PRODUCT MODEL BASED DESIGN
OF
PRECAST FACADES

Vesa Karhu

Licentiate Thesis
Division of Construction Management and Economics
Department of Real Estate and Construction Management
Royal Institute of Technology
Stockholm, Sweden
1997
PREFACE

The research presented in this thesis has been conducted at VTT Building Technology and the Royal Institute of Technology (KTH) between 1992 and 1997. KTH is for the moment the leading institute in product modelling research in the Scandinavian countries.

I would like to thank my supervisor Prof. Bo-Christer Björk for encouraging me to continue the work. I would also like to thank Mr. Matti Hannus for initialising the research and for valuable comments during the initial phases of the work.

This research was funded by the Technical Research Centre of Finland (VTT), The Finnish Association of Construction Products Industries (RTT) and the Technology Development Centre of Finland (TEKES). The companies that participated in the testing were the architectural company Arkkitehtitoimisto Innovarch Oy, the structural engineering company Tampereen Juva Oy and the manufacturer Partek Betonila Oy. The final architectural software was developed in Studio Kivi Oy. I would like to thank the persons from the above companies.

Additionally I would like to thank Prof. Brian Atkin for the English language aspects, Associate Prof. Jan Bröchner for valuable comments and suggestions. I would also like to thank my colleagues at VTT Building Technology and KTH for their valuable comments and suggestions.

An important factor during the research has been my kayaking hobby. Hard training is balanced by hard work. Special thanks go to my common-law wife Katarina who has given me support during the last phases of the research.


Vesa Karhu
ABSTRACT
In Finland, approximately 80% of the facades of buildings are manufactured as precast units. Currently one of the obstacles to making the overall design and construction of precast building facades more efficient is the inefficient exchange of data about facades between architects, structural engineers and precast element manufacturers. The product model approach seems to offer a new methodology for data exchange and sharing which would solve many of the current problems. This thesis presents the results of research in which this approach was tested.

The prevailing way of designing facades was chosen as a reference process model. Based on an analysis of data needs in the different stages of the process a product data model of a facade was developed. The product data model was restricted to facades only and does not include other information about the building. Central data structures in the conceptual schema define how a precast concrete facade consists of precast concrete units, i.e., elements. Structural wall layers that may have openings form the elements.

The conceptual schema was implemented as a prototype which was based on existing software, modified and further developed. The prototype was tested by an architectural design company, a structural design company and a manufacturer. The main conclusion of testing was that the data produced in the architectural design is directly usable in further design. The structural or element design may use the architectural data as such. Also, it is possible to create applications that take into account the architect's preferred design approach.

KEYWORDS: facade, precast concrete, data exchange, object oriented, architectural design
SAMMANFATTNING


NYCKEORD: Fasad, prefabricerad betong, dataöverföring, objektorienterad, arkkitektprojektering
<table>
<thead>
<tr>
<th>CONTENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>PREFACE ................................................. 3</td>
</tr>
<tr>
<td>ABSTRACT .................................................. 5</td>
</tr>
<tr>
<td>SAMMANFATTNING ........................................ 7</td>
</tr>
<tr>
<td>CONTENTS .................................................... 9</td>
</tr>
<tr>
<td>LIST OF ABBREVIATIONS ................................ 11</td>
</tr>
<tr>
<td>1. INTRODUCTION .......................................... 13</td>
</tr>
<tr>
<td>1.1 Background ........................................... 13</td>
</tr>
<tr>
<td>1.2 Research questions .................................. 14</td>
</tr>
<tr>
<td>1.3 Scope and phases of the research ................ 15</td>
</tr>
<tr>
<td>1.4 Structure of the thesis ............................. 16</td>
</tr>
<tr>
<td>2. RELATED RESEARCH .................................. 19</td>
</tr>
<tr>
<td>2.1 Methods for integration ............................ 19</td>
</tr>
<tr>
<td>2.2 Product data model vs. product model .......... 20</td>
</tr>
<tr>
<td>2.3 STEP standardisation ............................... 21</td>
</tr>
<tr>
<td>2.4 Generic building product model proposals ...... 22</td>
</tr>
<tr>
<td>2.5 Aspect models ........................................ 24</td>
</tr>
<tr>
<td>3. METHODOLOGY ......................................... 27</td>
</tr>
<tr>
<td>3.1 Methods used in this research ................. 27</td>
</tr>
<tr>
<td>3.2 Choice of modelling tools .......................... 27</td>
</tr>
<tr>
<td>3.3 Object definition cards ............................. 29</td>
</tr>
<tr>
<td>3.4 Kinds of evidence .................................... 30</td>
</tr>
<tr>
<td>4. THE BUILDING DESIGN PROCESS ................ 31</td>
</tr>
<tr>
<td>4.1 Level of prefabrication in Finland .............. 31</td>
</tr>
<tr>
<td>4.2 The overall design process ....................... 31</td>
</tr>
<tr>
<td>4.3 The detailed design stage design ............... 34</td>
</tr>
<tr>
<td>4.4 Analysis of current problems .................... 36</td>
</tr>
<tr>
<td>5. THE PRODUCT DATA MODEL OF A FACADE ........ 39</td>
</tr>
<tr>
<td>5.1 Modelling procedure ............................... 39</td>
</tr>
<tr>
<td>5.2 The decomposition hierarchy of facade .......... 41</td>
</tr>
<tr>
<td>5.3 Structural layer ....................................... 42</td>
</tr>
<tr>
<td>5.4 Opening ................................................ 43</td>
</tr>
</tbody>
</table>
LIST OF ABBREVIATIONS

Many of the following abbreviations are defined in the product modelling glossary [PM Glossary 1996].

AP
Application Protocol. A mechanism of the STEP standard for defining well-defined subsets of the total resource classes in STEP to be used in conceptual models supporting the data transfer needs of particular application areas.

BEC
The BEC-system includes a number of software modules for the receiving and manipulation of graphical BEC-files and for the management of tables. It also contains specifications of precast concrete structures. BEC was developed in Finland during the 1980s.

COMBINE
A European research project financed by the EU through the Joule programme. Computer Models for the Building Industry in Europe.

DXF
Data exchange format used in AutoCAD.

EXPRESS
An information modelling language in textual format. It is used for the definition of the international STEP product model standard.

EXPRESS-G
A graphical subset of the EXPRESS information modelling language.

GARM

IDEF0
A format for defining activity models as hierarchical diagrams. IDEF0 is a subset of SADT (see below).

NIAM
A graphical information modelling language. The abbreviation stands for "Nijssen Information Analysis Method".

OOCAD
Object-oriented CAD. A general purpose product modelling environment developed at VTT in the early 1990s.

OXF
Neutral Object eXchange File. Defined as part of the OOCAD model definition.
RATAS
Computer-aided design of buildings. An abbreviation for research, development and standardisation projects in Finland. RATAS-work is formally organised by a committee under the Building Information Institute.

SADT
A format for defining activity models as hierarchical diagrams (see also IDEF0 above). The abbreviation stands for Structured Analysis Design Technique.

STEP
Standard for the Exchange of Product Model Data.
1. INTRODUCTION

1.1 Background

In Finland, approximately 80% of the facades of buildings are manufactured as precast units. The design and manufacture of precast elements thus forms a substantial part of the total quantity of structural engineering activities. Currently, one of the obstacles to making this process more efficient is the inefficient exchange of data about facades between architects, structural engineers and precast element manufacturers.

Many of the problems that are encountered during the traditional building design process are caused by the lack of data exchange standards. Architectural design produces drawings and information in textual form about a building (see Figure 1). Today, these documents are predominantly produced using CAD systems and word processors, but still resemble traditional manually produced documents. On the other hand, manufacturers have computerised systems where digital data describing the parameters of precast elements are handled automatically. Consequently, data produced during architectural design do not meet the requirements set by the manufacturer. The cost effects of the architectural solution are in many cases not taken into account during the earlier design stages. Also, feedback from manufacturers is not used and tends to lead to standard design solutions. Current architectural design practice no longer meets the requirements as far as the element manufacturer is concerned.

In order to solve some of these problems, the so-called BEC standard was developed in Finland in the late 1980s [BEC 1991]. The aim of the standard was to standardise the exchange of data of precast concrete structures. The standard includes definitions of precast concrete structures, tables and a number of software modules for receiving and manipulating precast element data.

The product model approach seems to offer a way forward for data exchange and sharing which would solve many of the current problems. In product modelling, data are structured systematically and may thus be processed by computers without human interpretation. In product modelling, a clear distinction should be made between the term product data model and the product model. The product data model is, for instance, the information structure and attributes such as the width and height of a facade. The product model is a computer interpretable description of the structure, for instance, a facade element that is 5 m wide and 2.5 m high.

The product data model provides the template from which a number of different building descriptions (individual product models) can be defined. Put another way, the product data model provides the structure of a data
base system which can be filled with information about individual buildings.

Research covering product modelling has been undertaken in many countries, but the models presented have been somewhat theoretical. Testing of the models has generally been limited and, thus, they cannot be implemented directly.

Figure 1. Current situation of data exchange in Finland. Data from architectural design is transferred to structural design as drawings and textual data. Manufacturers may receive structured data, for instance, in BEC-format [BEC 1991].

1.2 Research questions

The integration of the design of a precast facade is of considerable importance to improving the overall efficiency of the building process. Traditional architectural design is not sufficiently efficient to enable complete integration. The questions that arise are:

- what kind of approach should be used to integrate the architectural design of precast facades with the design and manufacture of elements?
- how can the effects of this integration be verified and evaluated against the traditional design procedure?

Architectural design tends to concentrate on visible objects such as surfaces and edges of windows or door openings. Thus, the first schematic designs do not contain much data on concrete for use in element design or manufacture. Accordingly, the affected designs must be completed by the structural engineer. The various parties involved in a building design project have different ideas, concepts, information and computer applications concerning what shall be designed and manufactured. Further questions arise:
• what are the characteristic features of the traditional architectural design process?

• what kind of standardisation is needed to achieve integration?

The question about the features of the design process is interesting when compared to the integration requirements. Shall a process in itself remain the same or should the process be re-engineered? However, this question falls outside the scope of this research and is not, therefore, developed further.

1.3 Scope and phases of the research

The scope of the research presented in this thesis was the integration of the architectural design process of facades with the manufacturing process, by defining a product data model of a precast concrete facade. Also, the scope included the formalisation of the results into guidelines and instructions.

The primary focus was on data exchange, from the architectural design to the element design and manufacturing, during the tendering phase and the initial phase of the element design.

The key phases of the research were as follows.

• Define a basic activity model of the building design process of precast concrete facades emphasising architectural design.

• Analyse the problems occurring in the current design process.

• Define a product data model of a precast concrete facade.

• Define check-lists of the data input and output requirements.

• Develop prototype software based on the product data model.

• Test the prototypes with data from a real project.

• Conclude if a product model based architectural design process enhances the whole building design process.

• Propose guidelines for using the product model based approach in the architectural design of facades.

The research approach, comprising its six phases, is shown in Figure 2. The figure uses the SADT notation (see Appendix A for more details on SADT method). The software developer and other parties in the project had a role during the development of software prototypes and testing. These can be seen as mechanisms to the activities A4 and A5. All other activities were performed by the author.
1.4 Structure of the thesis

Chapter 1 introduces the subject and discusses the current problems in the design and manufacture of precast concrete elements. It also describes the scope and the phases of this research.

Chapter 2 describes related research that has been undertaken in Finland and in other countries. Published product modelling work has been theoretical for the main part.

Chapter 3 introduces the methods used in this research. The traditional architectural design process was modelled using the IDEF0 method. The product data model was modelled using EXPRESS and its graphical counterpart EXPRESS-G. The chapter also discusses some principles for the validation of product data models.

Chapter 4 presents the traditional building design process and the problems encountered during the process, and is described from an architectural perspective.

Chapter 5 discusses the criteria for a product data model of a facade. The product data model of a precast facade is defined in this chapter. The focus is on structural layers, openings, surfaces and edges of layers.

Chapter 6 introduces the check-lists that are used to help in defining the data requirements of the different parties during an overall design process.

Chapter 7 presents the software development procedure. Prototypes were created for research purposes, whereas the final version was developed by an architectural company.
Chapter 8 presents the test design process and the results. The test was based on the traditional building design process, but with a product model approach. A number of ideas and limitations are also discussed.

In Chapter 9 conclusions are presented. The traditional architectural design process may use the product model approach as this provides more accurate data definitions.

Appendix A presents a summary the function modelling techniques SADT and IDEF0. Selected examples are provided.

Appendix B presents a short introduction to the EXPRESS language and its graphical counterpart EXPRESS-G. The EXPRESS language is used in STEP technology.

Appendix C presents the full schema of the facade in textual format.

A shorter version of this thesis has been published in the Electronic Journal of Information Technology in Construction [Karhu 1997].
2. RELATED RESEARCH

2.1 Methods for integration

Luiten [1994] recognises six ways in which communication between computer applications may be accomplished:

- closed integrated systems;
- open integrated systems;
- communication with low semantic representations;
- classification and coding;
- product modelling; and
- knowledge-based technologies.

Closed integrated systems are vendor specific systems covering many applications. Some early examples of building design systems are those developed in the UK in the 1970s [Jones 1982]. Open systems, on the other hand, allow communication with other programs, but they use an internal data exchange. Low semantic representations are usually limited to, for instance, descriptions of the geometry. An example is the DXF format. Classification or coding provides a method for communication on a higher level. An example is the SfB classification for project information [CIB 1977] that comprises functional building elements, activities and resources. Product modelling provides a means for communicating on a higher level. Knowledge based systems concentrate on the formalisation of both data and knowledge.

During the 1970s and early 1980s, CAD systems became popular and were considered to be the primary vehicle for integration. CAD systems used geometrical information based on a number of different modelling methods (wire models, surface models, etc.). The geometry cannot be used as the only means of integration because [Luiten 1994]:

- the shape of the product is not stable during the design process;
- information exists before the shape is chosen; and
- parties in a project use different shape representations.

Since the early 1980s other integration techniques used in CAD systems include layering [Björk et al. 1996] and reference file techniques. These systems concentrate on splitting up an overall building model according to building elements and responsibility for their inherent information, as well as facilitating views of the models and supporting the plotting of drawings.
2.2 Product data model vs. product model

During the late 1980s, the concepts of geometric modelling evolved into product modelling. The product model approach seems to offer the tools needed to overcome some of the persistent problems in data exchange.

In a product model, data about an artefact are arranged in a systematic way using object oriented data base principles. Traditional CAD software primarily models the graphical appearance of the building parts. The product model approach describes building parts directly. This approach has been a prime research subject for some ten years. It is currently receiving increasing attention both in standardisation efforts, such as the ISO STEP process [ISO 1994b] and the Industry Foundation Classes initiative [IAI 1996] as well as in the development of commercial software.

A central concept in the following presentation is the conceptual schema. In the theory of conceptual modelling or data base design the conceptual schema denotes the formalised description of the structure of the information stored in an information base [Boman et al. 1991], [ISO 1985]. Conceptual schemata may be used to structure database applications as diverse as census records, banking applications, missile guidance systems or descriptions of aircraft.

A product data model is defined as [PM Glossary 1996]: A particular type of conceptual schema, which structures the information needed to describe a physical artefact, designed and manufactured by man. The central object classes of product data models describe the functional parts of the artefact and assemblies formed by them, rather than concepts needed for representing the parts in different kinds of documents.

A product model is a computer-interpretable description of an artefact, structured according to some predefined product data model.

Figure 3 illustrates the difference between a conceptual schema and an information base. It also provides an example of two different implementations of the same information base using a data base and the STEP physical file format [ISO 1994] which is presently used for product data exchange. The information base presented in the figure contains the actual values of a real designed object such as the facade and its position (x-coordinate, y-coordinate, etc.), shape, a unique id, etc.

In the literature and in practice, the term product model is often used to describe the conceptual schema even though the term product data model would be more correct. For a general discussion of product models and product data models see Björk [1995].
Figure 3. An example of a part of a small conceptual schema in textual format (upper left corner) and graphical presentation (upper right). The information base is presented in the lower left corner as a data base and as a STEP physical file in the lower right corner.

2.3 STEP standardisation

The STEP standardisation effort in product modelling is important both from the industry viewpoint, but also because it has provided a lot of impetus for research in the domain. The acronym STEP stands for Standardisation for the Exchange of Product model data. It is organised under the International Organisation for Standardisation (ISO). The standard currently contains 12 parts.
The modelling language used in STEP is EXPRESS (see further details in Appendix B) and its graphical counterpart EXPRESS-G. Other graphical languages that were used earlier in STEP are NIAM [Nijssen et al. 1989] and IDEF1X [Appleton 1985]. Data exchange may be undertaken using a STEP physical file format [ISO 1994]. The STEP physical file format is structured for any product data for which the schema is specified in the EXPRESS language.

The STEP development focuses on standardising conceptual models with different scopes. There are two types of conceptual models [ISO 1994b]:

- integrated resources; and
- application protocols.

The integrated resources specified in STEP define a generic information model for information about any product (i.e., geometry, topology, product configuration). These resources are by themselves not specific enough to support the information requirements of particular applications without the addition of specific constraints, relationships and attributes. Application protocols (APs) are developed for this purpose.

An AP provides a mapping to show how the interpretation of integrated resources is used to meet the information requirements of a particular application [WWW STEP 1996a, ISO 1994b]. These will be interpreted by selecting appropriate resource constructs. Their meaning is refined by specifying appropriate constraints, relationships and attributes.

Examples of application protocols currently under development by the building construction committee of STEP include:

- AP225: Structural Building Elements Using Explicit Shape Information;
- AP228: Building Services: Heating, Ventilation and Air Conditioning (HVAC); and

### 2.4 Generic building product model proposals

In Finland, the RATAS model [RATAS-committee 1988] proposed an entity-relationship model which was enhanced with inheritance as a general framework for a building product data model. Further work in Finland has resulted in the OOCAD model, which is a generic data model proposal based on a composition of objects with part-of relationships [Serén et al. 1993]. Figure 4 illustrates the RATAS building product model framework [Björk et al. 1989].
Several foreign authors have proposed generic building product models. One of the first building product models was the General Architecture, Engineering and Construction Reference Model (GARM). The GARM model may be used to model any product and its characteristics in different life cycle stages [Gielingh 1988].

The basic class of the GARM model is called a product definition unit (PDU). These may be specialised according to five classifications. The classification according to life cycle covers seven stages: as required, as designed, as planned, as built, as used, as altered and as demolished. Particularly important subtypes are the functional unit that corresponds to the as required stage, and the technical solution that corresponds to the as designed stage. The GARM model was intended to be instantiated directly as a product model.

Figure 4. The basic framework of the RATAS building product model [Björk et al. 1989].
The GARM model has been further developed in the IMPPACT reference model [Gielingh et al. 1993] which uses conceptual models of different scopes. The IMPPACT model tries to combine both process modelling and product modelling. The process modelling concentrates on production.

The AEC Building Systems Model is focused on the functional systems of which a building is composed [Turner 1990]. The functional systems are, for example, enclosure, structural and mechanical systems. It differs from the GARM model in that its analysis is oriented to the functions of building parts rather than to more generic classifications.

2.5 Aspect models

Recently, there has been a shift from all-encompassing product models to conceptual schemas describing more limited domains. Terms such as aspect models have been used to describe such types. A good example of an aspect model is the Integrated Data Model (IDM) provided by the COMBINE project [Dubois et al. 1992]. The IDM model concentrates on building energy and HVAC system information. An example of an entity in the IDM model [COMBINE 1995] is the wall that is defined as:

```plaintext
ENTITY wall
    SUBTYPE OF (elementary_space_enclosing_element);
    loadability : load_bearing ;
    has_wall_type: OPTIONAL wall_type ;
END_ENTITY ;
```

The loading attribute receives a Boolean value. The wall type is defined as

```plaintext
TYPE wall_type = ENUMERATION OF (
    framed, non_framed);
END_TYPE;
```

The wall has also a construction type which may be composed by layers.

For the particular field of structural design, some models have been proposed. Hannus [1990] discusses CAD systems based on product modelling for precast concrete structures. The main topics are the interchange of data between CAD systems, product modelling and implementation.

Dale [1991] uses object oriented modelling techniques for structural design. He also uses GARM as a reference model. He mentions the multiple representations of the structural design. The different models are a geometric model, structural model and a computational model. Dale’s model has been tested using a plate girder bridge model.

Lavakare uses the product model approach for a structural steel framing data model [Lavakare et al. 1989]. His model is divided into eight hierarchical levels where the entity building is on the highest level.

Karlshøy [1994] presents a construction information model KONIM which is a building product model. It is based on the principle of references, where
only a minor part of the building information is stored in the core of the product model. The remaining information is stored in the references.

Luiten [1994] used precast concrete structures to test product models of beams, columns, hollow core slabs and connections. The goal in his research was to develop a strategy to support integration of design and construction in building projects. He proposes a building project model (BPM) that integrates product, activity and resource information.

CIMSTEEL [Watson 1995] uses the product model approach for structural steel framing data models. Based on this work, a STEP application protocol (AP 230: Building structural frames: Steelwork) is under development for this area.

Attempts have been made in Finland to standardise the data exchange of precast concrete structures. Before the emergence of the product model technology these attempts used some of the principles of product modellers. In the 1980s, various software modules were developed for receiving and manipulating graphical files and for the manipulation of tables of data [BEC 1991]. An example of a STD card defined in the BEC project is shown in Figure 5.

The example card describes a sandwich element consisting of several layers including insulation. The front page of a STD card contains information in a graphical format. The back page contains the definitions in textual format. This textual format uses a syntax close to the LISP language, and is a mechanism for data exchange.

In the 1990s, so-called object definition cards were developed in Finland. These define a standard data structure for the description of various building components such as windows and doors.

All in all, many of the building product model proposals that have been proposed in the literature are rather theoretical and have not yet reached a stage where they can easily be implemented by software developers. Thus, there is a need to produce more detailed proposals, possibly for quite limited application areas, and to test them in real-life design situations. This has been one of the main motivations for the research described in this thesis.
**Figure 5.** An example of a STD card defined in the BEC-standard [BEC 1991]. The front page and a portion of the back page are shown.
3. METHODOLOGY

3.1 Methods used in this research

In this research the following general methods were adopted.

- Gather basic information and data in interviews with clients, architects, structural engineers, contractors, manufacturers and software developers.
- Use formal process modelling methods for activity modelling.
- Use formal conceptual modelling methods for product modelling.

The interviews were conducted as shown in Table 1.

Table 1. The number of companies and persons interviewed.

<table>
<thead>
<tr>
<th>Role</th>
<th>Companies</th>
<th>Persons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Architect</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>Structural engineer</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>Contractor</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Manufacturer</td>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>Software developer</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>Other</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>16</strong></td>
<td><strong>32</strong></td>
</tr>
</tbody>
</table>

The category ‘other people’ means researchers and representatives from organisations such as The Finnish Association of Construction Products Industries, VTT Building Technology, etc..

3.2 Choice of modelling tools

Systematic modelling methods were applied in the project. A number of modelling methods for both activity modelling and conceptual modelling are available. Even so, tools that combine these two aspects are relatively few.

In the project, methods included in the STEP standard development process were used. The first steps in the development of an AP are to define the process in which data exchange is taking place and a conceptual schema defining the information requirements of this data exchange (a so-called application reference model, ARM).

A basic process model of the design of precast concrete facades and a product data model of a facade were developed in this project. The product data model that was defined in the study could be developed further to a STEP application protocol, which would have involved redefining it to
reuse the data structures of STEP resource entity definitions. However, this was deemed to be outside the scope of the research because limited resources were available. The development of an AP would require international co-operation and a contribution of about 3-4 years of work.

The building design process from briefing to construction and final approval was modelled using SADT-charts [Marca et al. 1987]. Special interest was focused on tasks that are of importance in architectural design.

EXPRESS and its graphical counterpart EXPRESS-G [ISO 1994a] were used for defining the conceptual schema of the facade. The EXPRESS language defines entities that are classes of information defined by common properties. For more details on EXPRESS see Appendix B.

An example of EXPRESS is given below.

```
ENTITY facade
  SUPERTYPE OF (precast_facade ANDOR cast_in_situ_facade);
  position : position ;
  id       : id ;
  shape    : shape ;
  balconies: OPTIONAL SET [1:?] OF balcony;
  channels : OPTIONAL SET [1:?] OF channel;
END_ENTITY;
```

An entity named facade is defined in this example. The entity facade is a supertype of either precast facade or a cast in situ facade. It can also be both. The facade also has a number of attributes, for instance position.

The graphical representation of an entity is a box, as Figure 6 shows. Relationships are shown with normal lines, with optional attributes denoted by dashed lines. Thick lines are used to represent an inheritance relationship, i.e. a subtype and a supertype relationship.

![Figure 6. The graphical representation of an entity and its relationships.](image)
A number of both commercial and freeware STEP tools are available [WWW STEP 1996b]. Examples of the functionality of the software tools include:

- browsers for traversing EXPRESS models;
- compilers that convert EXPRESS into a programming language;
- converters that convert between modelling languages;
- decommenters that strip comments from EXPRESS source code;
- editors that assist in creating EXPRESS models in textual and graphical format;
- parsers that check the EXPRESS syntax and possibly semantics.

Also, a number of tools may be used to process the different files such as importing or exporting files in STEP physical file format. Some tools have incorporated many of the functionalities into one software package.

### 3.3 Object definition cards

So-called product model object definition cards of the object classes included in the precast facade data model were created. Such cards have recently been adopted in Finland by the RATAS committee [RATAS 1996a] in order to facilitate communicating product model definitions with software developers and end users. An example of an object definition card is shown in Figure 7.

![EDGE Card](image)

**Figure 7. An example of an object definition card [RATAS 1996b].**
The basic concepts behind the object definition cards are that:

- an overall summary data card is used to define a rough hierarchical composition of the facade;

- exact definitions of each object may be found in individual cards; and

- the cards may be updated independently from other cards.

The object definition cards contain a short explanation in text format, an EXPRESS-G-diagram and an optional picture of the object. The card may also contain an additional explanation in text-format and the EXPRESS-code of the object, for instance different edge shapes. The EXPRESS-code is also seen in Figure 7. Twenty different object definition cards were created in this research and they are found in reference Karhu et al. [1994].

### 3.4 Kinds of evidence

Clayton et al. [1997] discuss the subject of how to compare an innovative design process, such as the product model presented in this thesis, with a conventional process. He argues that the validation of the results from research (for example, a product data model proposal) usually falls into a few categories:

- logical argument;

- a worked example;

- a demonstration; and

- a trial.

The first argument, the logical argument, is not particularly useful in the case at hand in this research. A worked example, on the other hand, attempts to provide more evidence, but the examples are merely a complement to the logical argument. The examples are detached from the real world so that they suit the logical argument. The demonstration raises the standard a little, although the demonstration may be tailored to fit the research. Thus, it will be difficult to determine whether or not it will be of use to other practitioners. The last category, a trial, usually needs a larger effort to develop software since the application must be robust and bug-free.

In terms of Clayton’s categories the worked example and a demonstration were used in the initial software development phase of this research. The final test, the design of the facade, may also be categorised as a worked example though the data used was taken from a real project. These results are discussed in Chapter 8.
4. THE BUILDING DESIGN PROCESS

4.1 Level of prefabrication in Finland

Approximately 80% of facades are manufactured as precast units in Finland. This has been the prevailing situation in the 1980s and early 1990s. The use of this type of technology has resulted in a design, construction and manufacturing process which differs somewhat from the traditional in-situ construction process. The division of tasks and the exchange of data between the architect, the structural designer, the manufacturer of the elements and the contractor are critical factors in the strive to achieve an efficient design and prefabrication process.

In order to help us understand this division of tasks and exchange of data, the use of formal process modelling tools can be extremely useful. Such models can describe the current process and its problems more clearly than verbal accounts and can also be used as a basis for reengineering efforts. Such models can also support the development of application software which facilitates in integrating the process.

In this chapter a model of the building process which in the study was defined using the SADT notation is briefly discussed, and selected parts of the overall model are presented. The syntax and notations used in SADT-diagrams are explained in more detail in Appendix A. Since the emphasis of the thesis is on the product data model to be presented in section 5 the process model is not described in all its details. The complete activity model may be found in reference Karhu et al. [1994], in Finnish.

4.2 The overall design process

The building design process is in most countries carried out in a more or less standardised form, often particular to the country in question (a good example is offered by the British bill-of-quantities practice). Such standardisation is important, for instance in order to define the exact data contents of documents issued at the end of different process phases (e.g., building permit documents), as well as the responsibilities of different process participants. Mostly these standards have been formalised in the form of industry guidelines, check lists, standard agreements etc..

In Finland the traditional building design process is divided into stages according to task lists [RT 1995a, RT 1995b, RT 1995c, RT 1995d]. These lists have been issued by the Building Information Institute as the result of committee work:

- briefing;
- programming;
- global design;
- detailed design;
- design during construction; and
- design for usage and maintenance.

In this research project the design process was modelled using the SADT notation, based on the above-mentioned task lists. The model is furthermore based on the premise that all data are accumulated and processed further, and consequently become more accurate as the designs reach the production stage. The highest level of the process model is shown in Figure 8.

The traditional building design process is controlled by general knowledge and data. Such controls include, for instance,
- building regulations;
- legislation;
- the overall economic situation; and
- the market situation.

These are not shown in the diagrams separately. Also, strategic decisions etc. of a company are not shown in the model.

Figure 8. Produce and manage building design data.

The process starts with drawing up a project brief and a programme. These activities are normally performed by the client and the main designer. The
main architectural design process begins in the activity A3 Make global design. The architect’s tasks during briefing and programming are usually considered as the main designer’s tasks. The main designer may be an architect or a group of persons and the tasks include co-ordination of the design work.

There are other published divisions or classifications of the construction process. An ISO technical report divides the process into the design phase, production phase, use phase and the demolition phase [ISO 1993]. The design phase includes activities such as briefing, environmental and space design and constructional design. The integrated building process model (IBPM) divides the design process into the following activities: understand functional requirements, explore concepts, develop schematic, develop design, communicate design to others and maintain design information and models [Sanvido 1990]. These activities overlap partly the Finnish model of building design process. Luiten analyses the building process from the interaction point of view [Luiten 1994]. The process is divided into three subactivities: design building, manage construction and construct building. The model based on the existing Finnish guidelines was chosen as a starting point for this research since it reflects local conditions and the prevailing practice better than some of the foreign alternatives.

The project brief is a collection of basic information provided by the client concerning mainly space requirements. The information consists of needs, requirements and possibilities. The subactivities are:

- analyse present situation;
- define requirements;
- study alternatives for space acquisition; and
- prepare programme decision.

The project brief may lead to the definition of a building programme if the brief indicates a substantial change in spatial needs which cannot be accommodated through renovation or renting. The actual building program is usually defined by the client. In this process the design instructions are assembled. They contain:

- project-specific instructions;
- standards;
- instructions concerning documents;
- quality class definitions;
- special requirements for design;
- usage of quality management systems; and
- decision procedures concerning design solutions, etc.
The activities during the global design stage are based on the output of the briefing and programming phases. The global design stage is further divided into three main phases.

- The first phase yields a basic solution in which masses and general site designs are made.
- The second phase yields a proposed solution, which is based on a chosen basic architectural solution. Feasibility analyses from the structural point of view are carried out.
- The third phase, scheme design, is used to elaborate designs for the application of a building permit. The building permit can be obtained within one month for a normal building in Finland.

The architect's main decisions concerning the facade are repeatable visible surfaces, openings and some details, such as details of windows in the global design stage. The facade is designed as a whole. In such cases where the technology used is precast element, the facade is divided into elements only later by the structural engineer.

4.3 The detailed design stage design

In the following we will only concentrate on a process where precast elements are used. During the detailed design stage all the parties become actively involved (architect, structural designer and manufacturer of the building elements) and data exchange becomes essential. In the following the part of the model for the detailed design stage which concentrates on the facades is further elaborated (Figure 9). The process for designing the facade begins with the definition of the general shape of the facade. Openings and surfaces are usually designed next. Tiling may be designed if it is needed. In the next activity, A4124 Define element division and joints, it should be noted that the element division may be determined either by the architect or by the structural engineer, depending on the agreement that defines the responsibilities of each participant. The figure shows the alternative where the structural engineer defines the element division and joints. The last two activities are used to define some properties of the structural layers and the edge shapes.
Figure 9. A part of the activity model for designing of a facade.

On a more generic level a typical information flow in the traditional building design process (when precast elements are used) goes from the architect to the structural engineer and further to the manufacturer, see Figure 10. The data from the architectural design are usually presented in the form of drawings (today predominantly produced using computers) which need to be interpreted by humans on the receiving side.

Figure 10. A typical information sequence during a building design process during the design of structural prefabricated components.
Structural engineering and prefabrication applications can exchange data more directly without human interpretation since many software systems controlling the manufacturing process are able to receive data in some standardised formats. Some feedback information can also be fed back from the manufacturing process (dashed line).

4.4 Analysis of current problems

Information concerning present day problems, related to this process, was gathered in interviews with experts from the companies that collaborated in the research project (see also Table 1).

Current problems include:

- The data concerning the facade which are produced during the different phases of the architectural design process do not meet the information content and format requirements of the other parties using the information as input to their own activities.

- There is insufficient feedback of requirements, experience data etc. from the latter stages back to the architectural design

The traditional building design process as such no longer corresponds to the demands of today. If contractual aspects are set aside, one of the main causes of problems outlined above has been the lack of standard data definitions for information exchange in various design stages. This leads to a variety in contents and accuracy in the design documents. Paper drawings may be interpreted by humans but not by computers.

Traditionally, data exchange involving architectural design is done via paper documents, i.e., drawings, though the design data produced is in digital format. The usage of paper documents is caused partly by the lack of data exchange standards and partly by the fact that the data contents have not been agreed on beforehand. The results of the architectural design are passed to the structural engineer who then uses his own software applications to do the appropriate feasibility analyses. The documents produced are drawings and written specifications. These may be unstructured and their data accuracy and reliability is often insufficient. The input of data to other systems is made by manual inspection of the drawings despite the fact that CAD-systems may have been used to produce these drawings in the first place. This often leads to misinterpretation of the data.

At the end of this process, it is difficult for the manufacturer to estimate costs. Inaccurate designs produced during the tender stage cause difficulties in determining the costs. In many cases the cost objectives that were set during programming have to be revised or a lower quality level must be accepted. On the other hand, the data from the structural engineering design may be passed to the manufacturer in a computer interpretable format.
Tenders are based on the experience of the person who calculates them and on using typical elements as a basis for the calculations. A tender offer by the manufacturer is based on using a typical element as the basis for the cost estimation. The choice of this typical element is in many cases not correct which gives a wrong total cost effect. Experience from previous projects and solutions is not used sufficiently. The element design is in many cases done in a hurry. This easily leads to a lower level of quality.

The information process described above often leads to reuse of standard design solutions. It leaves little or no time for innovations and new technology suggested by the manufacturer (feedback). There are examples of feedback from the manufacturer to the architectural design but the feedback is usually not sufficient to enable new solutions instead of standard solutions. Luiten [Luiten 1994] points out that there is no formalised way to exchange comments or suggestions from construction to design. More information exchange is needed.
5. THE PRODUCT DATA MODEL OF A FACADE

5.1 Modelling procedure

In the following section the product data model of a facade, which was developed in the study, is presented. A schematic view of the procedure for defining the product data model, using the SADT notation, is shown in Figure 11. It may be noted that product modelling is not an exact science, in the sense that the requirements that a model should fulfil may well be satisfied with many alternative solutions. Thus, the activity of developing a product data model can in some respects be compared with the design of a building.

In this case the work started with a survey of existing information sources. The entities and their attributes were defined next. An evaluation at this point was also of importance since some entities were not accepted or were further developed during the research. The full product data model of the facade was assembled in the last activity. Some corrections were also needed at this point. These corrections are shown as a control for earlier activities.

Three major sources of information were used as input to the definition process. These consisted of:

- The data specifications of the BEC-project, which had been carried out in the late 1980s;
- Existing drawing material; and
- Interviews with the practitioners who participated in the project.

Of these, the BEC system [BEC 1991] contained information about the roles and parties, activities and information communication during a typical construction process. The specifications indirectly include information structures about facades, although these do not define the facade as such, but merely specify the information to be included in the drawings. The BEC system also contains specifications about precast elements such as beams, slabs and wall elements, documented in so called STD cards. The data needed by the manufacturer is also documented, but on a general level only. In addition to this typical facade and element drawings were studied and interviews with architects, structural engineers and manufacturers were conducted.

Based on this input information the entities and attributes of the model were chosen. The criteria on how to choose these were also set prior to defining the entities. The criteria were as follows:

- entities and attributes shall be understandable to practitioners, i.e., architects, structural engineers, manufacturers;
• entities and attributes shall be implementable in current software platforms; and
• entities shall have a close correspondence with real life objects.

The structural layer is a typical entity. An element consists of a number of structural layers. The outer layer is usually designed by the architect whereas the inner layers are determined by the structural engineer. Thus, a fluent information exchange is needed. The surfaces determine the manufacturing sequence. The casting of a structural layer, for instance, is done with the element face down which means that the tiles must be laid first on the mould.

Figure 11. The procedure used in this research to assemble the product data model of the facade.

In the following the model is presented in the following way. First the overall abstraction hierarchy is discussed. After that the major parts which are designed during the architectural design process (see also Figure 9) are shown and discussed. These include:

• structural layers;
• openings;
• surfaces; and
• edges.

Finally, the product data model as a whole is presented.
5.2 The decomposition hierarchy of facade

The facade is the front of a building usually given special architectural treatment [Webster 1996]. The hierarchical decomposition of a precast facade into its parts is as follows: a precast concrete facade consists of precast units, i.e., elements. An element in turn consists of structural layers. A structural layer is, for instance, an outer wall panel or an insulation layer. Structural layers and openings have edges, which are a certain 2D-shape. Layers may be arbitrary in shape. A precast facade has also element joints and an entity called element division. The EXPRESS-G subschema of the above classes is shown in Figure 12. Exact definitions may be found in reference Karhu et al. [1994].

Figure 12. The EXPRESS diagram of the main decomposition relationships of a facade.
Going upwards in the decomposition hierarchy we see that a facade is a part of the object called external wall structure, which in turn is a part of the structural system. Highest up we find the building object itself. This part of the product data model is important for positioning the data in the context of an overall building product model, but was not relevant for the testing performed in this project.

A facade may be a part of the external wall structure and a building frame. These in turn are parts of the structural system. In practice, a load bearing facade is considered as part of the building frame, a non-bearing facade belongs to the external wall structure. In this research the main focus was on the facade itself.

5.3 Structural layer

A structural layer may have openings, Figure 13. An opening is in most cases filled by a window, a door or both. A surface may have several configurations such as tiling and it has many attributes (properties) such as colour. Structural layers form the precast units of the facade, see Figure 12. A structural layer has certain material. The layer has surfaces on each side. Windows and doors may be attached to structural layers directly, as well as through openings (see opening).

Layer type is used for more accurate definition. Types include layers with a constant thickness, curved layers or layers where the thickness varies according to a mathematical function. The length, height and area of a layer may be derived from the edges. The reserved spaces (voids) are considered as to be filled with some equipment, usually HVAC equipment. It may be noted that an equipment may also fill an opening, see Figure 14. Also, the 2D-shape of a layer is derived from the edges.

The definition of a structural layer may be compared to other existing solutions. In the IDM model (Integrated Data Model) of the COMBINE project, layers form a construction type which in turn is an attribute to an element construction [Dubois et al. 1992]. In the French GSD (Groupe Structuration de Données) model, layers are defined as parts of composite enclosures. These in turn are a subtype of enclosure [GSD 1991]. A layer as proposed by Björk [1992] is a super-type of internal or external layers.
5.4 Opening

The information structure of an opening is shown in Figure 14. The opening is considered as passing through structural layers. The idea presented here is straightforward. It is also partly based on practical solutions and concepts adopted by designers. An opening is bounded by edges. Windows, doors and other equipment such as pipes are defined as optional attributes for the opening. Thus, an opening may also be empty which means that it may be interpreted as a hole.

Each layer may have an opening of its own. Thus, an opening of the whole element consists of individual openings for each layer. This approach is needed since the shape and size of an opening may vary in each individual layer.

In addition, an opening has an additional attribute in that it serves one or two spaces. This is not discussed here because spaces were not the main interest in this research.
Figure 14. The information structure for an opening.

A comparison to other models may be made. In the RATAS [RATAS-committee 1988], GSD [GSD 1991] and COMBINE’s IDM [Dubois et al. 1992] models the opening is a super-type of a door or a window. The IDM opening may contain a hole whereas the opening described in this research becomes a hole if the opening is empty.

5.5 Edge

Edges may be considered as special parts of a 3D-object. An edge is defined here as a line between two points. The edge shape is defined as a 2D-shape that is perpendicular to the trajectory between the two end points of the edge. The information structure for an edge is shown in Figure 15.

Edges are of two types: straight and curved. Edges could be defined in another way, i.e. edges are of one type which may have an optional property of radius of curvature. The developed software application uses a menu for selecting different edge shapes, see Figure 16.

Also, the basic shape of an edge, as shown in Figure 16, need not define the edge exactly. An edge may have a bevel in the practical structural software application solution. The practical solution for the edge shape is a select-type property where standard or user defined shapes are eligible. An edge may also have a surface. This surface may or may not be seen. For instance, a facade has a main surface that is seen from the outside. The surfaces of the edges may or may not be seen which in many cases depends on the joint width. The surface of an edge is usually considered a separate entity although it may be of the same material as the surface of the structural layer. The surface is explained in the next section.
Figure 15. The information structure for an edge.

An example of the interface for defining shapes of edges is shown in Figure 16. The user may select an appropriate edge shape by clicking on the desired shape.

Figure 16. Architect's edges-menu. Six different edge shapes in 2D are shown.

5.6 Surface

The information structure of a surface is shown in Figure 17. In practical applications the 2D-shape is needed to determine areas of certain surface types. A surface may have many material alternatives, it may have a surface finish and may also have different surface patterns. Tiling may be attached to a surface, see also Figure 20. A manufacturer may calculate costs in terms of areas of different surfaces. A surface finish is chosen from a list.

The definition of a surface is of importance since it in many cases determines exactly the work order in manufacturing.
One may compare the presented solution to other models. The RATAS surface [RATAS-committee 1988] is an attribute of a room. The surface is a supertype of three different surfaces which are floor, ceiling or wall surfaces. A surface is a continuous area on the same wall, where a uniform surface material has been applied. A surface finish may be applied to a surface.

In COMBINE's IDM model, a face is defined as a surface which may have a surface finish [Dubois et al. 1992]. It also defines sub-faces. On the element level an enclosure element's geometry has a 2D aspect which is an element surface.

5.7 Overall model

In Figure 18, the subschemas presented in Figures 13, 14, 15 and 17 have been integrated and are shown as one schema. For readability reasons some attribute definitions have been omitted from this figure. The full textual EXPRESS code of the precast concrete facade is shown in Appendix C.
Figure 18. The overall EXPRESS-G-diagram of the facade.
6. CHECK LISTS

6.1 Formulation of the check lists

Check lists are a practical solution that helps in defining the data requirements during the different design stages. In this study check lists were defined based on the activities in the process model (see Figure 9) and the entities and attributes of the product model (Figure 18).

The entities that are of importance in each design stage were selected from the product data model of the facade. The data accuracy and contents of an entity, such as a structural layer, changes during the design process. Thus, same entities are designed more than once but in different level of detail.

For instance, a manufacturer and an architect may agree that the data in the global design stage shall include the shape and position of the facade and the type of the element joints. The element division may be determined by the architect or by the structural engineer.

An example of a part of a check list is shown in Table 2. The example check list has two main entities that are the facade and the structural layer. A number of attributes to these are also shown in the list.

The global design stage will produce enough information for the application for the building permit. The data during the first stage in the detailed design stage will be used for the invitation of tenders. This stage may be called tender design. The manufacturer will calculate costs on the basis of these data. The invitation of tenders will be complemented by other parties involved, e.g., the structural engineer. After the selection of the manufacturer the detailed design will produce the initial data for the detailed element design. This stage may be called initial element design. The design is further elaborated by the structural engineer and it will finally produce all the necessary data for the manufacturing process.

The use of check lists results in a clear definition of the data generation and flow during the whole building design process. The data producer and the data user could also be shown in the lists.

Each activity outputs some data concerning the facade. To construction professionals the check lists are more readable than the product model or the activity model.

The areas of different surfaces and element types could also be added to the check lists. This is purely for practical reasons as manufacturers usually calculate costs based on areas.
Table 2. An extract from a check list.

<table>
<thead>
<tr>
<th>CHECK LIST</th>
<th>Global design</th>
<th>Tender design</th>
<th>Initial element design</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Facade</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>facade</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>position</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>element division</td>
<td>position</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>element joint</td>
<td>type</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>width</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>length</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2D-shape</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>position</td>
<td></td>
<td></td>
</tr>
<tr>
<td>joint material</td>
<td>name</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>colour</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Structural layer</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>structural layer</td>
<td>layer type</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>thickness</td>
<td></td>
<td></td>
</tr>
<tr>
<td>layer material</td>
<td>name</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>thermal property</td>
<td>u-value</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>layer edge</td>
<td>edge shape</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>corner points</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>bevel</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>position</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>edge surface</td>
<td>(as layer surface)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>layer surface</td>
<td>2D-shape</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>area</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>colour</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>position</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>surface material</td>
<td>name</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>colour</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>surface finish</td>
<td>type</td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>
6.2 Possible uses of the check lists

These checklists can be used in a number of ways:

• They can provide additional material that helps in determining subcontract boundaries;
• They can be used to educate project participants so that they understand the design management aspects better;
• They can be incorporated in quality insurance manuals; and
• They can be used as input to the software development projects.

The check-lists may be used as supplements to contracts. The same check list may be used to define data needs during different design stages as well as for defining limitations and subcontract boundaries. A typical example concerns the element division since it may be defined either by the architect or the structural engineer. The project participants may get a clearer view of the importance of the entities during the design process. The check lists may also be used in the quality insurance manuals, especially when new projects with new participants are started. Software developers may also profit from the check lists since the data management and information flow is clearly defined.

The check lists that were developed during this research were further included in a guideline issued by the Building Information Institute in Finland [RT 1994]. This guideline is focused on precast concrete structures. Thus, it contains information about wall structures, slabs, beams and columns. All design stages are included and examples of data exchange formats given.
7. DEVELOPING PROTOTYPE SOFTWARE

7.1 Implementation of type objects

A prototype computer application was developed to test the product data model. The software platform was AutoCAD. The reasons for this choice were that the architectural company involved in the research uses it and that it has been used for many prototype applications by the research group at VTT Building Technology.

An important data structure which was incorporated in the prototype is the use of so-called type objects to define common properties of one or more instance objects. This is an important aspect of several of the generic product model proposals that have been defined (GARM, OOCAD, IAI core). Instance objects (the actual individual elements of a facade) are usually defined as occurrences of type objects (template definitions of standard types of elements). Type objects may also be nested, that is consist of a number of child objects which are instances of other type objects [Serén et al. 1993].

The feature of AutoCAD which was used in the initial phase of the prototype development was the so-called block. Blocks can be used to implement type objects. A Block is a named assembly of graphical objects and attribute definitions. Graphical objects are lines, polylines, etc.. Attribute definitions contain alphanumeric information. Additionally, a Block may contain an instance of another block.

It turned out that it is difficult to modify blocks dynamically during the design procedure for which reason they were not used in later prototypes. The product model was consequently later implemented using AutoCAD's extended entity data feature [AutoCAD 1992].

The physical data exchange between the different applications was implemented as OXF files. OXF is an object-based data exchange format developed in a research project at VTT [Serén et al. 1993]. In scope, it is rather similar to the STEP physical file format [ISO 1994]. The main reason to use OXF was the availability of software supporting the generation of OXF files from AutoCAD as well as the technical features of the format. In particular OXF has been designed to support type and instance objects. A fuller explanation of OXF is, however, outside the scope of this paper.

The principle of the CAD-implementation is shown in Figure 19. The figure shows how type and instance objects were implemented in AutoCAD and in the neutral data exchange file.
Figure 19. Principle of the CAD-implementation illustrated by mapping between AutoCAD-entities and the neutral data exchange file [Serén et al. 1993].

7.2 Functions of the prototype

The programming of the new functions to the prototype was made using AutoCAD’s AutoLISP programming language. AutoLISP is a dialect of the LISP (LISt Processing) programming language. The main functions of the prototype were:

- define or modify an attribute set;
- display an attribute set on the screen;
- insert a type object;
- display a type object on the screen;
- edit an instance of a type object, regardless of that the instance may contain instances of other type objects;
• copy instance objects;
• create or show relationships between instances;
• export data to a neutral exchange file; and
• import data from a neutral exchange file.

The prototype was used to design a facade that consists of a few elements. Two designs were made: one for the whole facade, one for a single element. After these a neutral data exchange file was created in OXF-format.

An example of how the data were stored in the extended entity data of polylines is shown in Figure 20. A structural layer and a surface with tiling are shown in the figure. For instance, the tiling is stored in a polyline called TILING. The polyline has attribute sets named STRUCTURAL_LAYER and SURFACES. The DATA is identified as an area that is denoted by A. Other entities are treated the same.

Figure 20. The dimensional data of an element. The data are stored in AutoCAD's extended entity data.
7.3 The final prototype

The final prototype application that was used in a test project (see next chapter) was developed in the architectural company. This application also uses AutoCAD’s extended entity data for storing the information. The architect’s AutoCAD was version 12.

The final prototype application has a simple interface that enables a user to create an id for an element and attributes that are needed. The functions that were included in this application were:

- design of an element;
- design of an area;
- design of an edge and joints; and
- output to a file.

The prototypes take into account the architect’s normal way of design. This means that the produced data are more accurate and detailed in the later design stages (from the global design stage to the detail design stage). Also, as mentioned earlier, the architectural data focuses more on surfaces and openings, not structural details.
8. TESTING THE DEVELOPED METHODS IN PRACTICE

8.1 Purpose of the testing
The prototypes were tested with data from a real design project, in collaboration with the firms that participated in the research. The purpose was to test:

- the usefulness of having architectural design and element design data available in a data base format, rather than as ordinary drawings, during the detailed element design and manufacturing;
- if the particular product data model presented above in Chapter 5 has the functionality required (in terms of data content and data accuracy) for supporting the digital exchange of data between the project participants; and
- the effects of using a product model approach on the management of design modification data during the design and manufacturing process.

8.2 Test building
Testing was carried out as a dry run using information about of a real, recently designed and constructed residential building. The building which was used as a case is a multi-storey apartment house in a suburb of Helsinki (Figure 21), and is representative of the precast concrete technology currently used for residential buildings in Finland. The total area of the building is 7339 m² and it includes 95 apartments.

![Figure 21. The test building used in this research.](image)

This particular building was chosen because its design and construction timetable suited to the schedule of the research project and because the structural designer and the element manufacturer had been involved in the project. Consequently it was possible, at least subjectively, to estimate the
effects of using the product model approach, compared to the normal data exchange practice which had been used in the project. In the test, a part of the eastern facade of the building (storeys 13-15) was redesigned and the data were exchanged between the project participants as in the real project.

### 8.3 Software

In addition to the prototypes that were developed as a part of this study commercial applications had to be used in order to complete the whole design and manufacturing ”circle”. The resources available for the research were not sufficient for the development of prototypes that would have covered all the necessary tasks.

The software tools that were used in the testing are listed below:

The architectural designer used:
- Prototypes developed in the study on top of AutoCAD (version 12).

The structural or element designer used the following modules of ConcreteCAD:
- Gides (whole building, whole facades);
- Exwall (detailed element design).

The element manufacturer used:
- ConcreteCAD;
- In-house CAM-software of Partek.

ConcreteCAD is suite of software modules which have been tailor-made for the design and manufacturing of prefabricated concrete buildings [ConcreteCAD 1992]. It is widely used in Finland and has users also in several other countries. The structure of ConcreteCAD is centred on having one and only one model of the building being designed, which is stored in a customised database. Drawings and other output documents are generated from the information in the database. ConcreteCAD was also chosen because the industry practitioners who participated in the testing were already using it.

The manufacturer has a modern facility using computer controlled adjustable element moulds. The software used by the element manufacturer for production control has a relational data base as its platform. In addition the factory uses Concrete CAD for some in-house design applications.

### 8.4 Prototype software test

Some initial small-scale tests of the developed AutoCAD prototypes were carried out before the actual design started. Special interest was focused on how data from the architectural design stage may be directly used in the element design. The architectural data that are defined and may be used are
modular dimensions, surfaces, openings and edge shapes (see also Figure 9). In addition, the architectural data consists of details concerning window clamps and fasteners. A number of corrections were made based on the feedback received. The major points from the feedback were:

• The edges of elements were not stored in the architect’s model as it contained only surfaces and shapes of edges.

• The local element co-ordination in the ConcreteCAD program is calculated from the position of the first edge that is input to the database. This caused that an individual element had an arbitrary position when it was input to the ConcreteCAD program.

• Some openings were not defined, they were just empty areas between surfaces.

• Some edge shapes were missing due to a bug in the architect’s program.

• The DXF conversion did not match exactly.

The following conventions were adopted to ensure better co-ordination:

• The origin of the local co-ordinate system of an element will be placed on the lower left corner of the element (seen from the outside of the building). The first edge shall start from the origin.

• All openings shall be designed as real openings because a surface may consist of parts that form the whole surface. The initial tests showed that the architect's "opening" was just an area with no surface.

• The edges of an element shall be designed counter-clockwise.

• An element shall have an attribute which specifies the location of the element in the global co-ordinate system.

8.5 Test design

After the modifications the actual test design was started. The testing was divided into two main phases:

• phase A included a normal design procedure; and

• phase B included a change in the design solution.

In the phase A the eastern facade of the building for storeys 13-15 was designed. This included the design of the elements by the architect. The architectural designs were used as input to the structural (element) design. The architectural design was started by generating modular dimensions, surfaces, thickness of structural layers and materials. The openings were designed next. The shapes of edges were designed for each individual structural layer including shapes in the openings.
In the phase B the architect made two changes: an element was redimensioned by 300 mm (width) and the position of one opening was changed. These changes resulted in different proportions of surface material. The changes are critical from the manufacturer's point of view.

The main interest in each phase, A and B, was focused partly on the reuse of data from the architect's application by the structural engineer's application, and partly on the exchange of data from the structural engineer's application to the manufacturer's system. In addition, possible errors and lack of data were studied.

The testing was carried out in accordance with the activity model which had been developed in the project. The test design "stage" corresponds to the detail design stage in the activity model (see Figure 8 and Figure 9).

The architect used the application which had been developed in the project. This application creates a data base using the conceptual schema presented above, although this "database" lacks some of the functionality of object-oriented or relational databases, since it has been implemented as part of a sequential CAD file using AutoCAD’s extended entity data. The architect designed the surfaces, openings and some window details. Surfaces consisted of many sub-surfaces. The types of edges, i.e., edge shapes, varied in the openings. The types were chosen using a menu, see Figure 16. Finally a data exchange file was produced in ASCII format.

In the next stage these data were transferred to the structural engineer’s ConcreteCAD application. Although it would have been possible to write software that could have done this conversion automatically, the conversion was done manually, by interpretation of the human-readable files produced from the architect’s application. The reasons for this were purely due to resource restrictions, since developing such conversion software would have cost too much, in comparison to the overall budget of the project. The data exchange was thus only simulated. The important thing is, however, that the data contained in the architect’s model was transferred to the receiving application, not the speed of the conversion.

After this the element designer started his work using ConcreteCAD's Gides-module. The element division was chosen, based on the modular dimensions of the architectural design. The thickness and material types were designed for each structural layer. Elements were created according to the element division. The element division is decided either by the architect or the structural engineer, depending on what has been agreed. In this example, the architect decided the element division.

The edges were then defined. It showed that the architect had designed edges of the outer wall layer using modular dimensions, for which reason they had to be moved inward by 7.5 mm. The insulation layer and the inner structural layers were also treated in the same way. The openings were
designed using the same principle. The edges of openings had to be moved by a few millimetres because the architectural data used modular dimensions.

The rest of the structural design was accomplished as a normal design. Reinforcement and other equipment were defined. For further elaboration of the data, the following information for each element was saved in the database of the structural designer:

- ID of the element (precast unit);
- name of each layer, dimensions and relative position;
- surfaces and their materials;
- size and relative position of opening;
- shape of edge of each layer;
- reinforcement rebars of each layer;
- shapes, size and relative position of reserved spaces;
- name, shape, size and relative position of additional equipment.

The position of each structural layer was relative to the position of the element. Elements were positioned in the global co-ordinate system of the facade.

Element joints may cause problems: module dimension versus real dimension. The architect may design joints exactly by shapes and dimensions but uses modular dimensions on positioning on the element level. Another problem may occur: if the edge of an opening coincides with the edge of a structural layer, i.e., the opening may not be considered as an opening anymore.

In the last stage, the structural engineer passed the designs to the manufacturer electronically using an Ethernet network. Since the manufacturer also used ConcreteCAD up-front no conversion was needed. A macro in ConcreteCAD's Exwall module created an ASCII-file in which the manufacturer added some internal parameters such as product type number and the name of the "lowest" layer during casting. This ASCII file was then used as input information for the production planning software of the manufacturer.

In addition a material table in BEC-format [BEC 1991], [Hannus 1990] was created from the data transferred to the manufacturer. BEC was developed in Finland during the 1980s. The whole BEC-system includes a number of software modules for the receiving and manipulation of graphical BEC-files and for the management of tables. The BEC formats are currently widely used for data exchange to element manufacturers in Finland.
8.6 Results

The results of the tests were both quantitative and qualitative. The participating designers were asked to compare the time used when using the product model approach to the way the work had originally been carried out. The results were conclusive to some extent. The total amount of time spent was 20 % longer using the product model approach in phase A (normal design procedure). The design changes were 7 % longer. On the other hand this was largely due to the substantially increased workload of the element designer because some data were manually converted from the architect’s application to the element designer’s application. (The data existed in the architect’s designs but there was no conversion program.)

Both the architect and the manufacturer saved time. Compared to the drawing oriented approach the manufacturer may make changes approximately 30 % faster in the product model approach because the data of an element is in computer interpretable format.

More important were the qualitative advantages which the participants testified to in the interviews carried out. The conceptual schema of the facade proved to be sufficient. This led to a better data accuracy and the data content was clearly defined already at an early stage of the building design process. The test showed that the architectural data are usable as such in the element design which means that the normal procedures of architectural design need not be changed. Data in earlier building design stages are considered more general, i.e., facade is designed as a whole where surfaces and opening are placed.

The test also showed that having usable architectural data the element designer may concentrate on essential design matters. Especially, the shapes of edges as defined by the architect reduced the routine design work of the element designer. It is of importance to note that the shapes of edges depend on the manufacturer.

The manufacturer has an advantage: cost estimations may be done earlier than normally. This will also enable more data interchange between the architect and the manufacturer because solutions may be based on the manufacturer’s suggestions. The automation level of the manufacturing process may be improved only through more accurate data.

In the long run such qualitative advantages will no doubt be converted into monetary benefits due to shorted design times, fewer mistakes, etc.

8.7 Limitations and restrictions of the model

Limitations of the developed model exist. For instance, a window or a door may be included in an opening which also means that an opening is always surrounded by material of the structural layer. The opening is inside the
layer. But in some cases the opening is not inside of the layer. The problem is illustrated in Figure 22.

The edge of the opening coincides with the edge of the structural layer. The opening is no more a normal opening but it becomes an opening in a facade. It also means that the window or a door is no more in an opening of a layer but in an opening of the facade. The presented model is limited to openings in the structural layers only. This type of structural layers are common where balconies are attached, or generally where an entrance is needed. These “openings” are usually filled by a combination of a window and a door.

Figure 22. The problems of an opening: a normal opening on the left is inside of the structural layer, the opening on the right is not inside of the layer.
9. CONCLUSIONS

9.1 Usefulness of the results

The architectural description of a facade is complicated. The test verified that the data defined by the architect such as modular dimensions, surfaces, openings and edge shapes, are usable directly in the structural design. Thus, the data are also usable for the manufacturer of the precast facades. It also showed that it is possible to produce the data in computer interpretable format without changing the "normal" design procedure. However, this requires well-designed computer applications. The product model based approach forces more accurate data definitions which in the long run results in cost reduction in terms of error and redundant design work reduction.

The SADT method was well suited for the overall description of the activities that occur during the traditional building process. The integration of the activity model with the conceptual model is of importance. Also, the influence of using the product model approach on the process itself is of interest, especially when feedback and experience from past designs and solutions are to be used.

The basic results obtained in this project may be exploited in all stages of the process of designing and manufacturing prefabricated elements even if the test was focused on the detail design stage. Architects and other participants in a design project may use the check-lists to define their data needs during all stages of the process.

Developers of software applications may utilise the object definition cards and the conceptual schema of a precast facade to develop new software applications. Also, manufacturers may standardise their data needs. A number of requirements need to be defined for the software applications. An application shall be able to handle:

- surfaces (material and area);
- openings (dimension and area);
- edge shapes of openings and structural layers;
- the facade as a whole;
- element joints; and
- the co-ordinates of a structural layer.

The coordinate systems for each object should also be standardised or agreed at the beginning of each project. Furthermore, applications shall not restrict the design procedure of architect.
9.2 A possible way ahead - STEP and IFC

Extensive exploitation of the ideas and results presented here requires standardising the data exchange structures on the national or preferably international level. ISO/STEP development [ISO 1994b] could be a solution in the form of a possible application protocol for prefabricated facade elements. Also, IAI (International Alliance for Interoperability) and its IFCs (Industry Foundation Classes) may provide solutions [IAI 1996] in the near future. It is to be hoped that the results of this research could provide a useful input into these standardisation activities, in a sub-domain which is of particular importance to the Finnish construction industry.
ACKNOWLEDGEMENTS

This research was funded by the Technical Research Centre of Finland (VTT), The Finnish Association of Construction Products Industries (RTT) and the Technology Development Centre of Finland (TEKES). The companies that participated in the testing were the architectural company Arkkitehtitoimisto Innovarch Oy, the structural engineering company Tampereen Juva Oy and the manufacturer Partek Betonila Oy. The architect’s final software was developed by Studio Kivi Oy.

The following persons were interviewed or participated in the research: Christer Finne (Arkitehtiryhmä Kråkström Oy), Seppo Niemioja and Kimmo Setkänen (Arkkiitehtitoimisto Innovarch Oy), Pertti Alho and Tapio Ristimäki (CADEX Oy), Aarni Heikonen (CADOR Oy), Aarno Myllyniemi and Antti Laitakari (Finnmap Oy), Harto Räty (Finnmap Partners Oy), Heikki Järvinen (Juva Engineering Oy Ltd), Jouni Allen-Perkko, Pentti Jalonen and Timo Kannisto (Parma Oy), Heikki Sarín, Kalevi Granlund, Matti Raukola, Timo Tuominen and Tuomo Svanlund (Partek Betonila Oy), Jari Tanskanen (Partek Parametri Oy), Riitta Takanen (Rakennustoimisto A. Puolimatka Oy, Arto Suikka and Seppo Petrow (RTT), Matti Rasilainen and Pekka Timonen (Runktetek Oy), Arto Kiviniemi (Studio Kivi Oy), Aarno Vainio (Tampereen Juva Oy), Olli-Pekka Nordlund (TEKES) and Asko Sarja (VTT Building Technology).
REFERENCES


APPENDIX A: SADT/IDEF0 syntax

The activity modelling method SADT (Structured Analysis Design Technique) and its subset IDEF0 (Integration Definition for Function Modelling) are used to produce a structured, graphical representation of some process.

The activity model is in many cases presented from a chosen point of view, for instance, from the architectural design point of view. The activity models are independent of time which means that the activities may be presented in any order.

The activities are represented by a box as shown in Figure 23. The information or objects used are represented by arrows. These arrows are called input, output, control and mechanism.

- The input enters the box from the left. Example: raw data.
- The output is represented with an output arrow. Example: drawing.
- The control enters the box from the top. Example: design instructions.
- The mechanism enters the from the bottom. Example: computer program.

The arrows are needed by or produced by the activity. It should be noted that an activity cannot be done unless all the necessary arrows are present.

![Figure 23. The SADT/IDEF0 activity box.](image)

The distinction of the input arrows and controls is that the input arrows represent the information which are processed by the activity while the control arrows describe conditions or specifications that govern the activity. The output arrows are the information produced by the activity, the results.

The activities are drawn in hierarchical diagrams. A schematic view of the hierarchy of the diagrams is shown in Figure 24. An activity may be presented in more detail on a lower diagram. Thus, each diagram on a lower
level is a child diagram of the upper level diagram, the parent diagram. The parent diagram is a sum of the child diagrams.

The hierarchy of the diagrams may be traced by the node number. The node number is shown on the lower left corner of the diagram. The hierarchy may also be traced by the graphical context box that is shown on the upper right corner of the diagram.

Figure 24. The hierarchical structure and reference systematic of the IDEF₀ modelling system.
APPENDIX B: EXPRESS language

The EXPRESS language and its graphical counterpart EXPRESS-G are based on the language elements listed in Table 3.

Table 3. EXPRESS language elements.

<table>
<thead>
<tr>
<th>Schema</th>
<th>The schema defines a common scope for a collection of related entity and other data type declarations.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entity</td>
<td>A class of information defined by common properties</td>
</tr>
<tr>
<td>Type</td>
<td>System- or user-defined domain. Examples of types are INTEGER, REAL, NUMBER, STRING, BOOLEAN, LOGICAL, ARRAY, BAG, LIST, SET, or user defined</td>
</tr>
<tr>
<td>Attribute</td>
<td>Entity characteristics which may explicit, derived or inverse. An example of a derived attribute could be the area.</td>
</tr>
<tr>
<td>Rule</td>
<td>Constraints for attributes or entity instances. These may be local or global rules.</td>
</tr>
<tr>
<td>Function</td>
<td>Procedural algorithms for the calculation of attributes or for the evaluation of rules.</td>
</tr>
<tr>
<td>Procedure</td>
<td>Procedural algorithms for the calculation of attributes or for the evaluation of rules.</td>
</tr>
</tbody>
</table>

The entities may also be supertypes or subtypes of another entity.

An entity named facade is defined in the following example. The entity facade is a supertype of either a precast facade or a cast in situ facade. It can also be both. The facade has a number of attributes, for instance the position and the shape.

ENTITY facade
SUPERTYPE OF (precast_facade ANDOR cast_in_situ_facade);
position : position;
id : id;
shape : shape;
balconies : OPTIONAL SET [1:?] OF balcony;
channels : OPTIONAL SET [1:?] OF channel;
END_ENTITY;

The relations may be optional. A subtype of the facade is a precast facade as defined in the next example.

ENTITY precast_facade
SUBTYPE OF (facade);
prefcast_elements : SET [1:?] OF precast_element;
element_division : element_division;
element_joints : SET [1:?] OF element_joint;
END_ENTITY;

The complete schema for a facade in textual format is presented in Appendix C.
The graphical representation of an entity is a box, Figure 6. Relationships are shown with normal lines, optional attributes as dashed lines. Thick lines are used to represent an inheritance relationship, i.e., a subtype and a supertype relationship.

Figure 25. The graphical representation of an entity and its relations.
APPENDIX C: Schema of the facade in EXPRESS

SCHEMA Design_facade;

ENTITY acid_treatment;
END_ENTITY;

ENTITY acoustic_property;
END_ENTITY;

ENTITY balcony;
  position : position;
  id : id;
  relations : SET [0:?] OF relation_type;
END_ENTITY;

ENTITY basement;
END_ENTITY;

ENTITY beam;
END_ENTITY;

ENTITY bearing_wall;
END_ENTITY;

ENTITY bevel;
END_ENTITY;

ENTITY biological_property;
END_ENTITY;

ENTITY bond;
  bond_type : STRING;
  starting_point : LIST [1:?] OF point;
END_ENTITY;

ENTITY brick_tile
SUBTYPE OF (surface_tile);
END_ENTITY;

ENTITY brush_treatment;
END_ENTITY;

ENTITY building;
  structural_systems : SET [1:?] OF structural_system;
  technical_systems : SET [1:?] OF technical_system;
  id : id;
  position : position;
  shape : shape;
  address : STRING;
  number_of_storeys : INTEGER;
  area_of_storey : REAL;
  volume : REAL;
  purpose_of_usage : STRING;
  quality_level : STRING;
  total_costs : REAL;
END_ENTITY;
fire_class : STRING;
END_ENTITY;

ENTITY building_frame;
columns : SET [0:?] OF column;
beams : SET [0:?] OF beam;
slabs : SET [0:?] OF slab;
bearing_walls : SET [0:?] OF bearing_wall;
facades : LIST [1:?] OF facade;
END_ENTITY;

ENTITY cast_in_situ_facade
SUBTYPE OF (facade);
structural_layers : LIST [1:?] OF structural_layer;
END_ENTITY;

ENTITY channel;
END_ENTITY;

ENTITY clamp;
END_ENTITY;

ENTITY cleansing;
END_ENTITY;

ENTITY column;
END_ENTITY;

ENTITY curved_edge
SUBTYPE OF (edge);
radius_of_curvature : REAL;
END_ENTITY;

ENTITY curved_layer;
END_ENTITY;

ENTITY door;
d_type : STRING;
fire_class : STRING;
frame : frame;
position : position;
id : id;
END_ENTITY;

ENTITY durability_property;
END_ENTITY;

ENTITY edge
SUPERTYPE OF (ONEOF(straight_edge, curved_edge));
corner_points : LIST [2:2] OF point;
surfaces : OPTIONAL SET [1:?] OF surface;
edge_shape : edge_shape;
bevels : OPTIONAL LIST [1:?] OF bevel;
position : position;
id : id;
END_ENTITY;

ENTITY electrical_property;
ENTITY element_division;
END_ENTITY;

ENTITY element_joint;
ej_type : joint_type;
shape : OPTIONAL shape2D;
joint_material : SET [1:?] OF material;
size : size;
position : position;
id : id;
relations : SET [0:?] OF relation_type;
END_ENTITY;

ENTITY even_layer;
thickness : REAL;
END_ENTITY;

ENTITY external_wall_structure;
facades : LIST [1:?] OF facade;
END_ENTITY;

ENTITY facade
SUPERTYPE OF (precast_facade ANDOR cast_in_situ_facade);
position : position;
id : id;
shape : shape;
balconies : OPTIONAL SET [1:?] OF balcony;
channels : OPTIONAL SET [1:?] OF channel;
END_ENTITY;

ENTITY fire_property;
END_ENTITY;

ENTITY frame;
frame_depth : REAL;
module_dimensions : LIST [1:?] OF REAL;
position : position;
id : id;
END_ENTITY;

ENTITY functional_property;
END_ENTITY;

ENTITY grinding;
END_ENTITY;

ENTITY hard_burned_brick
SUBTYPE OF (surface_tile);
END_ENTITY;

ENTITY hewing;
END_ENTITY;

ENTITY hidden_joint;
END_ENTITY;
ENTITY id ;
END_ENTITY ;

ENTITY joint ;
  j_type : STRING ;
  shape : OPTIONAL shape2D ;
  materials : SET [1:?] OF material ;
  size : size ;
  position : position ;
  id : id ;
END_ENTITY ;

ENTITY material ;
  name : STRING ;
  colour : OPTIONAL STRING ;
  manufacturer : OPTIONAL STRING ;
  product_name : OPTIONAL STRING ;
  material_properties : OPTIONAL SET [1:?] OF material_property ;
  acoustic_properties : OPTIONAL SET [1:?] OF acoustic_property ;
  biological_properties : OPTIONAL SET [1:?] OF biological_property ;
  durability_properties : OPTIONAL SET [1:?] OF durability_property ;
  thermal_properties : OPTIONAL SET [1:?] OF thermal_property ;
  optical_properties : OPTIONAL SET [1:?] OF optical_property ;
  fire_properties : OPTIONAL SET [1:?] OF fire_property ;
  structural_properties : OPTIONAL SET [1:?] OF structural_property ;
  electrical_properties : OPTIONAL SET [1:?] OF electrical_property ;
  functional_properties : OPTIONAL SET [1:?] OF functional_property ;
END_ENTITY ;

ENTITY material_property ;
END_ENTITY ;

ENTITY mathematical_layer ;
END_ENTITY ;

ENTITY mosaic_tile
  SUBTYPE OF (surface_tile);
END_ENTITY ;

ENTITY mould ;
END_ENTITY ;

ENTITY natural_stone_tile
  SUBTYPE OF (surface_tile);
END_ENTITY ;

ENTITY opening ;
  edges : LIST [2:?] OF edge ;
doors : OPTIONAL SET [1:?] OF door ;
windows : OPTIONAL SET [1:?] OF window ;
other_equips : OPTIONAL SET [1:?] OF other_equipment ;
position : position ;
id : id ;
DERIVE
  o_area : REAL ;
END_ENTITY ;

ENTITY optical_property ;
END_ENTITY ;

ENTITY other_edge_shape ;
END_ENTITY ;

ENTITY other_equipment ;
END_ENTITY ;

ENTITY other_tile
  SUBTYPE OF (surface_tile) ;
END_ENTITY ;

ENTITY painting ;
END_ENTITY ;

ENTITY parging ;
END_ENTITY ;

ENTITY point ;
END_ENTITY ;

ENTITY position ;
END_ENTITY ;

ENTITY posture_fixing ;
END_ENTITY ;

ENTITY precast_element ;
  structural_layers : LIST [1:?] OF structural_layer ;
  element_type : STRING ;
  fireclass : OPTIONAL STRING ;
  manufacturing_series_number : OPTIONAL STRING ;
  element_series_number : OPTIONAL STRING ;
  position : position ;
id : id ;
relations : SET [0:?] OF relation_type ;
DERIVE
  pe_length : REAL ;
  pe_height : REAL ;
  pe_area : REAL := pe_length*pe_height ;
END_ENTITY ;

ENTITY precast_facade
  SUBTYPE OF (facade) ;
  precast_elements : SET [1:?] OF precast_element ;
  element_division : element_division ;
  element_joints : SET [1:?] OF element_joint ;
ENTITY structural_layer;
  material : SET [1:?] OF material;
  position : position;
  id : id;
  surfaces : SET [1:?] OF surface;
  openings : OPTIONAL SET [1:?] OF opening;
  reserved_spaces : OPTIONAL SET [1:?] OF reserved_space;
  doors : OPTIONAL SET [1:?] OF door;
  windows : OPTIONAL SET [1:?] OF window;
  edges : LIST [2:?] OF edge;
  clamps : OPTIONAL SET [1:?] OF clamp;
  otherEquipments : OPTIONAL SET [1:?] OF other_equipment;
  type_of_layer : layer_type;
  layer_name : STRING;
  shape : shape2D;
DERIVE
  sl_length : REAL;
  sl_height : REAL;
  sl_area : REAL := sl_length * sl_height;
ENTITY structural_property;
END_ENTITY;

ENTITY structural_system;
  basement : basement;
  external_wall_structures : SET [1:?] OF external_wall_structure;
  building_frame : building_frame;
END_ENTITY;

ENTITY surface;
  shape : shape2D;
  colour : OPTIONAL STRING;
  coarseness : OPTIONAL STRING;
  surface_kode : OPTIONAL STRING;
  materials : SET [1:?] OF material;
  surface_finishes : OPTIONAL LIST [1:?] OF surface_finish;
  surface_pattern : OPTIONAL SET [1:?] OF surface_pattern;
  part_surfaces : OPTIONAL SET [1:?] OF surface;
  surface_tilings : OPTIONAL SET [1:?] OF tiling;
  position : position;
  id : id;
  DERIVE
  s_area : REAL;
END_ENTITY;

ENTITY surface_pattern;
END_ENTITY;

ENTITY surface_tile
  SUPERTYPE OF (ONEOF(natural_stone_tile, hard_burned_brick,
                         mosaic_tile, brick_tile, other_tile));
  size : size;
  colour : STRING;
  manufacturer : OPTIONAL STRING;
  product_name : OPTIONAL STRING;
  quality_class : OPTIONAL STRING;
  fixing_system : fixing_method;
  position : position;
  id : id;
END_ENTITY;

ENTITY technical_system;
  relations : SET [0:?] OF relation_type;
END_ENTITY;

ENTITY thermal_property;
END_ENTITY;

ENTITY tiling;
  joints : SET [1:?] OF joint;
  bond : bond;
  surface_tiles : SET [1:?] OF surface_tile;
  position : position;
  id : id;
END_ENTITY;
ENTITY visible_joint ;
END_ENTITY ;

ENTITY window ;
  w_type : STRING ;
  frame : frame ;
  window_ledge_plate : OPTIONAL window_ledge_plate ;
  glasses : LIST [1:?] OF window_glass ;
  position : position ;
  id : id ;
DERIVE
  number_of_glasses : REAL ;
END_ENTITY ;

ENTITY window_glass ;
  shape : shape2D ;
  materials : SET [1:?] OF material ;
  thickness : REAL ;
  position : position ;
  id : id ;
END_ENTITY ;

ENTITY window_ledge_plate ;
END_ENTITY ;

TYPE
  joint_type = SELECT (hidden_joint, visible_joint) ;
END_TYPE ;

TYPE edge_shape = SELECT (standard_edge_shape, other_edge_shape) ;
END_TYPE ;

TYPE fixing_method = SELECT (mould, posture_fixing) ;
END_TYPE ;

TYPE layer_type = SELECT (even_layer, skewed_layer, curved_layer, mathematical_layer) ;
END_TYPE ;

TYPE surface_finish = SELECT (hewing, acid_treatment, brush_treatment, sand_blasting, grinding, sanding, painting, cleansing, parging, roller_application) ;
END_TYPE ;

END_SCHEMA ;