1.3.14	MinLampEnergyEfficiency	A											x					
B.1.3.15	MinLuminance	A											x					
B.1.3.16	NoDaylight	A					x											
B.1.3.17	ShadowFormation	D											x					
B.1.3.18	TaskLighting	D											x					
B.3	Adaptability Requirements	Type	Site	Build.	Story	Space	Env.	Circ.	Struct.	HVAC	Plumb.	Gas	Elect.	Telec.	IT	Audio	Sec.	Fire
B.3.3	FlexibilityOfSpace																	
B.3.3.1	AlternativeFurnishing	DL																
B.3.3.2	AlternativeUse	DL																
B.3.3.3	DivisionAndCombination	DL																

Space

		Type	Site	Build.	Story	Space	Env.	Circ.	Struct.	HVAC	Plumb.	Gas	Elect.	Telec.	IT	Audio	Sec.	Fire
B	**PERFORMANCE REQUIREMENTS**	**Indirect links**																
B.4	Safety Requirements																	
B.4.5	SecurityOfSpace																	
B.4.5.1	AccessControl	DL															x	
B.4.5.2	AccessZone	D															x	
B.5	Aesthetic Requirements	Type	Site	Build.	Story	Space	Env.	Circ.	Struct.	HVAC	Plumb.	Gas	Elect.	Telec.	IT	Audio	Sec.	Fire
B.5.1	VisualRequirements																	
B.5.1.4	InteriorDesignAndFunctionality	D																
B.5.1.5	InternalVisualContacts	DL																
B.5.1.6	ExternalVisualContacts	DL					x											

Building Envelope

		Type	Site	Build.	Story	Space	Env.	Circ.	Struct.	HVAC	Plumb.	Gas	Elect.	Telec.	IT	Audio	Sec.	Fire
B	**PERFORMANCE REQUIREMENTS**	**Indirect links**																
B.1	Indoor Condition Requirements																	
B.1.1	IndoorClimate																	
B.1.1.26	MaxEnvelopeAirLeakage	A																
B.1.1.27	EnvelopeVentilation	D																
B.1.2	Acoustics																	
B.1.2.9	MinEnvelopeSoundInsulation	A																
B.5.1	VisualRequirements																	
B.5.1.3	AestheticEnvelopeRequirements	DL																
D	**ENVIRONMENTAL REQUIREMENTS**																	
D.1	Sustainability Requirements	Type	Site	Build.	Story	Space	Env.	Circ.	Struct.	HVAC	Plumb.	Gas	Elect.	Telec.	IT	Audio	Sec.	Fire
D.1.1	EnergyInsulations																	
D.1.1.1	BaseFloorInsulation	A																
D.1.1.2	EnergySavingBufferSpaces	D																
D.1.1.3	ExternalDoorInsulation	A																
D.1.1.4	ExternalWallInsulation	A																
D.1.1.5	RoofInsulation	A																
D.1.1.6	SolarProtection	D																
D.1.1.7	WindowInsulation	A																
D.1.1.8	WindowShadingCoefficient	A																

Circulation System

		Type	Site	Build.	Story	Space	Env.	Circ.	Struct.	HVAC	Plumb.	Gas	Elect.	Telec.	IT	Audio	Sec.	Fire
B	**PERFORMANCE REQUIREMENTS**	**Indirect links**																
B.7	Circulation Requirements																	
B.7.1	CirculationArea																	
B.7.1.1	MaxCirculationAreaRatio	A																
B.7.2	CirculationSystems																	
B.7.2.1	LobbyRequirements	DL																
B.7.2.2	CorridorRequirements	DL																
B.7.2.3	StairRequirements	DL																
B.7.2.4	ElevatorRequirements	DL																
B.7.2.5	EscalatorRequirements	DL																
B.7.2.6	LoadingDockRequirements	DL																

Appendix B3: PREMISS Requirements Compared to the EcoProp System

Table 15 documents the PREMISS *Requirements* organized by categories compared to the EcoProp *Requirements* [*EcoProp* [117]]. Blank spaces in the EcoProp column indicate that the *Requirement* does not exist in the EcoProp system, and in some cases one PREMISS *Requirement* covers several EcoProp *Requirements*. Table 16 in Appendix B4 documents the full list of EcoProp categories and all rejected EcoProp *Requirements*.

Table 15: PREMISS *Requirements* compared to the EcoProp system

PREMISS						EcoProp	
A	**CONFORMITY REQUIREMENTS**						
	A.1	***General Objectives***					
		A.1.1	ProjectObjectives				
				A.1.1.1	GeneralObjectives	B5	General design objectives
						A1.1	Corporate quality (perceptivity, building location/site)
						A1.1	International: Level of industrialization
						A1.1	Number of locations (one, more than one)
						A1.1	Regional atmospheric conditions
				A.1.1.2	TotalBuildingArea		
				A.1.1.3	TotalBuildingVolume		
				A.1.1.4	TotalProgramArea		
	A.2	***Location Requirements***					
		A.2.1	SiteRequirements				
				A.2.1.1	GeographicalLocation	A1.1	Geographical location (domestic, international)
				A.2.1.2	SiteArea	A1.1	Construction efficiency and tightness of site
				A.2.1.3	SiteImage		
				A.2.1.4	SoilType	A1.1	Soil type (Foundation and establishment)
				A.2.1.5	SolarAvailability	A1.1	Orientation (solar availability)
						A1.3	Daylight
						A1.3	Heat absorptioin and reflected radiation
						A1.3	Winter sunlight
		A.2.2	InfrastructureRequirements				
				A.2.2.1	CoolingSupplyInfra		
				A.2.2.2	ElectricityNetwork	A1.1	Electricity distribution infrastructure adequacy
				A.2.2.3	GasSupplyInfra	A1.1	Local gas supply infrastructure adequacy
				A.2.2.4	HeatingSupplyInfra		
				A.2.2.5	ITNetwork		
				A.2.2.6	RoadInfra	A1.1	Local roads infrastructure adequacy
				A.2.2.7	SewageInfra	A1.1	Local sewage infrastructure adequacy
				A.2.2.8	TelecomNetwork		
				A.2.2.9	WaterSupplyInfra	A1.1	Local water supply infrastructure adequacy
				A.2.2.10	WasteInfra	A1.1	Local solid waste infrastructure adequacy
		A.2.3	TransportationRequirements				
				A.2.3.1	AirportDistance	A1.2	Accessibility/striking distance of air flights
				A.2.3.2	BikeAccess	A1.2	Public bicycle paths in the area
				A.2.3.3	CarAccess		
				A.2.3.4	PedestrianAccess	A1.2	Accessibility/striking distance by pedestrian and bicycle
				A.2.3.5	PublicTransportation	A1.2	Availability of public transport
				A.2.3.6	PublicTransportationDistance	A1.2	Accessibility/striking distance by public transport
				A.2.3.7	PublicTransportationFrequency	A1.2	Frequency of public transport service (quality)

PREMISS					EcoProp	
A	CONFORMITY REQUIREMENTS					
	A.2	***Location Requirements***				
		A.2.4	SiteDesignRequirements			
			A.2.4.1	EmergencyVehicleAccess		
			A.2.4.2	MinBikeParkingSpaces		
			A.2.4.3	MinCarParkingSpaces	A1.1	Parking spaces
			A.2.4.4	MinGreenSiteArea	A1.1	Green area on site
			A.2.4.5	SiteAmenities	B5	Site amenities for shade, relaxation and play
					B6	External spaces
			A.2.4.6	VechicleAccess	A1.2	Vehicular access to site
			A.2.4.7	SiteTrafficRequirements		
		A.2.5	ExistingSiteLimitations			
			A.2.5.1	CommunityRequirements		
			A.2.5.2	CulturalValue	A1.1	Cultural, historical or recreational value of site
			A.2.5.3	ExistingBuildings	A1.1	There are such works and buildings
			A.2.5.4	BuildingsToDemolish		
			A.2.5.5	BuildingsToPreserve		
			A.2.5.6	RelatedBuildings	A1.1	Availability of existing structure(s) with potential for renovation
			A.2.5.7	EcologicalSignificance	A1.1	Ecological and agricultural significance, contamination
			A.2.5.8	ExistingVegetation	A1.1	Existing vegetation quantity, condition, and extent
			A.2.5.9	PreservedVegetation	C2.1	Biodiversity
			A.2.5.10	FaunaEffects	*C2.1*	*Biodiversity*
			A.2.5.11	SiteContamination	*A1.1*	*Ecological and agricultural significance, contamination*
			A.2.5.12	SiteNoiseLevel	B1.2	Vehicular noise level lpa,eq,max (db), base noise
					B1.2	Industrial noise lpa,Eq,T (db), lpa,max (db) 22-6
					B1.2	Building outdoor areas lpa,eq,T (db), 6-18
					B1.2	Building outdoor areas lpa,eq,T (db), 18-22
					B1.2	Building outdoor areas lpa,eq,T (db), 22-6
					B1.2	Site acoustics
			A.2.5.13	StormWater	A1.1	Storm water
		A.2.6	SiteRequirementsForBuilding			
			A.2.6.1	MaxHeatEmissions	A1.3	Discharge heat from building
			A.2.6.2	MaxNoiseEmissions	A1.3	Noise factors from building
			A.2.6.3	MaxOdorEmissions	A1.3	Incident smells from building
			A.2.6.4	MaxOutdoorNoise		
			A.2.6.5	PermittedBuildingHeight		
			A.2.6.6	PermittedBuildingArea		
			A.2.6.7	PermittedBuildingFootprint		
			A.2.6.8	PermittedBuildingLocation	A1.1	Building placement on site
			A.2.6.9	PermittedBuildingVolume		
			A.2.6.10	PermittedNumberOfFloors		
			A.2.6.11	ShadingEffects		
			A.2.6.12	SurfaceGlare	A1.3	Glare of building surfacing
			A.2.6.13	WindEffects	A1.3	Wind
		A.2.7	SiteRequirementsForSystems			
			A.2.7.1	OutdoorAreaComfort		
			A.2.7.2	SiteHeating		
			A.2.7.3	SiteLighting		
			A.2.7.4	SiteDrainage		
	A.3	***Service Requirements***				
		A.3.1	ServiceRequirements			
			A.3.1.1	BusinessServices	A3	Accomodation services
					A3	Banking facilities/services (ATM)
					A3	Employment opportunities (Work places)
					A3	Maintenance services
					A3	Office services
					A3	Post services
					A3	Travel agency services

PREMISS				EcoProp	
A	**CONFORMITY REQUIREMENTS**				
A.3	***Service Requirements***				
A.3.1	ServiceRequirements				
		A.3.1.2	CommercialServices	A3	Car services
				A3	Hairdresser/barber services
				A3	Laundry
				A3	Market place
				A3	Shoe repairs
				A3	Specialty stores
		A.3.1.3	CulturalServices	A3	Culture services
				A3	Library
				A3	Religious services
		A.3.1.4	DayCareServices	A3	Nurseries (day-care) and schools
		A.3.1.5	FoodServices	A3	Bakery
				A3	Cafe (Eating/soup-kitchen services)
				A3	Commercial services (eg. Kiosk, grocer)
				A3	Fast food
				A3	Restaurant
		A.3.1.6	RecreationalServices	A3	Park
				A3	Pedestrian street/avenue
				A3	Recreational services, exercise and interest services
		A.3.1.7	SecurityServices	A3	Police
				A3	Safety/security services
		A.3.1.8	WelfareServices	A3	Health care and welfare services
				A3	Dentist
				A3	Pharmacy
A.4	***Space Requirements***				
A.4.1	SpaceProgramInstance				
		A.4.1.1	AdjacentSpaces		
		A.4.1.2	Department		
		A.4.1.3	EmployeeType		
		A.4.1.4	RequestedLocation		
		A.4.1.5	MaxOccupancyNumber		
		A.4.1.6	MaxRequiredArea		
		A.4.1.7	MinRequiredArea		
		A.4.1.8	NumberOfSpaceUnits		
		A.4.1.9	OccupancyType		
		A.4.1.10	StandardRequiredArea		
		A.4.1.11	NormalStartTime		
		A.4.1.12	NormalEndTime		
		A.4.1.13	UseDaysPerWeek		
		A.4.1.14	UseHoursPerDay		
A.4.2	SpaceProgramType				
		A.4.2.1	Activities		
		A.4.2.2	FunctionRequirements		
		A.4.2.3	SpecialLoadRequirements		
		A.4.2.4	VibrationControl	B1.4	Vibration conditions
A.4.3	SpaceProgramFixtures				
		A.4.3.1	AccessFloor		
		A.4.3.2	CeilingFinishes	B5	Materials
		A.4.3.3	CeilingHeight		
		A.4.3.4	Doors		
		A.4.3.5	Equipment		
		A.4.3.6	AvEquipment		
		A.4.3.7	Fixtures		
		A.4.3.8	FloorSurface	*B5*	*Materials*
		A.4.3.9	Furniture		
		A.4.3.10	WallFinishes	*B5*	*Materials*
		A.4.3.11	Windows		

PREMISS		EcoProp	
B	**PERFORMANCE REQUIREMENTS**		
B.1	***Indoor Condition Requirements***		
B.1.1	IndoorClimate		
B.1.1.1	MaxCO	B1.1	Carbon monoxide (CO)
B.1.1.2	MaxCO2	B1.1	Carbon dioxide (CO2)
B.1.1.3	MaxH2CO	B1.1	Formaldehyde (H2CO)
B.1.1.4	MaxNH3	B1.1	Ammonia and amines (NH3)
B.1.1.5	MaxO3	B1.1	Ozone (O3)
B.1.1.6	MaxOdorIntensity	B1.1	Odor intensity (intensity scale)
B.1.1.7	MaxMicrobes	B1.1	Microbes
B.1.1.8	MaxParticles	B1.1	Mass concentration of airborne particulate matter (PM10)
B.1.1.9	MaxRadon	B1.1	Radon (Rn)
B.1.1.10	MaxTVOC	B1.1	Volatile organic compounds(TVOC)
B.1.1.11	NaturallyVentilated		
B.1.1.12	AllowedTemporaryDeviation	B1.1	Temporary deviation from set value
B.1.1.13	IndividualRoomTemperatureControl	B1.1	Individual control of room temperature - Winter
		B1.1	Individual control of room temperature - Summer
		B5	Occupant control of heating, cooling, lighting and ventilation
B.1.1.14	MaxAirVelocity	B1.1	Air velocity - Winter (20øC)
		B1.1	Air velocity - Winter (21øC)
		B1.1	Air velocity - Summer (24øC)
B.1.1.15	MaxFloorTemperature	B1.1	Floor temperature
B.1.1.16	MaxHumidity	B1.1	Relative Humidity - Winter
B.1.1.17	MaxHvacNoiseLevel	B1.2	Equipment sound level LA,eq,T (db), sick room etc.
		B1.2	Equipment sound level LA,eq,T (db), class/office etc.
		B1.2	Equipment sound level LA,max (db), sickrooms
B.1.1.18	MaxTemperature	B1.1	Room temperature - Summer
B.1.1.19	MaxVerticalTemperatureDifference	B1.1	Vertical temperature difference
B.1.1.20	MinAirflowPerPerson	B1.1	Normal occupancy (no smoking, low-emitting materials)
B.1.1.21	MinFloorTemperature	*B1.1*	*Floor temperature*
B.1.1.22	MinHumidity	*B1.1*	*Relative Humidity - Winter*
B.1.1.23	MinNoOccupancyAirChangeRate	B1.1	Basic air change rate when no occupancy
B.1.1.24	MinTemperature	B1.1	Room temperature - Winter
B.1.1.25	TemporarilyVentilationControl	B1.1	Possibility to increase ventilation in each space
		B5	*Occupant control of heating, cooling, lighting and ventilation*
B.1.1.26	MaxEnvelopeAirLeakage	B1.1	The air leakage value of the building envelope < 3 stories
		B1.1	The air leakage value of the building envelope ≥ 3 stories
B.1.1.27	EnvelopeVentilation		
B.1.2	Acoustics		
B.1.2.1	AudioSystem		
B.1.2.2	MinImpactSoundInsulation	B1.2	Footfall sound level figure I'n,w(db), dining room
B.1.2.3	MinUnitSoundInsulation		
B.1.2.4	BackGroundSound		
B.1.2.5	MaxReverberationTime	B1.2	Reverberation, T (s), stairwell, corridor
		B1.2	Reverberation, T(s), dining room
B.1.2.6	MinReverberationTime		
B.1.2.7	MinSoundInsulation		
B.1.2.8	MaxTrafficNoiseLevel		
B.1.2.9	MinEnvelopeSoundInsulation		

PREMISS				EcoProp	
B	**PERFORMANCE REQUIREMENTS**				
	B.1	***Indoor Condition Requirements***			
	B.1.3	Lighting			
		B.1.3.1	ColorRenderingIndex	B1.3	Color rendering
		B.1.3.2	ContrastReproduction	B1.3	Contrast repetition/reproduction CRF
		B.1.3.3	Darkenable	B5	Possibility of darkness
		B.1.3.4	Daylight	B1.3	Daylight
				B5	Daylight in common rooms
		B.1.3.5	DirectionalLighting	B1.3	Directional lighting of visual tasks
		B.1.3.6	GlareIndex	B1.3	Glare (IES-IND)
		B.1.3.7	LightingAdjustability	B1.3	Adjustability
				B5	*Occupant control of heating, cooling, lighting and ventilation*
		B.1.3.8	LightingUniformity	B1.3	Uniformity
		B.1.3.9	LuminanceDistribution	B1.3	Luminance distribution
		B.1.3.10	LusterReflection	B1.3	Brightness/shine/luster reflection
		B.1.3.11	MaxColorTemperature	B1.3	Color appearance (Color temperature)
		B.1.3.12	MaxLuminance	B1.3	Recommended illuminances at the task area
		B.1.3.13	MinColorTemperature	*B1.3*	*Color appearance (Color temperature)*
		B.1.3.14	MinLampEnergyEfficiency	B1.3	Energy considerations (Energy efficiency)
		B.1.3.15	MinLuminance	*B1.3*	*Recommended illuminances at the task area*
		B.1.3.16	NoDaylight		
		B.1.3.17	ShadowFormation	B1.3	Modeling (Shadow formation)
		B.1.3.18	TaskLighting		
	B.2	***Service Life Requirements***			
	B.2.1	ServiceLifeOfBuilding			
		B.2.1.1	BuildingServiceLife	B2	Building design/planning
		B.2.1.2	EnvelopeServiceLife	B2	Service life of major functional elements (eg. shell cladding)
		B.2.1.3	StructureServiceLife	B2	Service life of load bearing structure
	B.2.2	ServiceLifeOfTechicalSystems			
		B.2.2.1	AudioSystemServiceLife		
		B.2.2.2	AutomationCableServiceLife	B2	HVAC-EL automation cabling
		B.2.2.3	AutomationControlsServiceLife	B2	HVAC-EL-automation systems (control devices)
				B2	Ventilation and AC operation, metering, and control devices
		B.2.2.4	DuctServiceLife	B2	Ventilation/air conditioning duct
		B.2.2.5	ElectricalCableServiceLife		
		B.2.2.6	ElectricalFittingsServiceLife		
		B.2.2.7	ElevatorServiceLife		
		B.2.2.8	EscalatorServiceLife		
		B.2.2.9	FireSafetySystemServiceLife		
		B.2.2.10	GasSystemServiceLife		
		B.2.2.11	HeatingDistributionSystemServiceLife	B2	Water circulation heat distribution machinery
		B.2.2.12	HeatMachineryServiceLife	B2	Heat yield machinery (boilers, accumulators)
		B.2.2.13	ItCableServiceLife		
		B.2.2.14	LightSourceServiceLife	B1.3	Maintenance factor (Light serviceability/maintainability)
		B.2.2.15	NonVisiblePipingServiceLife	B2	Inconveniently replaceable piping
				B2	Service life of components where replacement is expensive
		B.2.2.16	PlumbingSystemServiceLife	B2	Water plumbing system components
		B.2.2.17	PumpAndFanServiceLife	B2	HVAC pumps, fans
		B.2.2.18	RadiatorServiceLife	B2	HVAC equipment/machine heat transfer-element/installment
		B.2.2.19	SecuritySystemServiceLife		
		B.2.2.20	SewerSystemServiceLife	B2	Sewer system plumbing and componenets.
		B.2.2.21	TelecomCableServiceLife		
		B.2.2.22	VisiblePipingServiceLife	B2	Easily replaceable piping
		B.2.2.23	WaterSystemServiceLife	B2	Water and sewer fittings (wash basins, WC-seat, bath)
				B2	Water plumbing system components (control valve, mixers)

PREMISS		EcoProp	
B	PERFORMANCE REQUIREMENTS		
B.3	***Adaptability Requirements***		
B.3.1	FlexibilityOfBuilding		
B.3.1.1	BuildingFlexibility	B3	Changing the purpose of use in the building
B.3.1.2	DesignFlexibility	B3	Intial user's possibility to make individual choices
B.3.1.3	EnvelopeFlexibility		
B.3.1.4	Expandability	B3	Expandability
B.3.1.5	FloorFlexibility	B3	Structural system - Floor structures
B.3.1.6	FrameFlexibility	B3	Structural sytem - Frame
B.3.1.7	OccupancyFlexibility	B6	Flexibility in Use
B.3.1.8	PartitionFlexibility	B3	Structural - Space system, removability of separating walls
B.3.2	FlexibilityOfTechnicalSystems		
B.3.2.1	AudioSystemFlexibility		
B.3.2.2	BuildingAutomationFlexibility	B3	Automatics, IT systems
B.3.2.3	ElectricalInstallationFlexibility	B3	Distribution of electricity system
B.3.2.4	ElectricalSystemFlexibility		
B.3.2.5	ElevatorFlexibility		
B.3.2.6	EscalatorFlexibility		
B.3.2.7	FireSafetySystemFlexibility	B3	Fire alarm system
B.3.2.8	GasSupplyFlexibility		
B.3.2.9	HeatingSystemFlexibility	B3	Heating system
B.3.2.10	HorizontalFlexibility		
B.3.2.11	HvacSystemFlexibility	B3	Ventilation system, routing, surrounding structures
B.3.2.12	IlluminationFlexibility	B3	Illumination system
B.3.2.13	ItNetworkFlexibility		
B.3.2.14	SecuritySystemFlexibility	B3	Security system, passage control, video control
B.3.2.15	SprinklerFlexibility		
B.3.2.16	TelecomSystemFlexibility	B3	(Tele)communications system
B.3.2.17	VerticalFlexibility		
B.3.2.18	WaterSupplyFlexibility	B3	Water supply system
B.3.3	FlexibilityOfSpace		
B.3.3.1	AlternativeFurnishing	B3	Alternative furnishing of spaces
B.3.3.2	AlternativeUse	B3	Alternative use and dimensioning of spaces
B.3.3.3	DivisionAndCombination	B3	Division and combination of spaces
B.4	***Safety Requirements***		
B.4.1	SafetyOfSite		
B.4.1.1	SiteSecurity	B4.4	Area
B.4.1.2	MonitoringOfSite		
B.4.1.3	PerimeterControl		
B.4.1.4	ProtectionFromAttack		
B.4.1.5	ControlOfParking		
B.4.1.6	ProtectionOfVehicles		
B.4.2	SafetyOfBuilding		
B.4.2.1	BuildingSecurity	B4.4	Building
B.4.2.2	BuildingAccessControl	*B4.4*	*Building*
B.4.2.3	SeparationOfZones		
B.4.2.4	FireResistanceRating	B4.2	Fire-resistance class
B.4.2.5	FireResistanceTime	B4.2	Fire-resistance time
B.4.2.6	FireSafetySystem	B4.2	Extinguishing systems
B.4.2.7	LoadCapacity	B4.1	Bearing/load capacity
B.4.2.8	SurfaceFirePropagation	B4.2	Surface layer fire-propagation class
B.4.2.9	SurfaceInflammabilityRating	B4.2	Surface layer inflammability class
B.4.2.10	FireRatingForFittings	B4.2	Functional element and accessories technical fire classes
B.4.2.11	AirIntakeLocation		

PREMISS		EcoProp	
B	**PERFORMANCE REQUIREMENTS**		
B.4	***Safety Requirements***		
B.4.3	SafetyOfTechnicalSystems		
B.4.3.1	AudioSystemReliability	B2	Reliability/availability requirements
B.4.3.2	ElectricalBackupSystem		
B.4.3.3	ElectricalReliability	*B2*	*Reliability/availability requirements*
B.4.3.4	ElevatorReliability	*B2*	*Reliability/availability requirements*
B.4.3.5	EscalatorReliability	*B2*	*Reliability/availability requirements*
B.4.3.6	FireSafetySystemReliability	*B2*	*Reliability/availability requirements*
B.4.3.7	GasSupplyReliability	*B2*	*Reliability/availability requirements*
B.4.3.8	HvacReliability	*B2*	*Reliability/availability requirements*
B.4.3.9	ItNetworkBackupTime		
B.4.3.10	ItNetworkReliability	*B2*	*Reliability/availability requirements*
B.4.3.11	ItNetworkSecurity		
B.4.3.12	SecuritySystemReliability	*B2*	*Reliability/availability requirements*
B.4.3.13	SewerFloodingPrevention		
B.4.3.14	TelecomBackupTime		
B.4.3.15	TelecomReliability	*B2*	*Reliability/availability requirements*
B.4.4	SafetyOfStorey		
B.4.4.1	StoreyEnvelopeSecurity	*B4.4*	*Building*
B.4.4.2	StoreyDoorSecurity	*B4.4*	*Building*
B.4.4.3	StoreyWindowSecurity	*B4.4*	*Building*
B.4.5	SecurityOfSpace		
B.4.5.1	AccessControl	B4.4	Room/space
B.4.5.2	AccessZone	B4.4	Space groups
B.4.6	AccidentAndCatastropheRisks		
B.4.6.1	AccidentRisks	B4.5	Radiation accident
		B4.5	Toxic substance leak
B.4.6.2	CatastropheRisks	B4.5	Earthquake
		B4.5	Volcanic (eruption)
		B4.5	Flood
		B4.5	Storms
		B4.5	Snow
		B4.5	Bush fire
B.4.6.3	OtherRisks	B4.3	Safety against slipping
		B4.3	Security of information systems
		B4.3	Falling safety
		B4.3	Collision risks
		B4.3	Burn risk
		B4.3	Electrical shock risk
		B4.3	Malfunction safety
		B4.3	Radiation safety
		B4.3	Yard-areas
B.5	***Aesthetic Requirements***		
B.5.1	VisualRequirements		
B.5.1.1	AestheticAppearance	B5	Aesthetics
		A1.1	Perceptiveness
B.5.1.2	WayFinding	B5	Orientability
		B6	Simple and Intuitive Use
B.5.1.3	AestheticEnvelopeRequirements	*B5*	*Aesthetics*
B.5.1.4	InteriorDesignAndFunctionality	B5	Functionality and comfort of main spaces
		B5	Functionality and comfort of supporting spaces
		B5	Interior design and furniture
B.5.1.5	InternalVisualContacts	B5	Visual contact, internally and with the external world
B.5.1.6	ExternalVisualContacts	*B5*	*Visual contact, internally and with the external world*

PREMISS				EcoProp	
B	**PERFORMANCE REQUIREMENTS**				
	B.6	***Accessibility Requirements***			
		B.6.1	BuildingAccessibility		
		B.6.1.1	AccessibilityForHandicapped	B6	Equitable use (Building is applicable for disable/handicapped)
		B.6.1.2	AccessibilityForHearingImpared	B6	Applicability and suitability for sight and aural disabilities
		B.6.1.3	AccessibilityForSightDisabled	*B6*	*Applicability and suitability for sight and aural disabilities*
		B.6.1.4	ElevatorRequirements		
		B.6.2	StoreyAccessibility		
		B.6.2.1	StoreyAccess		
	B.7	***Circulation Requirements***			
		B.7.1	CirculationArea		
		B.7.1.1	MaxCirculationAreaRatio		
		B.7.2	CirculationSystems		
		B.7.2.1	LobbyRequirements		
		B.7.2.2	CorridorRequirements		
		B.7.2.3	StairRequirements		
		B.7.2.4	ElevatorRequirements		
		B.7.2.5	EscalatorRequirements		
		B.7.2.6	LoadingDockRequirements		
C	**COST REQUIREMENTS**				
	C.1	***Life Cycle Cost Requirements***			
		C.1.1	CostRequirements		
		C.1.1.1	ConstructionCosts		
		C.1.1.2	DesignAndCMCosts		
		C.1.1.3	InvestmentCosts	C1.1	Investment/initial costs
		C.1.1.4	SiteCosts		
		C.1.1.5	EnergyCosts	C1.2	Energy costs
		C.1.1.6	MaintenanceCosts	C1.3	Service and maintenance costs
		C.1.1.7	OperationCosts	C1.2	Operation costs
		C.1.1.8	DisposalCosts	C1.4	Disposal and value
		C.1.1.9	RecycleValue	*C1.4*	*Disposal and value*
D	**ENVIRONMENTAL REQUIREMENTS**				
	D.1	***Sustainability Requirements***			
		D.1.1	EnergyInsulations		
		D.1.1.1	BaseFloorInsulation		
		D.1.1.2	EnergySavingBufferSpaces	B5	Exploitation of 'half-warm' spaces for energy saving
		D.1.1.3	ExternalDoorInsulation		
		D.1.1.4	ExternalWallInsulation		
		D.1.1.5	RoofInsulation		
		D.1.1.6	SolarProtection	B5	Use of solar protection/screen
		D.1.1.7	WindowInsulation		
		D.1.1.8	WindowShadingCoefficient		
		D.1.2	EnergyRequirements		
		D.1.2.1	CoolingEnergyConsumption		
		D.1.2.2	HeatingEnergyConsumption	C2.2	Heating energy consumption
		D.1.2.3	HeatingEnergySource	C2.2	Heating power
		D.1.2.4	LightingEnergyConsumption		
		D.1.2.5	RecycledEnergy		
		D.1.2.6	RenewableEnergyRatio		
		D.1.2.7	TotalElectricalEnergyConsumption	C2.2	Electrical energy consumption
		D.1.2.8	TotalEnergyConsumption		
		D.1.2.9	TotalHvacEnergyConsumption		
		D.1.2.10	WaterConsumption	C2.2	Water consumption
		D.1.3	EnvironmentalPressure		
		D.1.3.1	MaxC2H4eqEmissions	C2.3	C2H4eq emissions
		D.1.3.2	MaxCO2eqEmissions	C2.3	CO2eq emissions
		D.1.3.3	MaxNonRenewableMaterials	C2.3	Non-renewable material
		D.1.3.4	MaxSO2eqEmissions	C2.3	SO2eq emissions
		D.1.3.5	MinRenewableMaterials	C2.3	Renewable material
		D.1.3.6	ProductionEfficiency	C2.3	Production and distribution efficiency

Appendix B4: EcoProp Categories and Rejected Requirements

Table 16 documents all the EcoProp *Requirements Categories* and the EcoProp *Requirements* which are not included in the PREMISS *Requirements Model Specification*. The rejected *Requirements* are redundant with some existing *Requirements*, although in some cases the name of the *Requirement* can be misleading. For example, in "Building extension design" the detailed description is about daylight *Requirements* for *Spaces* which are far from the envelope in the center part of a building [*EcoProp* [118]].

Table 16: EcoProp *Requirements Categories* and rejected *Requirements*
Source: EcoProp software by VTT Building and Transport

Gategories				Requirement	Reason for rejection
A CONFORMITY					
	A1 LOCATION				
		A1.1 Site characteristics			
			A1.1	Availability of infrastructure (urban, not urban)	Redundant with the detailed infrastructure requirements
		A1.2 Transportation			
			A1.2	Company initiative options (eg. company sponsored bus services)	Not relevant for design
			A1.2	Efficient use of company cars	Not relevant for design
			A1.2	Efficient use of deliveries etc.	Not relevant for design
		A1.3 Impacts on surroundings			
	A2 SPATIAL SYSTEMS				
			A2	Building maintenance and care	PREMISS system is open to any spatial grouping, the space types are not predefined
			A2	Business premises	
			A2	Circulation spaces	
			A2	Communal spaces, entry	
			A2	Cooking	
			A2	External spaces	
			A2	Internal circulation spaces, staircases	
			A2	Office and work premises	
			A2	Parking	
			A2	Reserve and storage	
			A2	Special spaces :eg. shop, workshop, laboratory spaces	
			A2	Telework space/room	
			A2	Welfare spaces	
	A3 SERVICES				
B PERFORMANCE					
	B1 INDOOR CONDITIONS				
		B1.1 Indoor climate			
			B1.1	Cigarette smoke in rooms for non-smokers	Defined by local building codes
		B1.2 Acoustics			
			B1.2	Building form	Not a acoustical requirement
			B1.2	Commissioning	Process, but not acoustical, requirements
			B1.2	Detailed design	
			B1.2	Retrofit	
			B1.2	Supervision	
		B1.3 Illumination			
			B1.3	Illuminances of immediate surroundings	Redundant with uniformity requirement
			B1.3	Light control grading	Redundant with adjustability requirement
			B1.3	Light disruption/interference	The meaning not clear
			B1.3	Luminaire luminance limits with downward flux	Redundant with luminance requirements
			B1.3	Shielding against glare	Redundant with glare index and luster reflection requirements
			B1.3	Veiling reflections and reflected glare	
		B1.4 Vibration conditions			

Gategories		Requirement	Reason for rejection
B2 SERVICE LIFE AND DETERIORATION RISK			
	B2	Damage prevention	System specific information, which can be included in reliability requirements
B3 ADAPTABILITY			
	B3	Waste disposal system	Flexibility of waste disposal system is not relevant
B4 SAFETY			
B4.1 Structural safety			
	B4.1	Stability	Structural requirements defiined by the local building codes
	B4.1	Stiffness	
B4.2 Fire safety			
B4.3 Safety in use			
B4.4 Intrusion safety			
B4.5 Natural catastrophes			
B5 COMFORT			
	B5	Aspects of spaces and surfaces (colour, texture, regularity, etc.)	Redundant with detailed spatial requirements
	B5	Building extension design	Redundant with daylight requirements
	B5	Building(s)	Redundant with aestethic requirements
	B5	Connection to surroundings	Redundant with aestethic and visual requirements
	B5	Dynamic requirements	Redundant with accessibility requirements
	B5	Natural and artificial lighting (illuminance, glare, luminance, etc.)	Redundant with detailed lighting requirements
	B5	Outdoor area comfort and usability, green architecture	Redundant with detailed site design requirements
	B5	Stress/pressures	Redundant with detailed spatial requirements
	B5	Tactile requirements	Redundant with detailed spatial requirements
	B5	The openings of spaces	Redundant with location and visual requirements
	B5	Townscape's presence/representativeness	Redundant with aestethic requirements
	B5	User experiences	Redundant with aestethic and visual requirements
B6 ACCESSIBILITY			
	B6	Low Physical Effort (Fittings and furniture)	Redundant with handicapped accessibility requirements
	B6	Size and Space for Approach and Use	Redundant with detailed spatial requirements
	B6	Tolerance for Error	Redundant for risk requirements
	B6	Usability	Redundant with functionality requirements
B7 USABILITY			
C COST AND ENVIRONMENTAL PROPERTIES			
C1 LIFE CYCLE COSTS			
C1.1 Investment costs			
C1.2 Operation costs			
	C1.2	Caretaking/janitor	Redundant with operation cost requirements
C1.3 Maintenance costs			
C1.4 Demolition and disposal costs			
C2 ENVIRONMENTAL PRESSURE			
C2.1 Biodiversity			
C2.2 Resources			
C2.3 Emissions			

Appendix B5: Serviceability Tools by International Centre for Facilities

Tables in Appendix 5 document the topics in the Whole Building Functionality and Serviceability (WBFS) system by the International Centre for Facilities [*ICF 2000* [119]]. The WBFS system divides these topics into two groups. The first is either "Occupant Requirement Scale" or "Facility Requirement Scale" depending on the *Requirements* and the other is "Facility Rating Scale." Each of these topics include detailed descriptions of the required features on a scale from 9 to 1. The WBFS system often combines several detailed *Requirements* under one topic. Thus, the direct comparison to the PREMISS *Requirements* is difficult (Section 3.2.2.1). However, I have tried to identify the corresponding PREMISS *Requirements* in the columns on the right.

Table 17: WBFS System: Occupant requirement and facility rating topics
Source: International Centre for Facilities: Whole Building Functionality and Serviceability [*ICF 2000* [120]], comparison to PREMISS *Requirements*.

A.1.	**Support for Office Work**			**PREMISS**	
	A.1.1.	Photocopying		A.4.2.1	Activities
		Occupant requirement topics			
			Access to copiers	A.4.1.1	AdjacentSpaces
			Location of copiers	A.4.1.4	RequestedLocation
			Disruption of copiers	A.4.1.1	AdjacentSpaces
		Facility rating topics			
			Power supply	B.4.3.3	ElectricalReliability
			Small table-top copiers	A.4.3.5	Equipment
			Convenience copiers	A.4.3.5	Equipment
			Large copiers	A.4.3.5	Equipment
	A.1.2.	Training rooms, general		A.4.2.1	Activities
		Occupant requirement topics			
			Room sizes	A.4.1.10	StandardRequiredArea
			Occupant comfort		Combination of requirements
			Location of rooms	A.4.1.4	RequestedLocation
		Facility rating topics			
			Mix, quantity, future capability	A.4	Combination of detailed space requirements
			Environment		Combination of requirements
			Acoustic control	B.1.2	Combination of acoustical requirements
			Fixtures and fixed equipment	A.4.3	Combination of fixtures and equipment
			Breakout/syndicate rooms		Requirement for additional spaces
			Floorplate and access		Combination of wayfinding & flexibility

A.1.	**Support for Office Work**			**PREMISS**	
	A.1.3.	Training rooms for computer skills		A.4.2.1	Activities
		Occupant requirement topics			
			Room sizes	A.4.1.10	StandardRequiredArea
			Occupant comfort		Combination of requirements
			Location of rooms	A.4.1.4	RequestedLocation
		Facility rating topics			
			Quantity, location, future capability	A.4	Combination of detailed space requirements
			Environment		Combination of requirements
			Acoustic control	B.1.2	Combination of acoustical requirements
			Fixtures and fixed equipment	A.4.3	Combination of fixtures and equipment
			Information technology		Combination of IT network and equipment
			Floorplate and access		Combination of wayfinding & flexibility
	A.1.4.	Interview rooms		A.4.2.1	Activities
		Occupant requirement topics			
			Frequency of use	A.4.1.14	UseHoursPerDay
			Visual and speech privacy	B.1.2	Acoustical requirements
			Location in office	A.4.1.4	RequestedLocation
			Future expansion	B3	Flexibility requirements
			Safety		Combination of safety requirements
		Facility rating topics			
			Present and potential quantity of interview rooms	A.4.1.8	NumberOfSpaceUnits
			Ventilation	B.1.1	Indoor climate requirements
			Enclosure and speech privacy	B.1.2	Combination of acoustical requirements
			Access and physical protection		Combination of safety requirements
	A.1.5.	Storage and floor loading		A.4.2.1	Activities
		Occupant requirement topics			
			Office floor storage		Combination of area and function requirements
			Office floor goods movement	B7	Combination of circulation system requirements
			Off the floor storage		Combination of area and function requirements
			Off the floor goods movement	B7	Combination of circulation system requirements
		Facility rating topics			
			Floor load capacity on office floor	A.4.2.3	SpecialLoadRequirements
			Storage off office floors, including in basement		Combination of area and function requirements
			Access to storage off office floors, including basement		Combination of area and function requirements
			Goods handling to and in storage off office floors		Combination of area and function requirements

A.1.	**Support for Office Work**			**PREMISS**	
	A.1.6.	Shipping and receiving		A.4.2.1	Activities
		Occupant requirement topics			
			Dock capacity	B.7.2.6	LoadingDockRequirements
			Goods movement	B7	Combination of circulation system requirements
			Protection of goods	B7	Combination of safety requirements
			Courier parking	A.4.2.2	FunctionRequirements
		Facility rating topics			
			Loading dock	B.7.2.6	LoadingDockRequirements
			Truck loading capacity	B.7.2.6	LoadingDockRequirements
			Holding area at loading dock	B.7.2.6	LoadingDockRequirements
			Elevator access	B.6.1.4	ElevatorRequirements
			Couriers	A.4.2.2	FunctionRequirements
A.2.	**Meetings and Group Effectiveness**			**PREMISS**	
	A.2.1.	Meeting and conference rooms		A.4.2.1	Activities
		Occupant requirement topics			
			Quantity and size of the rooms	A.4	Combination of detailed space requirements
			Location in office	A.4.1.4	RequestedLocation
			Frequency of meetings	A.4.1.14	UseHoursPerDay
			Privacy and freedom from distraction	B.1.2	Combination of acoustical requirements
			Audio visual aids	A.4.3.6	AvEquipment
		Facility rating topics			
			Mix, quantity	A4	Combination of detailed space requirements
			Floorplate and access		Combination of wayfinding & flexibility
			Acoustic control	B.1.2	Combination of acoustical requirements
			Environment		Combination of requirements
			Fixtures and fixed equipment	A.4.3	Combination of fixtures and equipment
	A.2.2.	Informal meetings and interaction		A.4.2.1	Activities
		Occupant requirement topics			
			Value to organization		Requirements intent
			Purpose of meeting and interaction		Requirements intent
			Participants in meetings and interaction		Combination of function and circulation req.
		Facility rating topics			
			Internal circulation node(s)		Combination of function and circulation req.
			Entrance node(s)		Combination of function and circulation req.
			Pause area(s)		Combination of function and circulation req.
			Food and public facilities		Combination of function and circulation req.

A.2.	**Meetings and Group Effectiveness**			**PREMISS**	
	A.2.3.	Group layout and territory		A.4.2.1	Activities
		Occupant requirement topics			
			Workgroup participation	A.4.2.2	FunctionRequirements
			Formation and duration of groups	B.3.3.3	DivisionAndCombination
			Workgroup size	A.4.1.5	MaxOccupancyNumber
			Configuration of workspaces	A.4.2.2	FunctionRequirements
			Separation of workgroups		Combination of requirements
		Facility rating topics			
			Layout for efficient group work		Combination of flexibility and circulation req.
			Layout for various group sizes		Combination of flexibility and circulation req.
			Environmental control		Combination of requirements
			Separation		Combination of safety and circulation req.
			Legibility of boundaries and territory		Combination of requirements
	A.2.4.	Group workrooms		A.4.2.1	Activities
		Occupant requirement topics			
			Workrooms required		Combination of requirements
			Audio visual and display	A.4.3.6	AvEquipment
			Security and privacy		Combination of safety and acoustical req.
			Layout of group workplaces	A.4.2.2	FunctionRequirements
		Facility rating topics			
			Group or project workroom(s)		Combination of requirements
			Acoustic separation for information security		Combination of safety and acoustical req.
			Environment		Combination of requirements
			Fixtures and fixed equipment	A.4.3	Combination of fixtures and equipment
			Access from individual workstations		Combination of safety and circulation req.
A.3.	**Sound and Visual Environment**			**PREMISS**	
	A.3.1.	Privacy and speech intelligibility			
		Occupant requirement topics			
			Speech privacy in workstation	B.1.2.3	MinUnitSoundInsulation
			Understanding speech in workstation	B.1.2.5	MaxReverberationTime
		Facility rating topics			
			Confidentiality	B.1.2.3	MinUnitSoundInsulation
			Background sound for speech privacy	B.1.2.4	BackGroundSound
			Speech intelligibility	B.1.2.5	MaxReverberationTime
	A.3.2.	Distraction and disturbance			
		Occupant requirement topics			
			Concentration on work	B.1.2	Combination of acoustical requirements
			Freedom from distractions	B.1.2	Combination of acoustical requirements
			Tolerance for overheard conversations	B.1.2	Combination of acoustical requirements

A.3.	**Sound and Visual Environment**			**PREMISS**	
	A.3.2.	Distraction and disturbance			
		Facility rating topics			
			Office noise	B.1.2	Combination of acoustical requirements
			Background sound as a means of masking distracting noise	B.1.2.4	BackGroundSound
			External noise	B.1.2	Combination of acoustical requirements
			Distracting conversations	B.1.2	Combination of acoustical requirements
			Reflected sound	B.1.2.5	MaxReverberationTime
			Movement of people	B7	Combination of circulation system requirements
	A.3.3.	Vibration			
		Occupant requirement topics			
			Tolerance of vibration	A.4.2.4	VibrationControl
		Facility rating topics			
			Movement due to people or equipment	B7	Combination of circulation system requirements
			Vibration from machines or vehicles	A.4.2.4	VibrationControl
	A.3.4.	Lighting and glare			
		Occupant requirement topics			
			Lighting levels to suit work	B.1.3	Combination of lighting requirements
			Tolerance of lighting defects	B.1.3	Combination of lighting requirements
		Facility rating topics			
			Illumination level	B.1.3	Combination of lighting requirements
			Visual defects	B.1.3	Combination of lighting requirements
			Glare	B.1.3.6	GlareIndex
	A.3.5.	Adjustments of lighting by occupants			
		Occupant requirement topics			
			Adjusting for type of work	B.1.3.7	LightingAdjustability
			Occupant lighting control	B.1.3.7	LightingAdjustability
			Task lighting requirement	B.1.3.18	TaskLighting
			Window covering adjustment	B.1.3.3	Darkenable
		Facility rating topics			
			Control of ceiling lights	B.1.3.7	LightingAdjustability
			Relocation of ceiling lights	B.3.2.12	IlluminationFlexibility
			Window coverings	B.1.3.3	Darkenable
			Power for task lights	B.1.3.18	TaskLighting
	A.3.6.	Distant and outside views			
		Occupant requirement topics			
			View from workplace	B.5.1.6	ExternalVisualContacts
			Seeing to a distance	B.5.1.5	InternalVisualContacts
		Facility rating topics			
			Relaxation of eyes	B.5.1.5	InternalVisualContacts
			View to outside	B.5.1.6	ExternalVisualContacts

A.5.	**Typical Office Information Technology**			**PREMISS**	
	A.5.1.	Office computers and related equipment			
		Occupant requirement topics			
			Location of workplaces	A.4.1.4	RequestedLocation
			Quality workplace environment		Combination of requirements
			Electronic equipment at the workstation		Combination of requirements
		Facility rating topics			
			Zones for high density of 'equipment		Combination of requirements
			HVAC services	B.1.1	Combination of indoor climate requirements
			Illumination	B.1.3	Combination of lighting requirements
			Acoustic control	B.1.2	Combination of acoustical requirements
	A.5.2.	Power at workplace			
		Occupant requirement topics			
			Location of available power	B.3.2.3	ElectricalInstallationFlexibility
			Plug-in points at workstation	B.3.2.3	ElectricalInstallationFlexibility
			Protection from power fluctuation	B.4.3.3	ElectricalReliability
		Facility rating topics			
			Power distribution	B.3.2.3	ElectricalInstallationFlexibility
			Plug-in points per workplace	B.3.2.3	ElectricalInstallationFlexibility
			Uninterruptible power supply (UPS)	B.4.3.2	ElectricalBackupSystem
	A.5.3.	Building power			
		Occupant requirement topics			
			Power for equipment at workstation	B.3.2.3	ElectricalInstallationFlexibility
			Power for future equipment	B.3.2.4	ElectricalSystemFlexibility
			Reliability and quality of supply	B.4.3.3	ElectricalReliability
		Facility rating topics			
			Present capacity		NA
			Potential increase	B.3.2.4	ElectricalSystemFlexibility
			Reliability and quality of supply	B.4.3.3	ElectricalReliability
	A.5.4.	Data and telephone systems			
		Occupant requirement topics			
			Quantity and location of cabling		Combination of IT and telecom network flexibility
			Access to cable distribution system		Combination of IT and telecom network flexibility
			Installation of local area network		Combination of IT and telecom network flexibility
			Spare capacity in cable routes	A.4	Spatial requirement
			Data cable shielding		NA
		Facility rating topics			
			Distribution		Combination of IT and telecom network flexibility
			Future capacity		Combination of IT and telecom network flexibility
			Shielding of data cables		NA
			Local area network		Combination of IT and telecom network flexibility
			Rooms for data and telephone connections	A.4	Spatial requirement

A.5.	**Typical Office Information Technology**			**PREMISS**	
	A.5.5.	Cable plant			
		Occupant requirement topics			
			Access to local area network		Combination of IT and telecom network flexibility
			Voice and data connections		Combination of IT and telecom network flexibility
		Facility rating topics			
			Unshielded twisted pair		NA
			Distance to cable connection rooms	A.4.1.4	RequestedLocation
			Coaxial cable		NA
			Fiber optic cable		NA
	A.5.6.	Cooling			
		Occupant requirement topics			
			Cooling capacity for increased electrical loads		NA
		Facility rating topics			
			Increased capacity		NA
A.6.	**Change and Churn by Occupants**			**PREMISS**	
	A.6.1.	Disruption due to physical change			
		Occupant requirement topics			
			Tolerance for disruption	B.3	Combination of flexibility requirem.
			Extent of staff disruption	B.3	Combination of flexibility requirem.
			Disruption of nearby staff	B.3	Combination of flexibility requirem.
		Facility rating topics			
			Disruption during relocation	B.3	Combination of flexibility requirem.
			Disruption to neighboring occupants	B.3	Combination of flexibility requirem.
	A.6.2.	Illumination, HVAC and sprinklers			
		Occupant requirement topics			
			Frequency of layout change	B.3	Combination of flexibility requirem.
			Adjustments due to relocated equipment	B.3	Combination of flexibility requirem.
		Facility rating topics			
			Relocating light fixtures	B.3.2.12	IlluminationFlexibility
			Relocating air diffusers	B.3.2.11	HvacSystemFlexibility
			Special air exhaust	B.3.2.11	HvacSystemFlexibility
			Relocating sprinkler heads	B.3.2.15	SprinklerFlexibility
	A.6.3.	Minor changes to layout			
		Occupant requirement topics			
			Frequency of change	B.3	Combination of flexibility requirem.
			Personnel required to make adjustments	B.3	Combination of flexibility requirem.
			Effects of changes	B.3	Combination of flexibility requirem.
		Facility rating topics			
			Changes in workplace layouts	B.3	Combination of flexibility requirem.
			Consequences of minor changes	B.3	Combination of flexibility requirem.
	A.6.4.	Partition wall relocations			
		Occupant requirement topics			
			Frequency of partition change	B.3.1.8	PartitionFlexibility
			Proportion of partitioned offices	B.3.1.8	PartitionFlexibility
		Facility rating topics			
			Floor to ceiling partition walls	B.3.1.8	PartitionFlexibility
			Extent of salvage	B.3.1.8	PartitionFlexibility

A.6.	**Change and Churn by Occupants**			**PREMISS**	
	A.6.5.	Lead time for facilities group			
		Occupant requirement topics			
			Advance notice of required change		NA
			Allowable time for completing change	B.3	Combination of flexibility requirem.
		Facility rating topics			
			Planning major realignment	B.3	Combination of flexibility requirem.
			Ordering and installation	B.3	Combination of flexibility requirem.
A.7.	**Layout and Building Features**			**PREMISS**	
	A.7.1.	Influence of HVAC on layout			
		Occupant requirement topics			
			Choice of open or closed offices	B.3	Combination of flexibility requirem.
			Constraints on use of closed offices	B.3	Combination of flexibility requirem.
			Constraints on population density	B.3	Combination of flexibility requirem.
		Facility rating topics			
			Type of layout	B.3	Combination of flexibility requirem.
			Location or rooms	B.3	Combination of flexibility requirem.
			Screens and furniture	B.3	Combination of flexibility requirem.
			Population density	B.3	Combination of flexibility requirem.
			Upgrade	B.3	Combination of flexibility requirem.
	A.7.2.	Influence of sound and visual features on layout			
		Occupant requirement topics			
			Tolerance of sound and visual conditions		Combination of requirements
			Avoiding glare on VDU screens	B.1.3.6	GlareIndex
		Facility rating topics			
			Main aisles	B.3	Combination of flexibility requirem.
			Location of workstations	B.3	Combination of flexibility requirem.
			VDU locations	B.3	Combination of flexibility requirem.
			Type of layout	B.3	Combination of flexibility requirem.
			Upgrade	B.3	Combination of flexibility requirem.
	A.7.3.	Influence of building loss features on space needs			
		Occupant requirement topics			
			None for this topic		
		Facility rating topics			
			Usable area lost		Combination of requirements
A.8.	**Protection of Occupant Assets**			**PREMISS**	
	A.8.1.	Control of access from building public zone to Occupant			
		Occupant requirement topics			
			Control of staff and visitor entry	B.4.2.2	BuildingAccessControl
			Control of mail and deliveries		
		Facility rating topics			
			Staffing of entry control station	B4	Combination of safety and security requirements
			Control of elevators	B4	Combination of safety and security requirements
			TV monitoring	B4	Combination of safety and security requirements
			Control of deliveries	B4	Combination of safety and security requirements
			Entry to reception zone	B4	Combination of safety and security requirements

A.8.	Protection of Occupant Assets			PREMISS	
	A.8.2.	Interior zones of security			
		Occupant requirement topics			
			Control of entry to operations zone	B4	Combination of safety and security requirements
			Control of entry to secure zone	B4	Combination of safety and security requirements
		Facility rating topics			
			Operational zone	B4	Combination of safety and security requirements
			Secure zone	B4	Combination of safety and security requirements
	A.8.3.	Vaults and secure rooms			
		Occupant requirement topics			
			Level of protection	B4	Combination of safety and security requirements
		Facility rating topics			
			Location	B4	Combination of safety and security requirements
			Floor loads	B4	Combination of safety and security requirements
			Wall construction	B4	Combination of safety and security requirements
			Doors and hardware	B4	Combination of safety and security requirements
			Ventilation	B4	Combination of safety and security requirements
			Alarms	B4	Combination of safety and security requirements
	A.8.4.	Security of cleaning service systems			
		Occupant requirement topics			
			Security for cleaning secure zones	B4	Combination of safety and security requirements
			Security clearance for cleaning staff		NA
		Facility rating topics			
			Staff security		NA
			Monitoring	B4	Combination of safety and security requirements
	A.8.5.	Security of maintenance service systems			
		Occupant requirement topics			
			Security for maintenance secure zones	B4	Combination of safety and security requirements
			Security clearance for maintenance staff		NA
		Facility rating topics			
			Staff security		NA
			Monitoring	B4	Combination of safety and security requirements

A.8.	**Protection of Occupant Assets**			**PREMISS**	
	A.8.6.	Security of renovations outside active hours			
		Occupant requirement topics			
			Level of protection of occupants assets		NA
			Control of contractor's personnel		NA
			Defining boundaries of work		NA
		Facility rating topics			
			Contractor's staff		NA
			Control of admission		NA
			Temporary enclosure		NA
	A.8.7.	Systems for secure garbage			
		Occupant requirement topics			
			Level of protection for secure wastes		NA
			Handling and disposal of secure waste		NA
		Facility rating topics			
			Storage containers		Combination of requirements
			Location of storage		Combination of requirements
			Separated waste		Combination of requirements
	A.8.8.	Security of key and card control systems			
		Occupant requirement topics			
			Level of protection of occupant premises	B4	Combination of safety and security requirements
			Occupant control of keying		NA
		Facility rating topics			
			Occupant keying system	B4	Combination of safety and security requirements
			Key identification	B.4.3.12	SecuritySystemReliability
			Key distribution		NA
A.9.	**Facility Protection**			**PREMISS**	
	A.9.1.	Protection around building			
		Occupant requirement topics			
			Level of protection from threats		Combination of safety and security requirements
			Possible threats	B.4.6.3	OtherRisks
		Facility rating topics			
			Electronic or acoustic intrusion		Combination of safety and security requirements
			Overview of site	A.2.7.4	SiteSecurity
			Information on activities in neighboring buildings	B.4.6.3	OtherRisks
			Personal safety		Combination of safety and security requirements
	A.9.2.	Protection from unauthorized access to site and parking			
		Occupant requirement topics			
			Protection of site	A.2.7.4	SiteSecurity
			Control of parking use	B.4.1.5	ControlOfParking
			Protection of on-site stored vehicles	B.4.1.6	ProtectionOfVehicles
		Facility rating topics			
			Perimeter control	B.4.1.3	PerimeterControl
			Easements		NA
			Permission for access to site		NA
			Control of access	B.4.1.5	ControlOfParking
			Security of stored vehicles	B.4.1.6	ProtectionOfVehicles

A.9.	**Facility Protection**			**PREMISS**	
	A.9.3.	Protective surveillance of site			
		Occupant requirement topics			
			Level of protection from intruders		Combination of safety and security requirements
			Level of protection of staff and visitors		Combination of safety and security requirements
			After hours and shift work		NA
			Surveillance of intruders	B.4.1.2	MonitoringOfSite
		Facility rating topics			
			Illumination of site	A.2.7.3	SiteLighting
			Monitoring of site	B.4.1.2	MonitoringOfSite
			Patrol of site		NA
			Placement of planting material	A.2.7.4	SiteSecurity
			Selection of planting material	A.2.7.4	SiteSecurity
			Berms and walls	A.2.7.4	SiteSecurity
	A.9.4.	Perimeter of the building			
		Occupant requirement topics			
			Protection from unauthorized entry and attack	B.4.1.4	ProtectionFromAttack
			Avoiding fumes in ventilation air intake	B.4.2.11	AirIntakeLocation
		Facility rating topics			
			Entry from adjacent building(s)	B.4.1.4	ProtectionFromAttack
			Access to roof from adjacent building(s)	B.4.1.4	ProtectionFromAttack
			Access to building	B.4.2.2	BuildingAccessControl
			Doors and windows secure	B.4.4.2	StoreyDoorSecurity
				B.4.4.3	StoreyWindowSecurity
			Air intake location	B.4.2.11	AirIntakeLocation
			Alarm, monitors and guards	B.4.1.2	MonitoringOfSite
	A.9.5.	Public zone of building			
		Occupant requirement topics			
			Control of staff entry outside of active hours	B.4.5.1	AccessControl
				B.4.5.2	AccessZone
			Security of entry to occupant zone	B.4.5.1	AccessControl
				B.4.5.2	AccessZone
			Overflow crowds in reception zone	B.7.2.1	LobbyRequirements
			Separate staff toilets	A.4	Spatial requirement
		Facility rating topics			
			Entry security desk	B.7.2.1	LobbyRequirements
			Separation of public and occupant zones	B.4.2.3	SeparationOfZones
			Support for crowd control	B.7.2.1	LobbyRequirements
			Public toilets	A.4	Spatial requirement
	A.9.6.	Facility Protection Services			
		Occupant requirement topics			
			Protection of services to the building		Combination of security req.
			Protection against threats inside the building		Combination of security req.
		Facility rating topics			
			Locking	B.4.5.1	AccessControl
			Access doors	B.4.5.1	AccessControl
			Alarms		Security system requirement
			External communication routing		Combination of security req.
			Communication redundancy		Combination of security req.

A.10.	**Working Outside Normal Hours or Conditions**			**PREMISS**	
	A.10.1.	Operation outside normal hours			
		Occupant requirement topics			
			Predicting work outside normal hours		NA
			Frequency of work outside normal hours		NA
			Advance notice for activation of services		NA
			Restriction of service to occupied area		NA
		Facility rating topics			
			Operating building		NA
			Lead-time to change operating hours		NA
	A.10.2.	Support after-hours			
		Occupant requirement topics			
			Food service	A.3.1.5	FoodServices
			Access to storage	B.4.5.1	AccessControl
			Security of staff leaving after hours		Combination of security req.
		Facility rating topics			
			Food service	A.3.1.5	FoodServices
			Access to storage	B.4.5.1	AccessControl
			Added physical protection		NA
	A.10.3.	Temporary loss of external services			
		Occupant requirement topics			
			Required standby services		Combination of realiability req.
		Facility rating topics			
			Disruption to occupants		Combination of realiability req.
			Continued occupant operations		Combination of realiability req.
			Standby during loss of external power	B.4.3.2	ElectricalBackupSystem
			Alternative telecommunication services	B.4.3.15	TelecomReliability
	A.10.4.	Continuity of work (during breakdowns)			
		Occupant requirement topics			
			Requirement for continuity of work		Combination of realiability req.
			Tolerance for loss of productivity		Combination of realiability req.
		Facility rating topics			
			Work during breakdown		Combination of realiability req.
			Frequency of breakdowns		Combination of realiability req.
			Duration of breakdowns		Combination of realiability req.
			Loss of productivity		Combination of realiability req.
A.11.	**Image to Public and Occupants**			**PREMISS**	
	A.11.1.	Exterior appearance			
		Occupant requirement topics			
			Appearance	B.5.1.1	AestheticAppearance
			Image	B.5.1.1	AestheticAppearance
		Facility rating topics			
			Overall appearance of building, aesthetics	B.5.1.1	AestheticAppearance
			Condition of exterior surfaces	B.5.1.3	AestheticEnvelopeRequirements
			Approach and entrance	B.5.1.2	WayFinding

A.11. Image to Public and Occupants		PREMISS	
A.11.2.	Public lobby of building		
	Occupant requirement topics		
	Quality of lobby	B.7.2.1	LobbyRequirements
	Standard of signage	B.5.1.2	WayFinding
	Requirement for information desk	B.7.2.1	LobbyRequirements
	Facility rating topics		
	General appearance	B.7.2.1	LobbyRequirements
	Materials and condition		Combination of space req.
	Layout and spaciousness	B.7.2.1	LobbyRequirements
	Interior signage	B.5.1.2	WayFinding
	Staffed information desk	B.7.2.1	LobbyRequirements
A.11.3.	Public spaces within building		
	Occupant requirement topics		
	Quality of public areas		Combination of space req.
	Quality of public washrooms		Combination of space req.
	Facility rating topics		
	Image of public areas		Combination of space req.
	Public circulation routes	B.7.2.2	CorridorRequirements
		B.7.2.3	StairRequirements
		B.7.2.4	ElevatorRequirements
		B.7.2.5	EscalatorRequirements
	Washrooms accessible to the public		Combination of space req.
A.11.4.	Appearance and spaciousness of office spaces		
	Occupant requirement topics		
	Image of office space		Combination of space req.
	Spacious appearance		Combination of space req.
	Facility rating topics		
	Appearance		Combination of space req.
	Sense of spaciousness		Combination of space req.
A.11.5.	Finishes and materials in office spaces		
	Occupant requirement topics		
	Significance of building standards		Combination of space req.
	Facility rating topics		
	Finishes		Combination of space req.
	Window coverings		Combination of space req.
	Hardware and fixtures		Combination of space req.
A.11.6.	Identity outside building		
	Occupant requirement topics		
	Public exposure	B.5.1.1	AestheticAppearance
	Ease of locating and identifying building	B.5.1.2	WayFinding
	Facility rating topics		
	Identity of building	B.5.1.1	AestheticAppearance
	Corporate identity and signage	B.5.1.1	AestheticAppearance
	Quality of external signs	B.5.1.2	WayFinding

A.11.	**Image to Public and Occupants**			**PREMISS**	
	A.11.7.	Neighborhood and site			
		Occupant requirement topics			
			Image of site	A.2.1.3	SiteImage
			Safety of site	B.4.1	SafetyOfSite
			Image of other occupants	A.1.1.1	GeneralObjectives
			Compatibility with other occupants	A.1.1.1	GeneralObjectives
		Facility rating topics			
			Image of neighborhood	A.2.1.3	SiteImage
			Organization and activities in the locality	A.2.1.3	SiteImage
			Site conditions and landscaping	A.2.1.3	SiteImage
			Organization and activities in the building	A.1.1.1	GeneralObjectives
			Compatibility with offices of units of the organization	A.1.1.1	GeneralObjectives
A.12.	**Amenities to Attract and Retain Staff**			**PREMISS**	
	A.12.1.	Food			
		Occupant requirement topics			
			Food facility in the building	A.4	Spatial requirement
			Food facilities in the neighborhood	A.3.1.5	FoodServices
		Facility rating topics			
			On-site service	A.4	Spatial requirement
			Potential for on-site service	A.4	Spatial requirement
			Neighborhood facilities	A.3.1.5	FoodServices
	A.12.2.	Shops			
		Occupant requirement topics			
			Shops available in the facility	A.4	Spatial requirement
			Shops in the neighborhood	A.3.1.2	CommercialServices
		Facility rating topics			
			Existing shops	A.3.1.2	CommercialServices
			Potential for shops in building	A.4	Spatial requirement
			Neighborhood shopping	A.3.1.2	CommercialServices
	A.12.3.	Day care			
		Occupant requirement topics			
			Day care in the facility	A.4	Spatial requirement
			Day care in the neighborhood	A.3.1.4	DayCareServices
		Facility rating topics			
			Existing day care on-site	A.3.1.4	DayCareServices
			Neighborhood facility	A.3.1.4	DayCareServices
	A.12.4.	Exercise room			
		Occupant requirement topics			
			Fitness facilities in the building	A.4	Spatial requirement
			Off-site private sector fitness centre	A.3.1.7	RecreationalServices
		Facility rating topics			
			Existing exercise facilities	A.3.1.7	RecreationalServices
	A.12.5.	Bicycle racks for staff			
		Occupant requirement topics			
			Requirement for racks	A.2.4.2	MinBikeParkingSpaces
			Location of racks	A.4.1.4	RequestedLocation
			Security of bicycles	B.4.1	SafetyOfSite
		Facility rating topics			
			Existing bicycle racks		NA
			Potential for additional bicycle racks		NA
			Risk of theft	B.4.1	SafetyOfSite

A.12.	**Amenities to Attract and Retain Staff**			**PREMISS**	
	A.12.6.	Seating away from work areas			
		Occupant requirement topics			
			Casual seating in public areas	A.2.4.5	SiteAmenities
			Staff lounge in facility	A.4	Spatial requirement
			Potential lounges in occupant space	A.4	Spatial requirement
		Facility rating topics			
			Existing seating		NA
			Potential for seating	A.4	Spatial requirement
			Separate ventilation for smoking areas	A.4	Spatial requirement
A.13.	**Special Facilities and Technologies**				
	A.13.1.	Group or shared conference center			
		Occupant requirement topics			
			Location of meeting space	A.4.1.4	RequestedLocation
			Size of meetings	A.4	Spatial requirement
			Future need for a conference center	B.3.1.4	Expandability
		Facility rating topics			
			Present provision	A.4	Spatial requirement
			Potential space	B.3.1.4	Expandability
			Potential services	B.3.1.4	Expandability
	A.13.2.	Video teleconference facilities			
		Occupant requirement topics			
			Present need for facility	A.4	Spatial requirement
			Future need for facility	B.3.1.4	Expandability
		Facility rating topics			
			Present provision	A.4	Spatial requirement
			Potential space	B.3.1.4	Expandability
			Potential services	B.3.1.4	Expandability
	A.13.3.	Simultaneous translation			
		Occupant requirement topics			
			Present need for translation facility	A.4	Spatial requirement
			Future need for facility	B.3.1.4	Expandability
		Facility rating topics			
			Present provision	A.4	Spatial requirement
			Potential for translation facilities	B.3.1.4	Expandability
	A.13.4.	Satellite and microwave links			
		Occupant requirement topics			
			Present need for link		Telecom requirement
			Future need for link	B.3.2.16	TelecomSystemFlexibility
		Facility rating topics			
			Present provision		Telecom requirement
			Potential for installation	B.3.2.16	TelecomSystemFlexibility
	A.13.5.	Mainframe computer centre			
		Occupant requirement topics			
			Present need for computer centre	A.4	Spatial requirement
			Future need for computer center	B.3.1.4	Expandability
		Facility rating topics			
			Present provision	A.4	Spatial requirement
			Potential for installation	B.3.1.4	Expandability

A.13.	**Special Facilities and Technologies**				
	A.13.6.	Telecommunications centre			
		Occupant requirement topics			
			Immediate need for access to a centre	A.4	Spatial requirement
			Future need for access to a centre	B.3.1.4	Expandability
		Facility rating topics			
			Present provision	A.4	Spatial requirement
			Potential for installation	B.3.1.4	Expandability
A.14.	**Location, Access and Wayfinding**			**PREMISS**	
	A.14.1.	Public transportation (urban sites)			
		Occupant requirement topics			
			Origin of staff and visitors	A.2.1.1	GeographicalLocation
			Proximity to transit routes	A.2.3.6	PublicTransportationDistance
			Frequency of visitors		NA
			Office hours	A.4.1.11	NormalStartTime
				A.4.1.12	NormalEndTime
		Facility rating topics			
			Staff commuting during peak hours	A.2.3.3	CarAccess
			Distance to transit stops	A.2.3.6	PublicTransportationDistance
			Visitors use of public transportation during off-peak hours	A.2.3.5	PublicTransportation
				A.2.3.7	PublicTransportationFrequency
	A.14.2.	Staff visits to other offices			
		Occupant requirement topics			
			Proximity to destination	A.2.1.1	GeographicalLocation
			Access to destination	A.2.3	Combination of transportation req.
		Facility rating topics			
			Location of other offices visited during work	A.2.1.1	GeographicalLocation
			Convenience of access to other sites	A.2.3	Combination of transportation req.
	A.14.3.	Vehicular entry and parking			
		Occupant requirement topics			
			Minimize pedestrian / vehicle accidents		
			Parking at urban sites	A.2.4.3	MinCarParkingSpaces
			Parking at small town or suburban sites	A.2.4.3	MinCarParkingSpaces
		Facility rating topics			
			Separation of pedestrians and vehicles	A.2.4.7	SiteTrafficRequirements
			Separation of cars and trucks	A.2.4.7	SiteTrafficRequirements
			Parking at urban sites	A.2.4.3	MinCarParkingSpaces
			Parking at small town or suburban sites	A.2.4.3	MinCarParkingSpaces
	A.14.4.	Wayfinding to building and lobby			
		Occupant requirement topics			
			Ease of wayfinding to building and lobby	B.5.1.2	WayFinding
			Type of visitors		NA
		Facility rating topics			
			Locating the building	B.5.1.2	WayFinding
			Wayfinding to entry	B.5.1.2	WayFinding
			Visitor drop-off	A.2.4.6	VechicleAccess
			Wayfinding to lobby	B.5.1.2	WayFinding

A.14.	**Location, Access and Wayfinding**			**PREMISS**	
	A.14.5.	Capacity of internal movement systems			
		Occupant requirement topics			
			Accommodation visitor traffic	A.2.4.6	VechicleAccess
			Occupant traffic in building	B.7.2.2	CorridorRequirements
			Convenience of elevator service	B.7.2.4	ElevatorRequirements
		Facility rating topics			
			Visitor traffic in elevators	B.7.2.4	ElevatorRequirements
			Capability to provide for staff traffic in elevators	B.7.2.4	ElevatorRequirements
			Elevators, escalators and stairs	B.7.2.3	StairRequirements
				B.7.2.4	ElevatorRequirements
				B.7.2.5	EscalatorRequirements
			One and two-story buildings	A.2.6.10	PermittedNumberOfFloors
	A.14.6.	Public circulation and wayfinding in building			
		Occupant requirement topics			
			Separation of incompatible visitors	B.4.2.3	SeparationOfZones
			Visitors finding their destination	B.5.1.2	WayFinding
			Convenience of elevator service	B.7.2.4	ElevatorRequirements
			Separating passenger and freight elevator service	B.4.2.3	SeparationOfZones
		Facility rating topics			
			Separation of incompatible groups	B.4.2.3	SeparationOfZones
			Wayfinding to elevators or stairs	B.7.2.4	ElevatorRequirements
			Wayfinding within building	B.5.1.2	WayFinding
			Separation of freight and passengers	B.4.2.3	SeparationOfZones

Table 18: WBFS System: Facility management requirement and facility rating topics
Source: International Centre for Facilities: Whole Building Functionality and Serviceability [*ICF 2000* [121]]

B.1.	**Structure and Building Envelope**			**PREMISS**	
	B.1.1.	Typical office floors			
		Facility management requirement topics			
			Areas for heavy loads	A.4.2.3	SpecialLoadRequirements
			Requirement for level floors		NA
		Facility rating topics			
			Information on allowable loading	A.4.2.3	SpecialLoadRequirements
			Floor load capacity	A.4.2.3	SpecialLoadRequirements
			Levelness and evenness		NA
	B.1.2.	External walls and projections			
		Facility management requirement topics			
			Condition of building external walls		NA
			Evidence of water penetration		NA
		Facility rating topics			
			Permanence of exterior finishes	B.2.1.2	EnvelopeServiceLife
			Water penetration		NA
			Signs of deterioration		NA
			Exterior projections		NA
	B.1.3.	External windows and doors			
		Facility management requirement topics			
			Weather tightness of windows and doors		NA
			Ease of operation of windows and doors		
		Facility rating topics			
			Weather tightness		NA
			Sealants	B.2.1.2	EnvelopeServiceLife
			Defects		NA
	B.1.4.	Roof			
		Facility management requirement topics			
			History of roof leaks		NA
			Anticipated time before repairs needed	B.2.1.2	EnvelopeServiceLife
		Facility rating topics			
			Leaks		NA
			Flashings		NA
			Condition		NA
	B.1.5.	Basement			
		Facility management requirement topics			
			Use of basement		
			Required environmental conditions		NA
			Acceptable physical condition		NA
		Facility rating topics			
			Settling		NA
			Cracking		NA
			Moisture penetration		NA
			Condition of concrete		NA

B.1.	**Structure and Building Envelope**			**PREMISS**	
	B.1.6.	Grounds			
		Facility management requirement topics			
			Required level of ground maintenance		NA
			Acceptable condition of site improvements		NA
		Facility rating topics			
			Paving		
			Landscaping		
			Site drainage		
			Site or street furniture		
B.2.	**Manageability**			**PREMISS**	
	B.2.1.	Reliability of external support			
		Facility management requirement topics			
			Frequency of power outages	B.4.3.3	ElectricalReliability
			Frequency of loss of listed services	B.4	Combination of reliability req.
			Work duration during loss of services		NA
			Need for evacuation		NA
		Facility rating topics			
			Electrical power supply	B.4.3.3	ElectricalReliability
			Building services (except power)	B.4	Combination of reliability req.
	B.2.2.	Anticipated remaining service life (Specified in Table B2.2			
		Facility management requirement topics			
			Remaining service life of building components and systems	B.4	Combination of service life req.
		Facility rating topics			
			Major building components	B.4	Combination of service life req.
	B.2.3.	Ease of operation			
		Facility management requirement topics			
			Storeroom for building operations	A.4	Spatial requirement
			Space for building operation personnel	A.4	Spatial requirement
		Facility rating topics			
			Storeroom	A.4	Spatial requirement
			Space for building operation personnel	A.4	Spatial requirement
			Operation instructions for services and equipment		NA
	B.2.4.	Ease of maintenance			
		Facility management requirement topics			
			Required level of maintenance		NA
			Storage and workshop	A.4	Spatial requirement
			Access to contractors and parts		NA
			Data for inventory and maintenance program		NA
			Ease of maintenance and repairs of surfaces and materials		NA
		Facility rating topics			
			Storeroom for maintenance	A.4	Spatial requirement
			Maintenance workshop	A.4	Spatial requirement
			Maintenance contractors		NA
			Availability of replacement parts		NA
			Data for maintenance		NA
			Painting and repairs		NA

B.2.	**Manageability**			**PREMISS**	
	B.2.5.	Ease of cleaning			
		Facility management requirement topics			
			Ease of cleaning of surfaces	A4.3	Combination of material req.
			Ease of cleaning of fittings and fixtures	A4.3	Combination of fixture and furniture requirements
			Facilities for proper waste removal and recycling	A4	Spatial requirement
		Facility rating topics			
			Types of surfaces and materials	A4.3	Combination of material req.
			Fixtures, furniture, etc.	A4.3	Combination of fixture and furniture requirements
			Condition		NA
			Accessibility		NA
			Waste handling	A4	Spatial requirement
			Recycling	A4	Spatial requirement
	B.2.6.	Janitorial services			
		Facility management requirement topics			
			Level of janitor facilities	A4	Spatial requirement
			Spaces for janitor facilities	A4	Spatial requirement
			Amenities for janitorial contractors and staff	A4	Spatial requirement
		Facility rating topics			
			Supplies store	A4	Spatial requirement
			Closets on each floor	A4	Spatial requirement
			Parking and facilities		Combination of requirements
	B.2.7.	Energy consumption			
		Facility management requirement topics			
			Requirement for heating and cooling costs		Combination of cost and consumption requirements
		Facility rating topics			
			Building envelope and systems	D1	Combination of insulation and consumption requirements
	B.2.8.	Energy management and controls			
		Facility management requirement topics			
			Level of energy management and controls	D1	Combination of insulation and consumption requirements
		Facility rating topics			
			Energy system components	D1	Combination of insulation and consumption requirements

B.3.	**Management of Operations and Maintenance**			**PREMISS**	
	B.3.1.	Strategy and program for operations and maintenance			
		Facility management requirement topics			
			Level of maintenance and operation		NA
			Tolerance for occupant loss of productivity		NA
			Availability of support services		NA
		Facility rating topics			
			Strategy and program		NA
			Adequacy of budget		NA
			Human resources		NA
			Availability of replacement parts		NA
			Maintenance contractors		NA
	B.3.2.	Competence of in-house staff			
		Facility management requirement topics			
			Required level of training and skills		NA
		Facility rating topics			
			Training		NA
			Cross-trade qualifications		NA
			Electrical systems		NA
			Electronic systems and controls		NA
			HVAC equipment		NA
			Piping systems and repair		NA
			Minor carpentry		NA
	B.3.3.	Occupant satisfaction			
		Facility management requirement topics			
			Level of satisfaction with O&M operations		NA
			Management support of O&M operations		NA
			Outsourcing for O&M operations		NA
		Facility rating topics			
			Actions to achieve confidence of occupant staff		NA
			Actions to achieve confidence of senior management		NA
			Response to surveys		NA
			Outsourcing		NA
	B.3.4.	Information on unit costs and consumption			
		Facility management requirement topics			
			O&M staff understanding of practices and costs		NA
			Analysis and correction		NA
			Cooperation of building occupants		NA
		Facility rating topics			
			Database on O&M operations		NA
			Comparison with recognized ext. standards and practices		NA
			Building operational parameters and their associated costs		NA
			Use of information for effective O&M operations		NA

B.4.	Cleanliness				PREMISS
	B.4.1.	Exterior and public areas			
		Facility management requirement topics			
			Level of cleanliness for building exterior and site		NA
		Facility rating topics			
			Site		NA
			Building		NA
			Interior public spaces		NA
			Fittings, fixture and furniture		NA
	B.4.2.	Office areas (interior)			
		Facility management requirement topics			
			Level of cleanliness of the building interior		NA
		Facility rating topics			
			Building surfaces		NA
			Fittings, fixture and furniture		NA
	B.4.3.	Toilets and washrooms			
		Facility management requirement topics			
			Maintained condition of toilets and washrooms		NA
		Facility rating topics			
			Toilets and washrooms		NA
			Other amenities		NA
	B.4.4.	Special cleaning			
		Facility management requirement topics			
			Level of cleanliness in special facilities		NA
		Facility rating topics			
			Food facilities		NA
			Computer center		NA
			Secure area		NA
	B.4.5.	Waste disposal for building			
		Facility management requirement topics			
			Location for waste containers		NA
			Requirements for waste handling		NA
			Recycling program		NA
		Facility rating topics			
			Office waste		NA
			Kitchen waste		NA
			Garbage compactor		NA
			Recycling program		NA

Appendix C: Some Implementation Issues Related to the IFC Specifications

The following issues are not crucial for my research; the practical implementation of *Requirements Management* software can be done by several methods. However, the issues which came up in the rapid prototyping phase are documented in this appendix as a guideline for future implementation.

C1 Automated Generation of Space Objects from the Space Program

Linking the *Requirements Objects* with the *Design Objects* can be an extensive task, depending on the size of the *Models*. If the number of objects is high, the likelihood of errors in such a task is high (Section 6.3.2). On most levels of detail the number of objects is limited; one project usually includes only one site and also the number of buildings, stories and systems is relatively small, and the recognition of the objects is easy to automate. However, the *Spaces* are an exception; their number can be very high. Thus, creating links between *Spaces* and their *Requirements* can be a problem. The possibility of generating the *Space* objects automatically from the *Requirements Model* would solve this problem. Technically the task is not difficult; it can be based on the required area in the *Requirements Model* and with some parameters defining the generated shape and *Location* of the *Spaces*.

At least two such applications already exist; both use an MS-Excel-based *Space Program*. I implemented the first application, KIVI, in 1992–1994, and based it on the extended data possibilities of the AutoCAD blocks and polyline objects. The first project where the application was used was the ICL Headquarters (Sections 1.2.1 and 7.1.1). The second application, Space Layout Editor, was implemented by Jiri Hietanen (Section 7.3.2). He based it on MS Visio and IFC data. Both applications generate initial *Space* objects into the design software where they can be edited by the designer.

This issue, linkage between different *Models*, relates closely to the identification problems discussed in Section 6.2.3.3; how to identify the objects and maintain their links? Although the automatically created links could be based on the use of Globally Unique Identifiers (GUID), my *Requirements Model Specification* does not use GUIDs because of the identified problems (Section 6.2.3.3). Section 6.3.2 describes my solution for the link.

C2 Model Server Technology

As described in Section 3.4, the main prerequisites for the rapid prototyping were (1) *Requirements Objects* which can be linked to the (2) *Space* objects, and (3) recognition of the *Bounding Elements* related to the *Space* objects. We can link the *Requirement Objects* and objects in the *Design, Production, and Maintenance Models* using several methods. Although the full implementation was not in the scope of my research, this Section gives a brief overview of the latest IFC implementations to explain the technical options for implementation.

IFC file exchange is now supported by many commercial software vendors (Section 7.3.4). However, IFC-based file exchange is an insufficient solution for real projects [*Kam and Fischer, 2002* [122]]. The key problems are:

- The different information content in different software -> It is impossible to maintain all the data when transferring the *Building Product Model* between different software applications, and
- The lack of partial *Model* exchange -> This causes two main problems:
    - The *Building Product Models* are large, which makes the file exchange of the whole *Model* time-consuming. However, usually only a small part of the *Model* has changed and transferring the whole *Model* would not be needed, if partial exchange was available.
    - Versioning and controlling user rights are practically impossible.

Also, the complexity of the *IFC Specifications* is a bottleneck for implementation, and easier access to the *Model* data using simple queries would improve the usability of the *IFC Specifications*. Thus, several projects have been developing IFC *Model Servers* since 2001 [*IMSvr 2002* [123]*, WebSTEP 2002* [124]*, and EPM*

2003 [125]. All *Model Servers* provide partial *Model* exchange and simple query access to the *Model* using standard technologies such as XML (Extensible Markup Language), SOAP (Simple Object Access Protocol), and STEP (STandard for the Exchange of Product model data) [*Adachi, 2002* [126], *Hemiö, 2002* [127]]. The use of standard XML can solve some of the problems addressed by the Behrman report [*Behrman, 2002* [128]]

However, from the implementation viewpoint, the different application interfaces to different *Model Servers* are a problem, because they either limit the use to one *Model Server* or require implementation of several application interfaces for each domain (Figure 92). A standardized application interface for each domain can solve these problems. The SABLE project is currently developing such interfaces based on SOAP [*SABLE 2002* [129], Figure 92 and Figure 93]. Each domain-specific API handles the information exchange needed by the client applications for each domain, which logically corresponds with the BLIS views (Section 3.5).

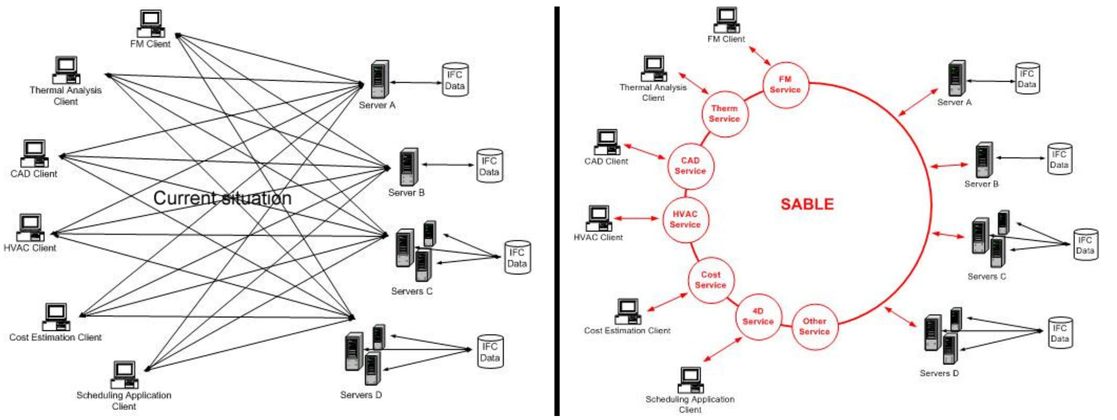

Figure 92: SABLE: advantage of the standardized interface approach *[© BLIS & SABLE]*

The best technical solution to implement the interface between the *Requirements Model* and the *Building Product Model* would be to use a standardized API, such as the SABLE interface. A standardized API would make the implementation easier and provide connections to several software products, including other design software if further research projects proposed in Section 8.3 or commercial software development use the same structures. The standardization of the software interfaces as well as the standardization of data structures is crucial for the development and use of interoperable software. However, as described in Section 3.4, this is not a crucial issue for my research.

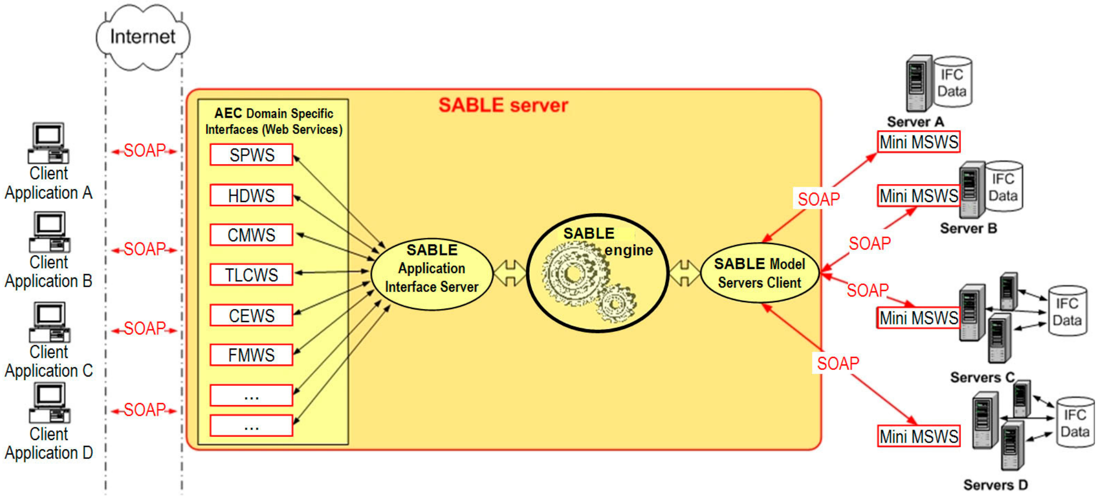

Figure 93: SABLE architecture *[© BLIS & SABLE]*

The proposed *Requirements Model Specification* can be implemented in a *Model Server* environment in two different ways (Figure 93):

- Option #1: The *Requirements Model* is stored in a separate database which has its own user-interface (UI), and the connection from the *Requirements Model* to the *Design Model* is through a domain-specific API (Figure 92). In this option, the *Requirements Management* software is a "stand-alone" application and needs the connection to the *Model Server* only when using the links between *Design Model* and *Requirements Model*. However, this means that the *Requirements Management* UI in the design software must be able to connect to the *Requirements Database* when the user wants to see the *Requirements* related to his design tasks.
- Option #2: The *Requirements Model* is stored in a *Model Server* database. In this option the *Requirements Management* software's UI communicates with the *Requirements Database* through the domain-specific API in the same way as design software's *Requirements Management* UI. The benefit of this approach is that all the shared project information is stored on the same *Model Server* and accessible using the same methods.

Option #2 is significantly better in meeting the requirements for a good solution to the *Requirements Management* problems (Section 6.1.1) than option #1, where the connection between *Requirements and Design Models* is less integrated. However, even option #1 would be a clear improvement to the current situation, where the link between the *Requirements* and design solutions is totally missing. Thus, option #1 is a useful solution if the integrated *Model Server* platform for the *Requirements Model* needed for option #2 is not available.

Appendix D: Expert Evaluations

The following five statements are responses to my request to check my *Requirements Model Specification* and asses specifically the implementability of the *Specification* (Section 7.3.5). However, many of the statements evaluate my *Specification* also from other viewpoints. The group includes the following people (in the chronological order of their statements): Jiri Hietanen, Patrick Houbaux, Kari Karstila, Robin Drogemuller, and Richard See.

PREMISS -

REQUIREMENTS MANAGEMENT INTERFACE

TO BUILDING PRODUCT MODELS

Statement about the implementability of the proposed Requirements Model Specification.

There are different ways of assessing if a model is implementable or not. First the model has to be such that it is technically possible to create software which is using it, secondly there are principles of good software design to be followed and thirdly there is the question of practicality and commercial feasibility. In the following I am providing statements about each of these aspects.

The proposed Requirements Model Specification can be implemented from the technical viewpoint. I have not noticed any conceptual mistakes or misunderstandings, which would make it impossible to use the model in implementations. The schema is valid and the solution for links between different data sets (model instances) is correct. The model is designed as an extension to the IFC model, and it makes correct use of existing IFC concepts whenever possible. However, there does not exist any exact and agreed way how the IFC model must be extended. From the viewpoint of the EXPRESS language the extension is valid, but there may exist published or unpublished agreements for extending the IFC specification that might be violated. For this reason any proposed extension to the IFC model goes through a detailed integration process, which would also be the proper process for this extension. It is my understanding that the proposed extension is advanced enough to enter the integration process, and going through this process would most likely lead to changes in some details of the model. However, in my opinion none of the principles of the model would have to be changed.

From the software design principles point of view the most important aspect of the proposed Requirements Model Specification is modularity. By strictly separating the requirements data from the design data on the object level and by allowing the requirements and design to be managed in separate data sets, it provides the possibility for modular software. This architecture makes it possible to separate the requirements management into a stand-alone application, or into an add-on of a design application. It is also possible to create and to verify requirements in separate applications, because the model can be used as an internal data model as well as a data exchange model. The biggest challenge for software design would be maintaining the links between the different requirement and design data sets, which is possible but would require special attention.

To be used in commercial software there would have to be agreements how the model is used in different use cases. This is the case for all IFC implementations (view definitions) and this requirement is correctly noted in the thesis. The modular structure will greatly increase the probability of commercial adoption, because there is no need for tight integration with existing design applications, although such integration would in many cases be beneficial. It is possible for innovative software developers to create new types of applications which make use of the proposed model. Another factor in favor of commercial implementations is the availability of reusable software components and model servers supporting IFCs. If the model is accepted as an extension to the IFC model it will automatically be supported by these components and servers. In theory one limiting factor for implementations may be the initial selection of requirements supported by the model, but I don't have any expertise in this area. Any missing requirements could be quite easily added later using the framework defined by the model.

As a summary; it is possible to implement the Requirements Model Specification. The model is technically feasible, it supports good software design principles and there are factors, which make it likely that it will be used by commercial software. However, official integration to the IFC model would probably lead to changes in some details, commercial implementations depend on the existence of commonly agreed view definitions and the scope of supported requirements would possibly have to be extended at some point.

Jiri Hietanen

Tampere, Finland, January 12th, 1.2005

Jiri Hietanen, Managing Director at qPartners Inc. and Research Scientist at Tampere University of Technology. Mr. Hietanen was the former Assistant Technical Director of IAI 1998–1999, and he is a co-founder of BLIS and the Technical Coordinator of BLIS since 1999. Mr. Hietanen has also worked as a consultant on IFC implementation for several companies and has defined implementation definitions and practical guidelines for the use of IFCs in building projects.

Statement on Arto Kiviniemi's thesis

The requirement object model specifications designed by Arto Kiviniemi in his thesis, scopes a domain that is currently out of the scope of most of the building information models available for the building industry. I do believe this model captures most of the need for the domain it is dealing with but like any other 'first of the kind' it will need some rework (mostly concerning the constructed pattern) for being integrated, for instance, within the IAI IFC model or harmonized with existing requirement object models in other industry like the STEP AP233 or the PLCS model.

In any case, Arto's work will certainly facilitate the creation of new software in this domain. Projects like SABLE will certainly benefit from such a work since this model can, to some extent, be used as the only input for the design specifications of a high level API in the field of requirement management.

I personally consider Arto's model as the only existing formulized requirement for an object model dealing with requirement management.

Patrick Houbaux
Helsinki, January 30th, 2005

Patrick Houbaux, Senior Consultant for Product Data Management at Eurostep Group since 2003. Mr. Houbaux was the former project manager for CSTB's STEP SDAI platform (QualiSTEP) from 1999 to 2001 in France. He joined the BLIS project in 1999. From 2001 to 2002, Mr. Houbaux worked at Solibri as the R&D advisor for the Solibri Model Checker [*Solibri* [130]]. Mr. Houbaux has been an active implementer of different IFC releases and has been involved in different groups within the IAI including the French Speaking Chapter, and the ISG, ITM and XML steering groups. He is one of the authors of the specification of the BLIS-XML methodology and facilitated the first IFC 2.0 certification workshop in 2001. Mr. Houbaux is currently the project manager of the SABLE project [*SABLE 2002* [131]]. Mr. Houbaux is one of Eurostep's leading and most experienced consultants in product model implementation, design of software infrastructures and components of traditional and web based data exchange using STEP Part 21, SOAP, XML and web services, for concurrent engineering in the building industry.

Statement on Arto Kiviniemi's dissertation for PhD

Reference

Arto Kiviniemi. Premiss - requirements management interface to building product models. Dissertation for PhD, Version 12th of February 2005.

Importance of the work

The work of Arto Kiviniemi addresses an important yet neglected area of requirements managements in AEC/FM. Hopefully the work leads also to further steps towards practical implementations. In AEC/FM the area of requirements management seem to be behind of systematics and practises used in other industries like aerospace, defence and software industries.

Contribution of the work

Kiviniemi's contribution in his work is especially in capturing the information requirements for the spatial requirements definition and management. The results come both from analysis of construction projects and his experience in architectural practice. His requirements model specification forms a good baseline for possible further work on evaluating how the current IFC specification addresses those requirements, and in identifying the needs for IFC extensions in better addressing the area of requirements management in AEC/FM.

Helsinki 17th of February 2005 *Kari Karstila*

Kari Karstila, MSc, Structural Engineering, has about 20 years of experience in working in the area of construction information technology, product and process modeling, and standards development. He worked at the Civil Engineering department of the Helsinki University of Technology (HUT) being involved in the basic CAD courses and department's IT systems management. From HUT he moved to VTT (Technical Research Centre of Finland) to work as a researcher in the Construction IT Group. During his tenure at VTT he participated in many national and European R&D projects for construction IT and product and process modeling. In 1996 he joined Eurostep, a consulting and software company for product data and life cycle management. While working at Eurostep on R&D and industry projects, he has among other things participated in the international standardization efforts of PLCS [Product Life Cycle Support, *PLCS 2005* [132]] and especially IFC. Since 1998 he has been a member of the IAI Modeling Support Group. Mr. Karstila's areas of expertise include construction IT in general, product and process modeling, enterprise/information architectures, software development, and standards like ISO STEP, PLCS and IAI/IFC.

PREMISS

Requirements Management Interface to Building Product Models

I have examined the PREMISS model from a number of perspectives based on my previous experience – as an architectural brief writer, as an architect using briefs prepared by others, as a facility manager assessing how closely a building design matches a brief and as a software implementer who has a detailed understanding of the IFC model and object-oriented CAD systems. I would assess such work against the following criteria – adequacy for storing the requisite information, ease of manipulation for adding, reading and manipulating the information and suitability for implementation in computer software. After a detailed analysis of the PREMISS model I consider that it is appropriate for storing the information that is within scope and has addressed the issues of interfacing with information which is currently out of scope. Both of these are necessary within any real world application of the results.

The PREMISS model identifies shortcomings with the IFC model, with which my software team within CSIRO are in agreement. We had identified some of these independently, but we had not considered others that have been addressed within PREMISS. The model follows its own recommendations for addressing the IFC issues for the entities defined within its scope, while maintaining compatibility with the currently defined IFC model. This is necessary due to the range of software that already supports the IFC interface. The recommendations within the PREMISS model improve the accessibility and ease of modification of the entities within scope. Consequently, PREMISS meets the second criteria.

My team have implemented 7 pieces of software using the IFC model and defined mappings between the IFCs and the internal models. Based on this experience I am confident that the PREMISS model can be implemented. This has lead to discussions with the CSIRO Corporate Property group, who are responsible for housing 6500 CSIRO staff, regarding the use of the PREMISS model in their requirements capture, together with other software, as part of their facilities management process. This will be a useful validation of the PREMISS work as some information from CSIRO Corporate Property was used in defining the PREMISS requirements.

Since the PREMISS work meets all of the above requirements, I would judge the PREMISS work as a success.

Robin Drogemuller
Melbourne, Australia, 19th February, 2005

Dr. **Robin Drogemuller**, leads a research team of 20 people within CSIRO (Australian Government research organization) working on the use of ICT within the AEC-FM industry, including interoperability issues. He worked as an architect and construction manager in both the private and public sectors before becoming an academic teaching CAD and construction management. Dr. Drogemuller has been a member of the IAI since 1996 when he was invited to join the Technical Advisory Committee. He is a foundation member of the IAI Australasia Chapter and has served as Technical Coordinator, Treasurer and Chairman of the Australasia Chapter. He has represented the Australasia Chapter at international meetings since 1998 on both the International Council and International Technical Management committee. Dr. Drogemuller was also a member of the Specification Task Force for IFC versions 1.5.1 and 2.0.

Reviewer Statement on Arto Kiviniemi's Dissertation for PhD

Richard See – 19-Feb-05

It was my pleasure to review Arto's dissertation as I believe the focus of PREMISS to be important. As a licensed architect here in the US, I know the importance of accurately capturing client requirements and of fully understanding them throughout the building design process. Unfortunately, I am also fully aware that this is an area that has not yet been well addressed in computer software tools and applications. This is unfortunate for the building industry. Having led a large number of software design and implementation projects in the past 20 years, I know this neglect to be unnecessary as capture of such client requirements and making them available through the design process is quite achievable.

In this project, Arto has done a very credible job of synthesizing and prototyping a model schema for such requirements capture and representation. I applaud his pragmatic approach to this; learning from previous less ambitious attempts, designing it as an extension to the IFC model (the most logical context for implementation), and focusing on what is most important, based on real world projects and his own industry experience.

What I find most notable and interesting in this work is that Arto did not stop at requirements capture and modeling, but has proposed a viable scheme for relating these requirements to elements/assemblies in design models. As he notes, this will enable design performance assessment, relative to client requirements, a possible extension to this work. In the past 20 years, I have worked with many of the industry visionaries in the area of building modeling software and projects, and have followed most projects in this field. I find PREMISS to be a notable contribution that is important, ground breaking, and achievable. I look forward to seeing it implemented in a software product that is used in the building industry.

Richard See
Lead Program Manager – Microsoft Real Time Collaboration
Chairman – BLIS Project

Richard See holds a Master of Architecture degree from the University of Washington in Seattle. He is a licensed Architect in the State of Washington, and practiced architecture with some of the leading design firms in the Pacific Northwest region of the U.S. Mr. See has also contributed to the development of 3 CAD systems and led design and/or development for a number of computer graphics applications at industry-leading companies including Autodesk, Visio, and Microsoft.

In the 9 years before joining Microsoft, Mr. See led several teams developing technology and methodologies for enabling interoperability between applications in the design, construction, and real estate industries. In the role of International Technical Director for the International Alliance for Interoperability (IAI), Mr. See led development of 3 releases of the Industry Foundation Classes, a software object model representation for building industry projects that has emerged as the industry standard for software interoperability in the building industry and has since been endorsed as a formal ISO standard.

Mr. See came to Microsoft with the acquisition of Visio Corporation, where he was lead program manager for advanced technology development. In that role, his team created and shipped multiple releases of the Visio Viewer, Visio IFilter, and the Visio IFC model exchange solution. They also built the new foreign and legacy graphic data translation system that first shipped in Visio 2003.

After the release of Visio 2003, Mr. See co-founded a startup to develop a new product in the Microsoft Greenhouse. The product includes both hardware and software, is cited internal to Microsoft as a truly innovative addition to the Real Time Collaboration products by Microsoft. The product will launch in 2006.

List of References

[*acadGraph*] acadGraph is an German software vendor and Alberti was one of their products at least in 2000–2002. However, it seems that it is not on the market anymore; website, last accessed on February 18th, 2004: http://www.acadgraph.de/start.html

[*Adachi, 2002*] Adachi, Yoshinobu: Introduction of IFC Model Server, 2002, website, last accessed on February 18th, 2005: http://cic.vtt.fi/vera/Seminaarit/2002.04.24_IAI_Summit/Adachi.pdf

[*Adachi, 2005*] Adachi, Yoshinobu: Solution for the model server object reference linkage, sent as a part of the GUID problem discussions in the BLIS group, January 2005

[*ADT*] Architectural Desktop by Autodesk, an object-based design software, web site last accessed on February 18th, 2005: http://www.autodesk.com/

[*aecXML 2005*] IAI North American Chapter, aecXML website, last accessed on February 18th, 2005: http://www.iai-na.org/aecxml/staging_areas.php

[*Alexander et al., 1977*] Alexander, Christopher; Ishikawa, Sara; and Silverstein, Murray: A Pattern Language, Oxford University Press 1977. Library of Congress Catalogue Card Number 74-22874

[*ArchiCAD*] ArchiCAD by Graphisoft, an object-based design software, web site last accessed on February 18th, 2005: http://www.graphisoft.com/

[*Behrman, 2002*] Behrman, William: Best Practices for the Development and Use of XML Data Interchange Standards. CIFE Technical report TR131, Stanford University 2002, available also at http://cife.stanford.edu/online.publications/TR131.pdf

[*Best and De Valence, 1999*] Best, Rick and De Valence, Gerard: Building in Value, Arnold 1999. ISBN 0 340 74160 0

[*BLIS 2000*] BLIS software demonstrations: IAI International Council Summit, London, October 26, 2000; Vera Seminar, Helsinki, November 13, 2000, IFC Workshop, Copenhagen, November 14, 2000, website, last accessed on February 18th, 2005: http://www.blis-project.org/demos/index.htm

[*BLIS 2002*] Official BLIS/IAI certification workshop documents: GUID Tracking Reports: GUID_IFCR2_Buildings_1_JP5_ADT.htm, GUID_IFCR2_Buildings_1_ JP5_pbld.htm, and GUID_IFCR2_Buildings_2_JP5_nec.htm. Tokyo, October 2002

[*BLIS 2004*] BLIS SW website, last accessed on February 18th, 2005:

- Main page http://www.blis-project.org/BLIS_Product_Public.html
- View definitions http://www.blis-project.org/views/index.htm
- Concepts http://www.blis-project.org/IFCR2_Concept_block_approach_000524_jh.pdf

[*Clark Center 2004*] Clark Center Information and Resources, last accessed on February 18th, 2005: http://mednews.stanford.edu/clark-center-index.html

[*Corenet 2004*] Construction and Real Estate Network, Integrated Plan Checking System, last accessed on February 18th, 2005: http://www.corenet.gov.sg/

[*Drogemuller, 2004*] Drogemuller, Robin: Comments to the preliminary *Requirements Model Specification*, version December 11th, 2004.

[*Eastham, 2002*] Eastham, G. M: ECI Fast Track Projects Study "The Effective Management of Fast Track Projects," Executive summary, pp. 1–6, European Construction Institute, July 2002

[*EcoProp*] EcoProp is a *Requirements Management* software developed at VTT Building and Transport (Technical Research Centre of Finland). EcoProp version 4.1.0 is used as the basis of PREMISS *Requirements Hierarchy*. Short description of EcoProp system is available at http://www.vtt.fi/rte/esimerkkeja/research_results_2004.pdf, page 4

[*Ekholm and Lehtonen, 2002*] Ekholm, Anders and Lehtonen, Riikka: Creative construction Briefing with Object-Oriented Technology and IFC, eSM@rt 2002 Conference Proceedings Part A, pages 229–234. University of Salford 2002. ISBN 0902896415

[*EPM 2003*] EDM (Express Data Manager), EPM Technology, last accessed on February 18th, 2005: http://www.epmtech.jotne.com/products/index.html

[*Evbuomwan, 1994*] Evbuomwan Nosa F. O.: Design Function Deployment: A Concurrent Engineering Design System, PhD Thesis, City University of London 1994. Indirect reference from Kamara et al., 2003: Page 42.

[*Fällman, 2003*] Fällman, Daniel. "Design-oriented Human-Computer Interaction," Proceedings of CHI2003, Conference on Human Factors in Computing Systems, Fort Lauderdale, Florida, published in CHI Letters, Vol. 5, Issue No. 1, pages 225–232. ACM Press, 2003.

[*Froese, 2002*] Froese, Thomas: Current Status and Future Trends of Model Based Interoperability, eSM@rt 2002 Conference Proceedings Part A, pages 199–208. University of Salford 2002. ISBN 0902896415

[*Garcia et al., 1993*] Garcia, Ana Cristina; Howard, H. Craig; and Stefik, Mark J.: Active Design Documents: A New Approach for Supporting Documentation in Preliminary Routine Design, CIFE Technical report TR 82, Stanford University 1993

[*Garcia et al., 2003*] Garcia, Ana Cristina; Kunz, John, Ekström, Martin; and Kiviniemi, Arto: Building a Project Ontology with Extreme Collaboration and Virtual Design and Construction. CIFE Technical report TR152, Stanford University 2003, available also at http://cife.stanford.edu/online.publications/TR152.pdf

[*Gehry, 2003*] Gehry, Frank O.: Quote from Gehry Technologies website, last accessed on February 18nd, 2005: http://www.gehrytechnologies.com/company-press-10-23-2003.html

[*GUID 2005*] Globally Unique ID, Open Group website, last accessed on February 18th, 2005: http://www.opengroup.org/dce/info/draft-leach-uuids-guids-01.txt

[*Haymaker et al., 2003*] Haymaker, John; Suter, Ben; Kunz, John; and Fischer, Martin: PERSPECTORS: Automating the Construction and Coordination of Multidisciplinary 3D Design Representations. CIFE Technical Report TR145, Stanford University 2003, available also at http://cife.stanford.edu/online.publications/TR145.pdf

[*Hemiö, 2002*] Hemiö, Tero: A Project Model Server Approach, WebSTEP, last accessed on February 18th, 2005: http://cic.vtt.fi/vera/Seminaarit/2002.04.24_IAI_Summit/WebSTEP.pdf

[*Hietanen, 2000*] Hietanen, Jiri: Space Layout Editor Demo, last accessed on February 18th, 2005: http://www.blis-project.org/demos/01_BLIS_Helsinki_SLE.zip

[*Hietanen, 2003*] Hietanen, Jiri: BLIS view definitions, website, last accessed on February 18th, 2005: http://www.blis-project.org/views/index.htm

[*IAI ISG 2004*] IAI ISG (Implementation Support Group) website, last accessed on February 18th, 2005: http://www.iai.fhm.edu/ImplementationOverview.htm

[*IAI NA 2003 a–b*] IAI North American Chapter website, last accessed on February 18th, 2005:

a) IAI NA News website: http://www.iai-na.org/news/092204.php
b) IAI NA Projects website: http://www.iai-na.org/technical/projects.php

[*Ibrahim and Paulson, 2004*] Ibrahim, Rahinah and Paulson, Boyd C. Jr.: Discontinuity in Organizations: How Environmental Characteristics Contribute to The Project's Knowledge Loss Phenomenon. CRGP Working Paper #12, Stanford University 2004. Available at http://crgp.stanford.edu/publications/working_papers/RahinahPaulson1.pdf

[*ICF 1993*] Davies, Gerald; Thatcher, Carrol; and Blair, Lynne: Serviceability Tools 1993, Volume 1, Methods for Setting Occupant Requirements and Rating Buildings and Volume 2, Scales for Setting Occupant Requirements and Rating Buildings International Centre for Facilities 1993. ISBN 1-896021-00-X and ISBN 1-896021-01-8

[*ICF 2000*] International Centre for Facilities: ASTM Standards on Whole Building Functionality and Serviceability. Second Edition. American Society for Testing and Materials 2000. ISBN 0-8031-2734-0

[*IFC 2004 Add1*] IFC 2x2 Addendum 1 web site, last accessed on February 18th, 2005: http://www.iai-international.org/iai_international/Technical_Documents/R2x2_add1/index.html

[*IFC 2004a - k*] IAI International website, last accessed on February 18th, 2005:

a) IfcConstraint: http://www.iai-international.org/iai_international/Technical_ Documents/R2x2_add1/ifcconstraintresource/lexical/IfcConstraint.html
b) IfcControl: http://www.iai-international.org/iai_international/Technical_ Documents/R2x2_add1/ifckernel/lexical/IfcControl.html
c) IfcSpaceProgram: http://www.iai-interntional.org/iai_international/Technical_ Documents/R2x2_add1/ifcarchitecturedomain/lexical/IfcSpaceProgram.html
d) IfcSpace: http://www.iai-international.org/iai_international/Technical_ Documents/R2x2_add1/ifcproductextension/lexical/IfcSpace.html
e) IfcOwnerHistory: http://www.iai-international.org/iai_international/Technical_ Documents/R2x2_add1/ifcproductextension/lexical/IfcSystem.html
f) IfcGloballyUniqueID: http://www.iai-international.org/iai_international/Technical_ Documents/R2x2_add1/ifcutilityresource/lexical/IfcGloballyUniqueId.html
g) IfcIdentifier: http://www.iai-international.org/iai_international/Technical_ Documents/R2x2_add1/ifcmeasureresource/lexical/IfcIdentifier.html
h) IfcApprovalStatus: http://www.iai-international.org/iai_international/Technical_ Documents/R2x2_add1/ifcapprovalresource/lexical/IfcApproval.html
i) IfcDocumentReference: http://www.iai-international.org/iai_international/Technical_Documents/R2x2_add1/ifcexternalreferenceresource/lexical/IfcDocumentReference.html
j) IfcRelSpaceBoundary: http://www.iai-international.org/iai_international/Technical_ Documents/R2x2_add1/ifcproductextension/lexical/IfcRelSpaceBoundary.html
k) IfcSystem: http://www.iai-international.org/iai_international/Technical_ Documents/R2x2_add1/ifcproductextension/lexical/IfcSystem.html

[*IMSvr 2002*] IMSvr IFC Model Server project, Yoshinobu Adachi, Secom/VTT 2001–2002, web site, last accessed on February 18th, 2005:

- http://cic.vtt.fi/projects/ifcsvr/index.html
- http://cic.vtt.fi/vera/Projects/e_ifc_model_server.htm

[*ISO 2004*] International Organization for Standardization, Publicly Available Standards, web site last accessed on February 18nd, 2005: http://www.iso.org/iso/en/ CatalogueDetailPage.CatalogueDetail?CSNUMBER=38056&scopelist=PROGRAMME

[*Kagioglou et al., 1998*] Kagioglou, Michael; Cooper, Rachel; Aouad, Ghassan; Hinks, John; Sexton, Martin; and Sheath, Darryl: A Generic Guide to the Design and Construction Process Protocol. University of Salford 1998. ISBN 0-902896-17-2

[*Kam and Fischer, 2002*] Kam, Calvin and Fischer, Martin: PM4D Final Report. CIFE Technical Report 143, Stanford University 2002. Available also at http://www.stanford.edu/group/4D/download/c1.html

[*Kam and Fischer, 2004*] Kam, Calvin and Fischer, Martin: Capitalizing on early project decision-making opportunities to improve facility design, construction, and life-cycle performance POP, PM4D, and decision dashboard approaches. Journal of Automation in Construction, Volume 13, Issue 1, pages 53–65, Elsevier B.V. 2004

[*Kam, 2005*] Kam, Calvin: "Dynamic Decision Breakdown Structure – Ontology, Methodology, and Framework for Information Management in Support of AEC Decision-Enabling Tasks," currently unpublished Ph.D. research, Civil and Environmental Engineering, Stanford University, 2005

[*Kamara et al., 2003*] Kamara, John M.; Anumba, Chimay J.; and Evbuomwan, Nosa F. O.: Capturing client requirements in construction projects, Thomas Telford Ltd 2003. ISBN 0 7277 3103 3

[*Karatmaa, 2000*] Karatmaa, Arno: "Coordination in Distributed Product Development Teams," Master's thesis in Helsinki University of Technology, Department of Computer Science, Laboratory of Information Processing Science, 2000, available also at http://www.soberit.hut.fi/gecos/deliverables/thesis_arno.pdf

[*Kiviniemi et al., 2004*] Kiviniemi, Arto; Fischer, Martin; Bazjanac, Vladimir; and Paulson, Boyd C. Jr.: PREMISS – Requirements Management Interface to Building Product Models: Problem Definition and Research Issues. CIFE Working Paper 92, Stanford University 2004. Available also at http://cife.stanford.edu/online.publications/WP092.pdf

[*LBNL 1995–2003*] Lawrence Berkeley National Laboratory, all web sites last accessed on February 18th, 2005:

- Simulation Research Group 2003, http://simulationresearch.lbl.gov/
- [*LBNL BLISS 1997*] Building Life-Cycle Information Support System (BLISS), http://eetd.lbl.gov/BTP/CBS/BPA/bliss.html
- [*LBNL DIT 2003*] Design Intent Tool DIT, http://ateam.lbl.gov/DesignIntent/home.html

[*MicroStation*] MicroStation by Bentley, an object-based design software, web site, last accessed on January 5th, 2005: http://www.bentley.com/

[*MS Visio*] Visio by Microsoft, an object-based design software, web site, last accessed on January 5th, 2005: http://office.microsoft.com/

[*Oinas-Kukkonen, 1997*] Oinas-Kukkonen, Harri: Improving the Functionality of Software Design Environments by Using Hypertext, Ph.D. thesis at University of Oulu, 1997. ISBN 951-42-4602-0

[*PLCS 2005*] The Product Life Cycle Support (PLCS) initiative is a joint industry and government project to accelerate the development of a new international standard for exchange of assured product and support information. Web site, last accessed on March 2nd, 2005: http://www.plcsinc.org/

[*Prasad, 1996*] Prasad, B.: Concurrent Function Deployment — An Emerging Alternative to QFD: Conceptual Framework. Advances in Concurrent Engineering: Proceedings of CE 96 Conference, USA, pages 105–112. Indirect reference from Kamara et al., 2003, page 42.

[*Programs 2003*] Analyzed building programs:

- ICL Headquarters, Helsinki: Space program February 1996, total gross area 27,350 m^2
- Aurora II, Joensuu University: User Requirements Memorandum, August 25th, 2003; Space program January 26th, 2004; EcoProp Report, March 10th, 2004; total gross area 7,120 m^2
- CSLI-Media X / EPGY Annex Building, Stanford University: Programming Report April 10th, 2003; total gross area 1022 m^2
- Kavli Institute, Stanford University: Schematic Design Narrative July 25th, 2003; total gross area 2,330 m^2
- Lucas Center Expansion, Stanford University 2003: Several versions of Program Area Summary, February 1st, 2002, April 17th, 2002, September 11th, 2002, October 18th, 2002, and November 26th, 2002; MPE Utility Planning and System Description V1, March 5th,2002; Architects Requirements Database, November 6th, 2002; Design Meeting Memorandum, January 31st, 2003; Several versions of sketches and drawings; total gross area 1,960 m^2

[*ProIT 2004*] ProIT project's Building Product Model vocabulary, web site, last accessed on February 18th, 2005: http://www.vtt.fi/rte/cmp/projects/proit/julkiset_tulokset/proit_sanasto_v10.pdf

[*Revit*] Revit by Autodesk, an object-based design software, web site last accessed on February 18th, 2005: http://www.autodesk.com/

[*Rittel and Kunz, 1970*] Rittel, Horst and Kunz, Werner: Issues as Elements of Information Systems. Working Paper 131, University of California Berkeley, 1970. Available also at http://www-iurd.ced.berkeley.edu/pub/WP-131.pdf

[*RT 2002*] Confederation of Finnish Construction Industries RT, ProIT project web site, last accessed on February 18th, 2005: http://www.vtt.fi/rte/cmp/projects/proit_eng/indexe.htm

[*SABLE 2003*] Eurostep's "Simple Access to Building Lifecycle Exchange" project, Houbaux, Patrick and Hemiö, Tero. Web site last accessed on February 18th, 2005:

- Home page http://www.blis-project.org/~sable/
- Motivation http://www.blis-project.org/~sable/about/why.html

[*Senate 2004*] GUID report on December 22nd, 2004 from Aurora 2 project, Senate Properties. Created using Information Model Reporter by qPartners Oy

[*Solibri*] Solibri Model Checker, web site last accessed on February 18th, 2005: http://www.solibri.com/

[*SPADEX 2002*] SPADEX, Short description of IFC Project 1.3.1999–30.9.2001, Final Report. Also available at http://cic.vtt.fi/vera/Documents/SPADEX_final_report.pdf

[*Spormann, 2004*] Interview of Dr. Alfred M. Spormann, Associate Professor of Civil and Environmental Engineering and, by courtesy, of Geological and Environmental Sciences and of Biological Sciences at Stanford University, November 16th 2004

[*Stanford 2001*] The Project Delivery Process at Stanford, Process Phase and Control Summaries, Stanford University Capital Planning & Management, Volume 1, Version 1.0, September 2001. Also available at http://cpm.stanford.edu/pdp.pdf

[*Swebok 2004*] Guide to the Software Engineering Body of Knowledge, SWEBOK®, The Institute of Electrical and Electronics Engineers, Inc, 2004. Website last accessed on February 18th, 2005: http://www.swebok.org/ironman/pdf/Swebok_Ironman_June_23_%202004.pdf

[*Syed et al., 2003*] Syed M. Ahmed; Li Pui Sang; and Zeljko M. Torbica: Use of Quality Function Deployment in Civil Engineering Capital Project Planning. Journal of Construction Engineering and Management. Volume 129, pages 358–368. ASCE 2003ISSN 0733-9364

[*WebSTEP 2002*] Eurostep's "WebSTEP IFC Model Server" project, Karstila, Kari and Hemiö, Tero. Web site last accessed on February 18th, 2005:

- http://www.eurostep.com/prodserv/ems/ems.html
- http://cic.vtt.fi/vera/Projects/e_ifcnextstep.htm

[*Whelton and Ballard, 2003*] Whelton, Michael and Ballard, Glenn: Dynamic States of Project Purpose: Transitions from Customer Needs to Project Requirements – Implications for Adaptive Management, 11th Annual Conference of International Group of Lean Construction 2003. Available also at http://strobos.cee.vt.edu/IGLC11/PDF%20Files/12.pdf

Endnotes

[1] Stanford 2001: Page 8

[2] Fällman, 2003. Page 229

[3] Kagioglou et al., 1998: Section 1, page 1:11

[4] Discussions and interviews 2002-2003: Robin Drogemuller/CSIRO, Stephen Hagan/GSA, Jiri Hietanen/TUT, Reijo Kangas/Tekes, Auli Karjalainen/Senaatti, Markku Kaskimies/Pöyry, Tapio Koivu/VTT, Tuire Kujala/Engel, Jarmo Laitinen/TUT, Pekka Metsi/Pöyry, Olli Nummelin/YIT, Andrea Papanastassiou/Mid-Peninsula Housing Coalition, Vesa Pirinen/YIT, Sointu Rajakallio/Pöyry, Ilkka Romo/RT, Ben Schwegler/Disney Imagineering, Richard See/Microsoft, Mika Soini/NCC, Riitta Takanen/NCC, Juha Tammivuori/Skanska, Eija Virtasalo/Tekes, John Voller/BV Solutions Group, Inc

[5] Garcia et al., 2003: Project Ontology, figure 1, page 7

[6] Syed et al., 2002: Pages 358–368

[7] Discussions and interviews 2002–2003, see endnote [4]

[8] Discussions and interviews 2002–2003, see endnote [4]

[9] Gehry, 2003: Internet

[10] Alexander et al., 1977

[11] Rittel and Kunz, 1970

[12] Froese, 2002: Section 2.1, pages 199–200

[13] Kamara et al., 2003: Section 1.3.3, page 6

[14] Best and De Valence, 1999: Section 1.4, page 9

[15] Kamara et al., 2003: Section 7.4, page 156

[16] Eastham, 2002: Executive Summary, pages 3 and 5

[17] RT 2002

[18] Froese, 2002: Section 2.2, page 200

[19] Garcia et al., 1993

[20] Whelton and Ballard, 2003

[21] Kamara et al., 2003: Chapter 3, pages 35–60

[22] Kam, 2005

[23] Evbuomwan, 1994: From Kamara et al., 2003: Chapter 3.4.4 page 42

[24] Prasad, 1996: From Kamara et al., 2003: Section 3.4.4 page 42

[25] Syed et al., 2002: Pages 358–368

[26] Kamara et al., 2003: Figure 3.1, page 39

[27] Kamara et al., 2003, Chapters 4–6, pages 61–145

[28] Kamara et al., 2003, Chapter 5, pages 96–106

[29] Kamara et al., 2003: Chapter 5, pages 85–93

[30] Oinas-Kukkonen, 1997

[31] Karatmaa, 2000

[32] Swebok 2004

[33] Swebok 2004: Page 2-2, Figure 1

[34] ICF 1993

[35] ICF 2000

[36] EcoProp

[37] Corenet 2004: Internet

[38] LBNL 1995–2003

[39] LBNL BLISS, 1997

[40] LBNL DIT, 2003

[41] LBNL DIT, 2003

[42] ArchiCAD: Internet

[43] ADT: Internet

[44] Revit: Internet

[45] MicroStation: Internet

[46] MS Visio: Internet

[47] ISO 2004: Internet

[48] IAI NA 2003a: Internet

[49] Behrman, 2002

[50] aecXML 2005: Internet

[51] IMSvr 2002: Internet

[52] WebSTEP 2002: Internet

[53] EPM 2003: Internet

[54] acadGraph: Internet

[55] SPADEX 2002: Section 5.4, page 11

[56] IAI NA 2003b: Internet

[57] IAI NA 2003b: Internet

[58] SABLE 2002: Internet

[59] Ekholm and Lehtonen, 2002: Pages 229–234

[60] BLIS 2004: Space Layout Editor, qPartners Oy

[61] SABLE 2002: Internet

[62] IMSvr 2002: Internet

[63] WebSTEP 2002: Internet

[64] EPM 2003: Internet

[65] ArchiCAD: Internet

[66] ADT: Internet

[67] MS Visio: Internet

[68] BLIS 2004: Internet

[69] Hietanen, 2003: Internet

[70] Programs 2003

[71] Kiviniemi et al., 2004: Section 6.1, pages 41–43

[72] Kiviniemi et al., 2004: Section 6.1, pages 41–43

[73] Kam and Fischer, 2002: Section 6.2, pages 40–41

[74] Haymaker et al., 2003

[75] IFC 2004a: IfcConstraint, Internet

[76] IFC 2004b: IfcControl, Internet

[77] IFC 2004c: IfcSpaceProgram, Internet

[78] IFC 2004d: IfcSpace, Internet

[79] IFC 2004e: IfcOwnerHistory, Internet

[80] IFC 2004f: IfcGloballyUniqueID, Internet

[81] GUID 2005: Globally Unique ID, Internet

[82] BLIS 2002: BLIS/IAI certification workshop in Tokyo, 2002

[83] Senate 2004: GUID report from Aurora 2 project

[84] Adachi, 2005: Model server object reference linkage

[85] IFC 2004h: IfcApprovalStatus, Internet

[86] IFC 2004i: IfcDocumentReference, Internet

[87] IFC 2004j: IfcRelSpaceBoundary, Internet

[88] IFC 2004k: IfcSystem, Internet

[89] Drogemuller, 2004

[90] IFC 2004d: IfcSpace, Internet

[91] IFC 2004g: IfcIdentifier, Internet

[92] BLIS 2004: Internet

[93] Clark Center 2004: Internet

[94] Spormann, 2004: Interview

[95] Kam and Fischer, 2004

[96] Ibrahim and Paulson, 2004: Pages 2–3

[97] Ibrahim and Paulson, 2004: Page 3

[98] Discussions and interviews 2002–2003, see endnote [4]

[99] Hietanen, 2000: Internet

[100] BLIS 2000: Internet

[101] IAI ISG 2004: Internet

[102] BLIS 2004: Internet

[103] Kam and Fischer, 2002: Section 6.2, pages 40–41

[104] Haymaker et al., 2003

[105] IFC 2004k: IfcSystem, Internet

[106] Kam and Fischer, 2002: Section 6.2, pages 40–41

[107] Haymaker et al., 2003

[108] Haymaker et al., 2003

[109] Kam, 2005

[110] ICF 2000: Page 142, quality of lobby, level 9

[111] Kam and Fischer, 2002: Section 6, pages 36–41

[112] Solibri 2004; Internet

[113] IFC 2004 Add1: Internet

[114] ProIT 2004: Internet

[115] Kamara et al., 2003: Page 73

[116] Programs 2003

[117] EcoProp: Extracted from a comprehensive printout of the requirements report

[118] EcoProp: Extracted from a comprehensive printout of the requirements report

[119] ICF 2000

[120] ICF 2000: Pages 24–180

[121] ICF 2000: Pages 184–226

[122] Kam and Fischer, 2002: Section 6, pages 36–41

[123] IMSvr 2002: Internet

[124] WebSTEP 2002: Internet

[125] EPM 2003: Internet

[126] Adachi, 2002: Internet

[127] Hemiö, 2002: Internet

[128] Behrman, 2002: Pages 25–26

[129] SABLE 2002: Internet

[130] Solibri 2004; Internet

[131] SABLE 2002: Internet

[132] PLCS 2005: Internet

Published by

Series title, number and report code of publication

VTT Publications 572
VTT–PUBS–572

Author(s)
Kiviniemi, Arto

Title

Requirements management interface to building product models

Abstract

In current AEC practice client requirements are typically recorded in a building program, which, depending on the building type, covers various aspects from the overall goals, activities and spatial needs to very detailed material and condition requirements. This documentation is used as the starting point of the design process, but as the design progresses, it is usually left aside and design changes are made incrementally based on the previous design solution. As a consequence of several small changes and without any conscious decisions to change the scope, this can lead to a solution that may no longer meet the original requirements.

In addition, design is by nature an iterative process and the proposed solutions often also cause evolution in the client requirements. However, the requirements documentation is usually not updated accordingly. In the worst case the changes are recorded just in the memory of the participants, and in the best case in meeting or personal notes. Finding the latest updates and evolution of the requirements from the documentation is very difficult, if not impossible.

This process can lead to an end result which is significantly different from the documented client requirements. Some important client requirements may not be satisfied, and even if the design process was based on agreed-upon changes in the scope and requirements, differences in the requirements documents and in the completed building can lead to well-justified doubts about the quality of the design and construction process.

My observation is that even a simple active link between the client requirements and design tools can increase the use of requirements documentation throughout the design and construction process and facilitate necessary updates of the client requirements. The key limitation is the lack of a theory to link the requirements to the design systems.

A solution to the above mentioned problems can build on the following five main points of departure: (1) design as an information process, (2) existing client requirements documentation and hierarchies, (3) Lawrence Berkeley National Laboratory's Design Intent Tool for technical systems, (4) existing IFC specification and its implementation, and (5) Building Lifecycle Interoperable Software (BLIS) implementation views to the IFC specification. My research is also part of CIFE's Virtual Design and Construction (VDC) framework. Objects in the requirements model specification represent desired product form in the Product-Organization-Process (POP) ontology.

I addressed the challenges by formalizing a requirements model specification which can be linked to a building-product-model-based design model of the project. My research consisted of four phases: (1) analysis of client requirements, (2) development of a requirements model specification and its links to the IFC specification, (3) extension of the BLIS view for IFC implementation, and (4) validation of the requirements model specification.

Based on the requirements analysis, the number of possible requirements is high but only a few of them are used on most projects. However, the linkage of direct and indirect requirements to the design model is complicated and cannot be defined on a project by project basis only. Thus, my requirements model specification is based on an inclusive approach; all relevant requirements which were identified in my research are included in the specification, and each requirement object includes the direct and indirect links to the different levels of detail in the design model.

The specification covers 300 requirements in 14 main and 35 sub-categories. It is based on a synthesis of two large, widely used requirements hierarchies, analysis of requirements in five building programs and spatial requirements in the current IFC specifications. These requirements are organized in the specification into 7 main-level and 30 sub-level requirements objects which have direct links to 5 levels of detail and 2 systems in the building product model plus indirect links to 4 levels of detail and 12 systems. The size and complexity of the specification can be managed by a good user-interface design, which is one of the proposed future research topics.

The main scientific contribution of my research is this requirements model specification, based on the following main concepts: (1) division of a project's data set into four main models; requirements, design, production, and maintenance models, (2) requirements related to the different levels of details in building product models, and (3) direct and indirect requirements. Although the detailed requirements relate mainly to the architectural design, the main concepts of the specification are not domain-specific and apply to a general interface between requirements and building product models. The same link mechanism which is used between objects in the requirements and design models applies also between objects in different design and production models.

My specification defines the structure of the requirements model. Its purpose is to serve as the basis for software development. For AEC professionals it is useful only if implemented into software products. Thus, the main practical implications of my work are that (1) the requirements model specification enables implementation of requirements management applications linked to building product models, and that (2) the use of such applications can improve the management of detailed client requirements in the building process. In addition, I propose some improvements in the current IFC specifications.

One of the goals for my research was to create a basis and a wide framework for future research topics in this area. Thus, the documentation is inclusive rather than exclusive. In general the future research topics can be divided into two categories. (1) Research which expands the requirements model specification, such as the relation between high-level strategic owner requirements and detailed end-user requirements, requirements for other design domains, other parts of the process, and different building types. (2) Research which relates to the use of the requirements model, such as implementation of requirements management applications using model server technology, utilization of requirements history, automated verification of design, and semi-automated design software.

Keywords
buildings, requirements management, interfaces, client requirements, design tools, IFC specifications, linkage, production models, maintenance models, software development

Activity unit
VTT Building and Transport, Lämpömiehenkuja 2, P.O.Box 1800, FI–02044 VTT, Finland

ISBN		Project number
951–38–6655–6 (soft back ed.) 951–38–6656–4 (URL:http://www.vtt.fi/inf/pdf/)		R2SU00818
Date August 2005	Language English	Pages 328 p.
Name of project PREMISS	Commissioned by Tekes, VTT, Confederation of Finnish Construction Industries RT, NCC Oy, Pöyry Oy, Skanska Oy, YIT Rakennus Oy	
Series title and ISSN VTT Publications 1235–0621 (soft back ed.) 1455–0849 (URL: http://www.vtt.fi/inf/pdf/)	Publisher VTT Information Service P.O. Box 2000, FI–02044 VTT, Finland Phone internat. +358 20 722 4404 Fax +358 20 722 4374	

This Ph.D. dissertation approved in Stanford University addresses a current problem of the requirements management in the current building design and construction process: In current practice requirements are recorded in a building program, which is used as the starting point of the design. Later in the design process, however, changes are often made based on the previous design solution. Without any decisions to change the scope, this can lead to a design solution that may not meet the original requirements.

The research is based on the observation that an active link between the client requirements and design tools can increase the use of requirements documentation throughout the process and facilitate updates of the requirements. The key limitation has been the lack of a theory to link the requirements to the design systems, which in this research has been addressed by formalizing a requirements model specification which can be linked to building-product-model-based design models.

The requirements model specification developed in this research is based on three main concepts: (1) division of a project's data set into requirements, design, production, and maintenance models, (2) requirements related to the different levels of details in building product models, and (3) direct and indirect requirements. Although the detailed requirements relate mainly to the architectural design, the main concepts of the specification are not domain-specific and apply to a general interface between objects in different models. The research also proposes some improvements in the current specifications for the IFC (Industry Foundation Classes) – a standard format for the representation of building product models.

Tätä julkaisua myy	Denna publikation säljs av	This publication is available from
VTT TIETOPALVELU PL 2000 02044 VTT Puh. 020 722 4404 Faksi 020 722 4374	VTT INFORMATIONSTJÄNST PB 2000 02044 VTT Tel. 020 722 4404 Fax 020 722 4374	VTT INFORMATION SERVICE P.O.Box 2000 FI–02044 VTT, Finland Phone internat. +358 20 722 4404 Fax +358 20 722 4374

ISBN 951-38-6655-6 (soft back ed.)
ISSN 1235-0621 (soft back ed.)

ISBN 951-38-6656-4 (URL: http://www.vtt.fi/inf/pdf/)
ISSN 1455-0849 (URL: http://www.vtt.fi/inf/pdf/)

Arto Kiviniemi