PRAGMATIC IMPLEMENTATION OF AN INTEGRATED BUILDING DESIGN SYSTEM

Carsten Rode and Karl Grau

ABSTRACT: A set of simple procedures has been developed to facilitate data exchange between applications for building design and analysis using the ISO STEP standard. A system architecture has been devised that uses these procedures to bind small building analysis models together to fulfill a complete design task. A system of small program modules is currently under development which, in the first place, will function as a replacement for a conventional, stand-alone transient thermal analysis program. The system is based upon a common data model for a building, which is actually the union of partial data models for the small program modules involved. New submodels will be added in a later stage so the scope of the system becomes broader than thermal analysis alone. The anticipated end result is a pragmatic version of an integrated building design system that can easily be adapted in today's building design practices.

KEYWORDS: STEP, integration, pragmatism

1. INTRODUCTION

One of the main obstacles for practitioner's use of advanced computer models for building analysis, such as analysis of thermal or daylight conditions is the work it requires to process data for input to the models. However, many such data used by different models are similar and therefore should be shared between the applications.

The European COMBINE projects (I and II, 1990 - 95) were among the first international projects with relevance for the building sector to adopt the then emerging ISO STEP standard for exchange of product model data. The COMBINE projects were supported by the EU JOULE program for non-nuclear energy (Augenbroe, 1993 and 1995). The scope of the projects was to develop prototypical demonstrations of a so-called Integrated Building Design System (IBDS) where data integration between architectural and engineering design tools facilitates a more intense, efficient, and fail proof use of the programs. Once adopted by several types of building design offices, such systems should provide easier communication of design information between the actors in the design process. At an equal or lower design cost, this should result in buildings which are both more energy efficient and have more pleasant indoor environments.

The COMBINE projects succeeded in developing and demonstrating the functionality of an integrated building design system. However, the project also made it clear that such comprehensive systems face a problem with complexity which has an influence on the required effort for development work as well as on the efficiency of executing the final system. Besides requiring plenty of state of the art software and hardware technology to run the system, it also requires quite some training of the users and of

future developers to use and manage the system. As a result, despite of the technical virtues of the deliverable, it would not be immediately presentable as a working tool in today's design practices.

One of the conclusions from the COMBINE projects was an identified need for methods for rapid prototyping of integrated building design models even if such rapid developments would be more limited in scope (van Nederveen, 1995).

All in all, there is no doubt that the track pursued in the COMBINE project has been the right one - the deliverables just need some further development before they can be adopted by building design practices. Therefore, the European Commission was applied for a new project *WIND-UP*, Workplace Introduction of Integrated Design Systems - Utilisation in Practice, but this application has not been successful so far. In the meantime other attempts to get similar, although simpler technology developed at a more practical level have been ongoing by a number of former COMBINE partners. In due time, this will hopefully show to be one of the most important results of the COMBINE projects - the follow-up research initiated among a substantial number of European research institutions and the consensus among these institutions on the basic concepts. This paper describes one such follow-up activity - so far restricted to developments going on in Denmark, but one of the intentions with this paper is to invite interested co-developers to join.

The Danish Building Research Institute (SBI) participated in the COMBINE projects as a supplier of the institute's *tsbi3*-program as one of the design tools that were made available within COMBINE's integrated design system. *tsbi3* is a program for dynamic building thermal analysis (Johnsen and Grau, 1993), and besides using and testing COMBINE's procedures for interfacing such a program with the IBDS, SBI also gave input to the formulation of the project's Integrated Data Models.

The stand-alone *tsbi3*-program, has about 200 users among building designers, public organizations, and educational and research institutions in Denmark and Northern Europe. New extensions have been planned for the program as well as have new couplings to related analyses, e.g. for assessment of environmental impact of buildings. However, rather than extending the program in its current monolithic (stand-alone) form, it has been desired to decompose the program into smaller modules which communicate semantically meaningful data with one another as well as with new models. The resulting new program system should be easy to accept and implement in today's design practices. Furthermore, the system should be flexible and extensible.

These intentions, together with the experience from the COMBINE project has led SBI to develop new methods for plain STEP data exchange which can easily be implemented in existing or new C++ programs. The institute is now progressing in the development of a new suite of small computer models whose combination extends from the functionality of the *tsbi3*-program and which will be able to carry out STEP data exchange with other computer models by using the same data about a building, e.g. models for daylighting, indoor air quality, life cycle assessment, etc. The following sections of this paper describes the architecture of the system SBI is creating and the methods developed to facilitate STEP data exchange.

The evolution of SBI's program development from the stand-alone *tsbi3*-program, over participation in COMBINE to the current pragmatic way of implementing STEP communicating building design tools is illustrated in Figure 1. To give the pragmatic data exchange system a provisional name, the acronym DIS has been chosen, which stands for *Danish Integrated System*. A national project to utilise the Danish system in practice has already been secured as indicated in the figure with the acronym "UP" (for Utilization in Practice).

As can be seen from Figure 1, one of the ways to differentiate the different ambition levels is the level of STEP data sharing. Traditionally, STEP operates with four such levels: (1) File exchange, (2) Working form exchange (in memory), (3) Exchange through shared databases, and (4) Exchange through knowledgebases. While the COMBINE 1 deliverables used file exchange (level 1), and COMBINE 2 used a shared database (level 3), SBI is in the current phase targeting a working system that uses STEP levels 1 and 2 for the data exchange, i.e. a combination of data exchange using file transfer and data sharing in memory. A reason for this choice is the overhead the implementation of routines for support of STEP/SDAI data access (STEP Part 22) seem to have caused in COMBINE 2.

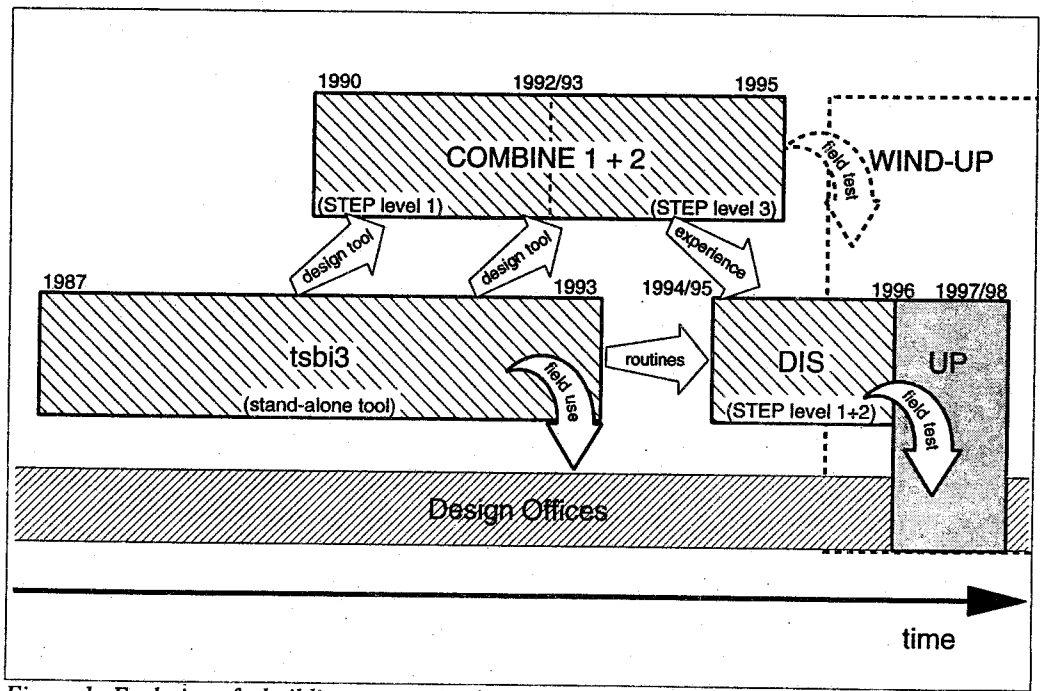

Figure 1. Evolution of a building energy simulation tool from a stand-alone program (tsbi3), over its role as a design tool in an advanced integrated building design environment (COMBINE 1+2), to its new, extensible form with pragmatic facilities for STEP data exchange (DIS).

2. A PRAGMATIC APPROACH TO AN INTEGRATED SYSTEM

For the reasons mentioned in the introduction, SBI has decided to develop a more pragmatic version of a system to facilitate data exchange between various building analysis models. The rest of this paper describes SBI's current developments, including some already developed procedures to facilitate

exchange of product data between applications using STEP. The idea is that a number of computer tools (or *applications*) for building design and analysis shall be able to share information about the buildings they analyze. A configured set of such applications will be called an *Integrated Building Design Environment (IBDE).*

SBI's Integrated Building Design Environment supports two ways in which data can be shared between applications:

1. One way is by exchange of so-called physical STEP files with actual building data according to a common building data model.
2. The other way is by sharing instances of the data model's entities in the computer's memory. Different applications will be implemented as so-called *Dynamic Link Libraries* (DLLs) for MS-Windows which can be activated by the system as necessary. While exchanging data in this mode, the applications are also able to read or write physical STEP files.

In either case, the applications exchange data through a common conceptual data model. Each application has its own data model, and the common data model is simply the combination of all active submodels. Potentially, there could be a problem with overlaps between the definition of entities, which is handled by letting each entity belong to only one submodel. Other uses of this entity must include and refer to the submodel where it is originally defined.

3. APPLICATIONS IN THE INTEGRATED BUILDING DESIGN ENVIRONMENT

One reason to implement a computer program in an Integrated Building Design Environment, as opposed to as a stand-alone tool, is to make its functionality available to other applications. When the application is implemented in an Integrated Building Design Environment, it is able to benefit from the services provided by other applications. Thus, each application should be able to combine the virtues of being small, simple, and specialized. The most flexible way this can be done is by creating the applications such that their functions can be called dynamically.

It has been decided to realize the system in MS Windows using Microsoft Foundation Classes (MFC) for constructing the user interface. The programming language should be object oriented, and for this, MS Visual C++ has been chosen. The current implementation platform is not very high end (Windows 3.1 and Visual C++ ver 1.5), which can be seen as a reflection of the overall purpose of this project: Not to use more sophisticated tools than necessary, but rather to get some basic product data technology into practical use.

Most applications in the system will be implemented as so-called Dynamic Link Libraries (DLL) for MS-Windows. The DLLs will be activated by other DLLs or, at the entrance to the Integrated Building Design Environment, by a system application which is an independently executable program. The system application controls the execution of the DLLs and acts as a client of the services provided by the DLLs. The DLLs in turn, serve their calling applications (e.g. the system application) and may in turn be served by other DLLs.

Each of the applications may be quite small, and carry out only very dedicated elements of an overall analysis. Calculation of U-value of an opaque construction could be one example of such a small

application. In one configuration of the Integrated Building Design Environment, the U-value calculator may provide input for a tool that checks compliance with building regulations (i.e. whether each building element's U-value complies with certain threshold values). In another configuration of the Integrated Building Design Environment, the same U-value application may be called to provide heat transfer coefficients for a detailed building thermal analysis tool.

Each application has its own data model which is a definition of the data types it uses either internally or to exchange information with other applications:

- *Exchangeable data.* Most building data are used by several, different applications. Every data type will have one original declaration within one of the data models in a system of co-operating applications. An application that uses data from another application's data model will refer to such data through so-called external references.
- *Internal data* that will not be needed by other applications should not be represented in the common data model for the integrated system. The representation of internal data is beyond the scope of this paper.

Implementing a new application is carried out through the following phases, as will be detailed in section 5:

1. Definition of a data model for the application.
2. Developing an interface for the application, i.e. setting up the application's data exchange facilities.
3. Implementing the application's functions, and its user interface (if it has one), in the form of a DLL for MS-Windows.

4. COMPONENTS OF SBI'S INTEGRATED BUILDING DESIGN ENVIRONMENT

Two basic developments have been prepared in order to facilitate STEP data exchange within the system: (1) A code generator *Expresso*, and (2) a set of C++ classes with functionality for accessing data from or writing data to a STEP file. A third development is the *DIS*, an overall tool to handle the applications in the system.

4.1 Expresso

Expresso is a stand-alone program which takes the data definition of an EXPRESS file and produces C++ classes corresponding to each entity of the EXPRESS file's data model. In addition, an overall C++ class is created for the data model as such. The Expresso generated code forms the basis for the programmer's writing of code for a new application as it mainly consists of the classes to build upon when writing the functionality of an application.

4.2 The STEP classes

Each of the Expresso generated classes inherit from certain C++ classes that provide the functionality of a STEP interface. The Expresso generated class that represents the model will inherit from a *STEPmodel* class, while the individual classes that belong to the model will inherit from a *STEPheader* class. An instance of STEPmodel contains a number of instances of STEPheader. These two classes together make it possible for a sibling of STEPmodel, with the STEPheader siblings it contains, to read from a STEP file - or to write one. Another important class in the STEP interface kit is the *STEPaggr*

which is used to hold aggregate members of variable size and has functionality for recursing, adding and deleting members to the aggregate.

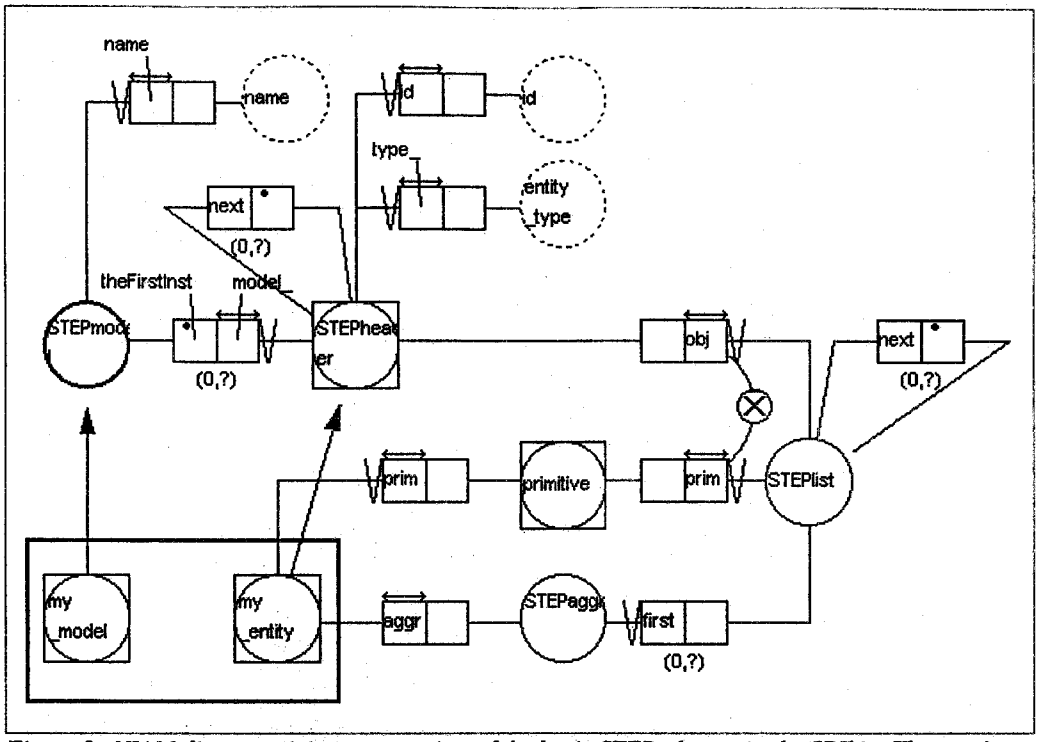

Figure 2. NIAM diagram giving an overview of the basic STEP-classes in the SBIkit. The user's own classes for a submodel inherit from STEPmodel and STEPheader.

Expresso and the STEP classes work hand in hand, and therefore, have been developed concurrently. The two developments together have been called SBIkit, for *STEP Building data model Interface kit* (Grau and Rode, 1996).

Figure 2 gives an overview of the classes in the SBIkit, how they relate to each other, and how the user's own classes inherit from the STEP classes.

4.3 DIS

An application called *DIS* (*Danish Integrated System handler*) has been developed for configuration and execution of the applications in various arrangements of the Integrated Building Design Environment. The system application is the main entrance to the integrated system and acts as the master application for the DLLs. The DLLs in turn are divided into two types:

- *IBDE Applications* performing separate stand-alone type tasks but utilizing data from and contributing data to the common data model. An energy simulation tool could be an example of one such IBDE Application.

- Below the IBDE Applications are the *Submodels*, which are small DLLs that act as servers for the other applications in the system. The Submodels contribute to the system's overall data model but have no access to other than their own data model. A small tool to calculate the thermal transmittance (U-value) of constructions could be one example of such a submodel.

Figure 3 illustrates a sample configuration of an Integrated Building Design Environment using the SBIkit and the DIS system application. STEP.DLL and MFC.DLL in the bottom of the figure, are Dynamic Link Libraries which hold the SBIkit's data exchange methods and Microsoft Foundation Classes (MS-Windows functionality), respectively.

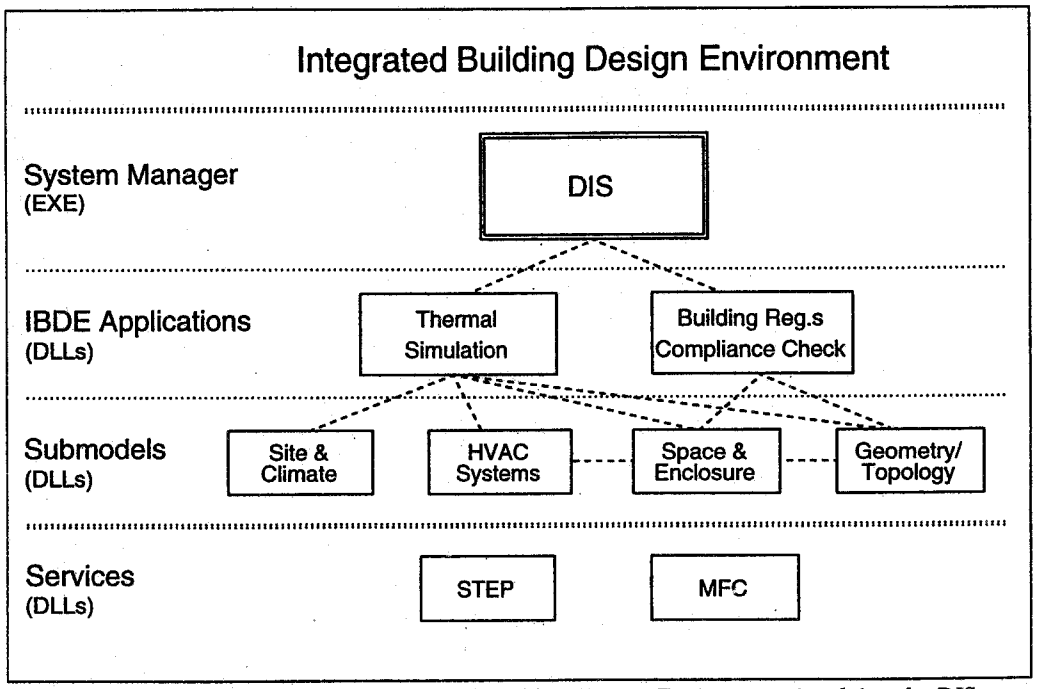

Figure 3. Example of a configured Integrated Building Design Environment involving the DIS system application, 2 IBDS applications and 4 submodels as DLLs.

5. OVERVIEW OF THE IMPLEMENTATION OF A NEW APPLICATION

The procedure that has to be followed to produce a working STEP interface for a new application encompasses the following steps:

1. Declare the application's data model as a NIAM diagram (Nijssen Information Analysis Method, Nijssen and Halpin, 1989) using a graphic NIAM editor capable of exporting the data definition in EXPRESS format.
   The NIAM diagram describes the data model's *object types* and relations between the object types. Of course, any other graphic diagramming technique which is able to export EXPRESS data definitions can be used in stead of NIAM, e.g. EXPRESS-G.

2. Let the NIAM editor export the data model as an EXPRESS file. The EXPRESS file is text based, and thus, the structure can be parsed by a computer program. The EXPRESS schema can also be written manually using a text editor.
   The data model for one application must be contained within one so-called *schema* which contains a number of *entities* - corresponding to NIAM's notion of object types.
3. Run SBI's Expresso tool to interpret the EXPRESS data model and produce a corresponding data structure in C++. Expresso generates a C++ class for each of the entities in the EXPRESS file, and a special C++ class to represent the model as such.
4. Program the C++ source code of the application's functionality in a normal way, but using the Expresso generated data structure.

6. CURRENT IMPLEMENTATIONS

The current status of SBI's developments is that the above mentioned components are available in an almost stable state. SBI is now using this as the basis to create a set of applications which in combination will have at least the same functionality as the institute's *tsbi3*-program for transient analysis of the thermal performance of multi-zone buildings. Compared to *tsbi3*, the additional functionality concerns matters that can utilize the fact that full 3D geometric information will be available in the new setting, e.g. 3D geometric/topological modelling of the building, stratification of room temperature, and calculation of internal longwave radiation.

First of all, as much experience as possible from *tsbi3* (written in C) will be carried over and ported to a similar but new and extended data structure in EXPRESS and C++. Roughly, one could say that *tsbi3* is being split into parts and reassembled in the new setting. Different submodels are developed for the various subjects employed in a building thermal analysis tool:

1. A 3D Geometry/topology model for overall layout of the building as Cells, Faces, Edges and Vertices.
2. A Space model for handling the real life space objects (Indoor, Outdoor, Ground) that have the topology model's Cells as representations and uses of the spaces.
3. An Enclosure model for handling the real life enclosing elements (and "holes") which have the topology model's Faces as representations.
4. A Component model for access to standardized building components and databases for these.
5. A System model for handling various HVAC systems and operations.
6. A Site model to handle issues about the building's surroundings.
7. A Climate model for access to weather data.
8. A Psychrometric model for handling hygrothermal data.

...

Each of the above mentioned models have been (or will be) implemented as DLLs, and thus need only be called upon when necessary. When a DLL is activated, a pointer to the data model for the calling application is passed as a parameter and the called DLL takes care of binding the models together. One instance of the composite data model exists as a global variable such that all DLLs in the system have access to the union of data models for the active DLLs.

As indicated by the ellipsis in the listing above, the set of models is far from complete and will be extended and modified in the near future. Until today most work has been devoted to the implementation of code for the first five models. As of today, a simple hour-by-hour thermal calculation can be carried out for a composition of a building described in 3D with windows and doors positioned in the faces.

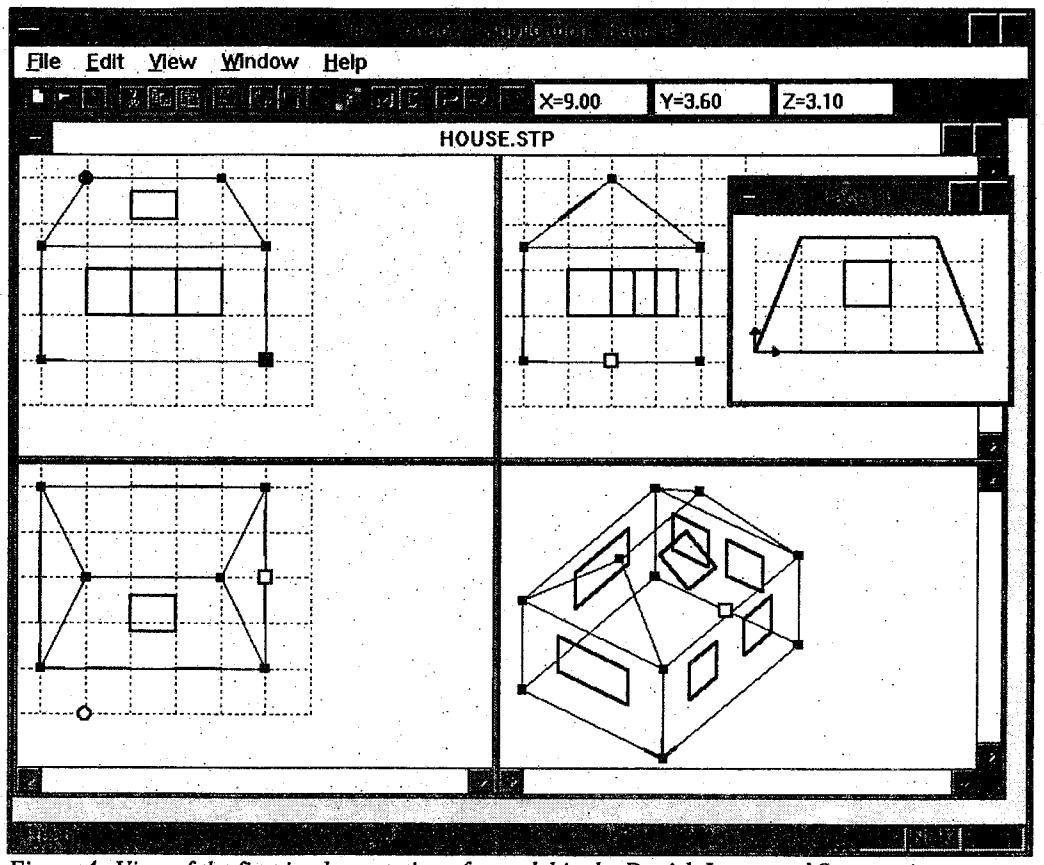

Figure 4. View of the first implementation of a model in the Danish Integrated System: A geometric/topologic modeller.

A first version of a system for building thermal analysis is scheduled for August 1996. This system will be presented to two engineering companies with architect partners in a project where further adjustments will take place in collaboration with the people from practice. The practitioners will also attach some of their in-house models to the integrated system, such that they can benefit from inheriting an already defined building description.

The intention for the further development is to attach models for other areas which can utilize the same building data. An obvious extension is to attach models for analyses of the environmental impact of

buildings. Such analyses can be carried out by summation of specific data about the same building elements as were used in the thermal simulation.

7. DISCUSSION AND CONCLUSION

Product data technology and systems for integration of architectural and engineering software have been research issues for quite some years now. While we are still waiting for stable versions of STEP Application Resources and Application Protocols that cover the construction area, the basic parts of the ISO STEP standard - the data definition and exchange methods - have been final standards for a couple of years and have been tested even earlier than that in for instance the European COMBINE project. The time is ready, therefore, to try to actually get some of this technology out to be used by practicing building designers.

The aim of the work described in this paper is to improve on today's stand-alone, monolithic analysis tools by providing methods for exchange of STEP data. Soon hereafter these developments will be presented to building designers in order to get rapid feedback on how the system actually performs in an actual user's setting and to carry out immediate improvements. As shown in Figure 1, a "Utilization in Practice" project of the Danish system has already been granted on national funds and is waiting for the first prototype of the DIS to be completed. The field test is anticipated to commence in the early fall '96.

One of the intentions with this paper is to invite interested co-developers to work along the ideas presented. Developers who are interested in trying to use the SBIkit are welcome to contact the authors and have access to further documentation as well as to the actual routines.

8. REFERENCES

Augenbroe, G.L.M. (editor) (1995). COMBINE - Final Report. Delft University of Technology.

Augenbroe, G.L.M. (editor) (1995). COMBINE 2 - Final Report. Delft University of Technology.

Grau, K. and C. Rode (1996). SBIkit, STEP Building data model Interface kit. Danish Building Research Institute.

Johnsen K. and K. Grau (1994). *tsbi3* - User's Guide. Danish Building Research Institute.

G. A. van Nederveen (1995). COMBINE 2 Task 2 Report. TNO Building and Construction Research.

Nijssen, G.M. & T.A. Halpin (1989). *Conceptual Schema and Relational Database Design*. Prentice Hall.