INTEGRATION OF BUILDING ENERGY SIMULATION AND HVAC DESIGN TOOLS IN THE COMBINE PROJECT

Marc Mellotte, Jim Flynn and Paul F. Monaghan
Thermal Engineering Research Unit (CATERU)
Manufacturing Research Centre
University College Galway
Ireland

ABSTRACT

Software packages form an integral part of the practices of most building design professionals today. Studies in the UK show that CAD is used in 81% of architectural practices [11. However, these packages fali short of ideal when it comes to data exchange.

The objective of the COMBINE project is to demonstrate a solution to this problem based on the developrnent of an Integrated Building Design System (IBDS) [2]. This solution revolves around two principies, namely (a) the developrnent of a centralised. building data model and (b) the integration. of existing software applications with this data model to form the IBDS. At the conclusion of the project, this paper discusses the authors' work on the integration of three conunercial packages, DOE-2DX, AutoCAD and VENT, into the IBDS.

INTRODUCTION

The problem of data transfer between existing commercial software packages is not a new one. Many have been frustrated by having to transfer detailed data from one application to another by hand. Partial solutions exist in the "Integrated Toolkits" on the market and in standards such DXF. However, these toolkits are proprietary in nature and not very compatible with other applications. Standards such as DXF were developed for a specific purpose, e.g. transfer of geometric data, and struggle when it comes to other areas of data transfer.

The approach adopted in the COMBINE project is to apply a standardised method of data exchange, incorporating an entire set of techniques for the complete description and implementation of a data exchange system. The ISO Draft International Standard 10303 STEP (Standard for the Exchange of Product Model Data) [3] is just such a standard. Wilson [4] describes how it has been developed drawing the experience of the European Esprit Standard CAD*I[5] work and the US Product Data Exchange Standard (PDES)[6,7].

The core of the IBDS is the data model (i.e. building model). This model is designed to be capable of supplying the data input and output needs of all the applications in the system (e.g. HVAC, Architectural, etc.). Models of this type are referred to as *product models*. Within COMBINE, the product model is called the IDM (Integrated Data Model). The STEP standard was created specifically to support the development of such product models. Currently the standard consists of 10 parts (it is still under development). The sections most relevant to this discussion are:

PART 11: Description Methods: The EXPRESS language reference manual. This specifies an object-oriented data modelling language, suitable for describing the data structures of a wide range of programs[3].

PART 21: Implementation Methods: Clear text encoding of the exchange structure. This part of ISO 10303 specifies a text file format for storing the data described by the EXPRESS model[3].

PART 22: Standard Data Access Interface (SDAI). The SDAI specifies an Application Program Interface (API) to data which has been defined using the EXPRESS language [3].

Fig. 1 shows the relationship between parts 11 and 21 of the standard. The file on the left is an example of what EXPRESS code looks like. Here, we have a trivial model describing just walls and doors. The model has two entities, *wall* and *door*. These two entities have attributes. The attributes of wall are its *id*, *area*, *U-value* and the *doors* it contains. The type of each attribute is also given, id is an integer, area and U-value are real numbers and the doors attribute relates to a SET of zero or more door entities. The file on the right of the diagram is an example of a STEP file that contains data as modelled by the EXPRESS file on the left. Each line has a similar format. It begins with a # symbol and a reference number, relevant only within this particular file. Next is the entity type, e.g. WALL and in parentheses the value of each attribute of this wall.

These attributes are listed in the same order as they are described in the EXPRESS model. So, on the first line of this example, we have a wall with:

- id = 23
- area = 5.4
- U-value = 0.9
- doors = (#2,#3)

The doors attribute tells us that this wall has two doors in it and these are doors #2 & #3 in the STEP file. The next two lines of data in the STEP file describe these doors.

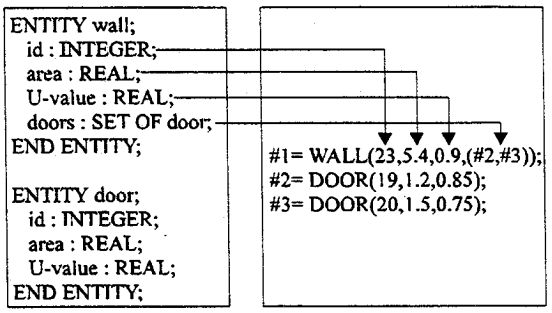

Figure 1. Relationship between EXPRESS code and STEP data files

Attributes that are common to many entities can be placed into an entity that is a *supertype* of others. In the above example, the attributes id, area and U-value could be placed in to an entity called *enclosing_element*. Wall and door then inherit from enclosing_element (*subtype* of enclosing_element) and thus have the attributes id, area and U-value automatically. In the IDM, every entity inherits (at least) the attributes id (integer) and description (string).

The structure in an actual data file for a building would obviously be quite complex. The top layer would be the building, which would have spaces and these spaces would have walls, floors, etc. with materials and properties, thus forming an object-oriented hierarchy.

SYSTEM ARCHITECTURE

The overall objective of COMBINE II, (Nov. '92-May '95) is to follow on from COMBINE I (Aug. '90-Nov. '92)[8] in the development of a prototype Integrated Building Design System (IBDS). Fig. 2 shows the architecture of this IBDS. Briefly, the components this system are:

Data Exchange Kernel (DEK): The DEK is the core of the system. It is an ObjectStore [9] object oriented database that stores the actual building data modelled by the IDM (i.e. STEP files). The database is connected to each application in the system by Data Interaction Managers (described below).

Simulation Programs: These programs take a set of input data, perform calculations and produce output data. For this reason, their communication with the database is file based (input & output data are stored in files) and these tools are termed *off-line*. The integration of one off-line tool, DOE-2DX, is described in this paper.

CAD Program: The CAD program used here is AutoCAD customised for HVAC air-based ductwork design. This tool is termed *on-line* because, unlike the simulation programs whose input and output is a data file, the database is updated after each action performed by the user.

Data Interaction Manager On-Line (DIMON): The DIMON is the communication link between the CAD tool and the DEK (database). Having a link like this between the CAD tool and the database allows either of them to be changed easily. If the CAD tool spoke directly to the database, the database would need to be re-coded each time it was changed.

Also, the on-line nature of the CAD tool could cause problems if a requested action led to an invalid model e.g. to place a duct through a solid wall. The DEK cannot perform this type of checking, so the DIMON performs the task. When the CAD tool is started, the user sends a request (through a menu item) to open a building in the database and at the end of a session, the user closes the database. The DIMON handles these requests.

Data Interaction Manager Off-Line (DIMOFF): The DIMOFF is the communication link between the off-line simulation tools and the DEK. The role of the DIMOFF is to supply the input STEP file to the application when requested and to read in the output STEP file produced by the application when finished its calculations. The DIMOFF writes this data to the database.

Exchange Executive: The exchange executive is the software that controls the system. It graphically displays the options available at each stage in a project using a petri-net model. If the user selected the option of performing a DOE-2DX run, the Exchange Executive prompts the DIMOFF to produce the input file for the application. Also among its duties is intersection checking i.e. comparing the output schema of one application with the input of others to determine if there are any overlaps. If there are it requests whether the user wishes to rerun the other application due to the

change in its input data (e.g. if the heating load of the building changes, the ducting may need to be resized).

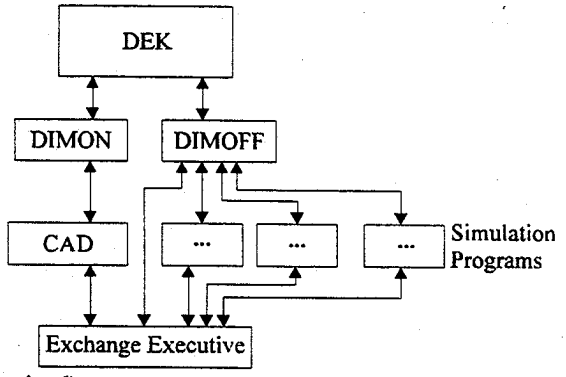

Figure 2. IBDS architecture

INTEGRATION OF DOE-2DX INTO THE IBDS

DOE-2DX [10] is a Fortran program developed by the US Department of Energy for simulating the energy behaviour of buildings and their associated heating, ventilation and air-conditioning (HVAC) systems. To fully integrate an application such as this into the IBDS, there are two fundamental tasks to perform:

- Create a DOE-2DX input file out of the STEP file from the DEK.
- When DOE-2DX has finished its calculations, create a STEP file from the DOE-2DX output file to return to the DEK.

All of the development work was done in C++ using Microsoft Visual C++, versions 1.0 & 1.5. The other main tool used was the Interface Kit (IntKit). This COMBINE tool parses an EXPRESS file and generates C++ code from it, creating a class for each entity (see fig. 1) in the EXPRESS model. This generated code is then SDAI compliant, i.e. it can access a library of standard functions that enable it to read in STEP files, manipulate the data they contain and write out STEP files.

The process by which this application was integrated into the IBDS is described here and can also be followed in fig. 3 & fig. 4.

Creation of a DOE-2DX input file using a STEP file from the DEK

To run DOE-2DX, an input file is needed. The steps involved are:

1. The data required by DOE-2DX to perform its calculations was first modelled in the EXPRESS language. This leads to the production of an input model or *aspect model* of DOE-2DX.
2. The subsection of the IDM from which the input data required by DOE-2DX can be extracted was isolated into a *subschema* or partial IDM model in EXPRESS
3. The IDM subschema was merged with the DOE-2DX aspect model to form one large file.
4. A mapping strategy was drawn up to determine how to fill the DOE-2DX model with the data contained in the IDM model.
5. The combined model of IDM plus DOE-2DX was passed to the IntKit which generated the SDAI compliant C++ code (see fig. 3).

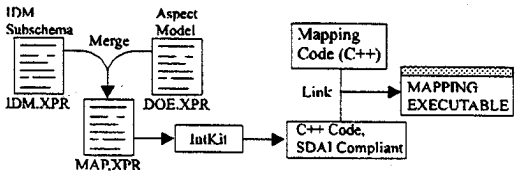

Figure 3. Schematic of mapping executable creation

6. The code that maps the data in the IDM model to the DOE-2DX model was then written. The mapping functions common to all teams (doing this type work) were identified and their development shared out. Typical mapping functions identified were to get the area of a floor in a room, to transform all co-ordinates to local co-ordinates, to calculate the tilt of a wall etc..
7. This mapping code was then linked with the generated code to create a mapping executable (see fig. 3). At run time, this executable reads in a STEP file of a building from the DEK and creates C++ objects that correspond to instances of the entities in the IDM EXPRESS model (e.g. buildings, spaces etc.). For each IDM building object found, a DOE-2DX building object is created and written to the output STEP file. The details of the building (ID, Name etc.) are mapped to the new DOE-2DX building. The procedure then moves on to the individual spaces in the IDM building and maps these to DOE-2DX building spaces etc..
8. As well as writing an input STEP file for DOE-2DX, the mapping executable also writes an input DOE-2DX file. This is the file that DOE-2DX will actually use to perform its calculations.

Creation of a STEP file to return to the DEK from the DOE-2DX output file

The second step in the integration process is to create a STEP file from the output file of DOE-2DX for return to the database. The stages of this process are:

1. The output data of DOE-2DX was modelled in EXPRESS to produce an output aspect model and

C++ code was generated from this (using the IntKit).

2. It was originally planned to include in the IDM a performance model that would be representative of the output data of all the off-line tools on the project. In the end, a performance model generic enough to cope with all the tools was not achieved and the aspect model of each off-line tool was inserted into the IDM (see conclusions section). This means that there is no mapping to be performed with the output data.
3. To read in the output file created by DOE-2DX, code was written that parses this output file and extracts the data, writing it to an output STEP file.
4. This STEP file sent back to the DEK.

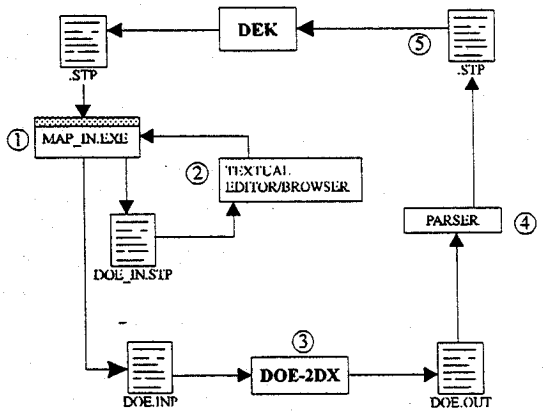

Figure 4. Integrated DOE2-DX

The completed integration of DOE-2DX is shown in fig. 4. The only component not mentioned till now is the *Textual Editor/Browser*. This a Windows™ GUI application. It reads in a STEP file and displays the data on screen under various categories (e.g. occupancy schedules, space conditions, run periods, etc.). This allows the user to manipulate (exclude, change etc.) the data and when finished, produce modified STEP and DOE-2DX input files.

With DOE-2DX now integrated into the system, the process of performing an energy simulation analysis on a building is as follows (see steps in fig. 4):

1. Download the building STEP data file from the DEK. The data is mapped into DOE-2DX format. An input STEP file (fig. 5) and DOE-2DX input file are produced.

2. The user can examine the input STEP file produced in the textual editor/browser and, for example, exclude some data from the DOE-2DX input file. The editor then instructs the mapping executable to create a new DOE-2DX input file.

3. DOE-2DX performs its simulation and produces an output file of results.

4. The parser strips the data from the DOE-2DX output file and writes it to an output STEP file (fig. 5).

5. This output STEP file is mapped back to IDM format and returned to the DEK.

```
#1=BUILDING(21,'simple house',981,(#2,#3,#4));
#2=SPACE(32,'hall',#6,12.1,23.8,(#7,#8,#9,#10,#11,#12));
#3=SPACE(33,'bedroom' ,#6,12.1,23.8,(#7,#8,#9,#10,#11,#12));
#4=SPACE(34,'kitchen' ,#6,12.1,23.8,(#7,#8,#9,#10,#11,#12));
#5=SPACE_COORDINATE_SYSTEM(21,$ , 12.1, 11.5, 32.0);
#6=INTERIOR_WALL(105,' INT_WL105',3.0,1.5,4.5,,#17);
#7=EXTERIOR_WALL(64,'EXT_WL64',5.5,3.0,16.5,#17);
#8=DYNAMIC_MATERIAL(1714, 'MAT-1714',0.016404,5.777, 0.624,0.0023);
```

DOE2-DX input STEP file

```
#15=DOE2_SPACE_PEAK_LOADS_SUMMARY_REPORT(-1,'LS-A report',$,#14,
(#7));
#14=DOE2_BUILDING_PEAK_LOADS(-1,$,#13,#12);
#12=DOE2_COOLING_LOAD_REPORT(-1,$,50.786000,#8);
#8=DOE2_LOAD_CONDITIONS(-1,$,32.777778,25.555556,#9);
#9=DOE2_DATE(-1,$,1995,20,.JUN.,$);
#10=DOE2_LOAD_CONDITIONS(-1,$,-21.666667,-21.666667, #11);
#11=DOE2_DATE(-1,$,1995,12,.JAN.,$);
#13=DOE2_HEATING_LOAD_REPORT(-1,$,#10,-64.092000);
#47=DOE2_SPACE_PEAK_LOAD_COMPONENTS_REPORT(-1,'LS-B report',$,
#20);
```

DOE-2DX output STEP file

Figure 5. Sample DOE-2DX input and output STEP files

This completes the process of integrating the application DOE-2DX.

INTEGRATION OF THE AUTOCAD HVAC TOOL INTO THE IBDS

A major component of the IBDS is the CAD application, here AutoCAD customised for ductwork design. As mentioned in the section on IBDS architecture, the CAD tool is an on-line application. The work on the CAD tool can be broken into two areas:

1. Customisation of AutoCAD for the Ductwork Sizing Calculations.
2. The communication links between:
   - AutoCAD and the DIMON
   - AutoCAD and the Building Components Database
   - AutoCAD and the VENT calculational module

The BCD was developed on the project and is simply a database of industrial standard fittings (dampers, bends, etc.). Based on data it receives from other applications, the BCD can suggest a list of suitable fittings for the user to choose one.

The other component in this section is the VENT calculational module. VENT was developed by HevaComp[11] and performs ductwork sizing calculations. Here it performs its calculations based on data it receives from AutoCAD. The completed system is schematically shown in fig. 6.

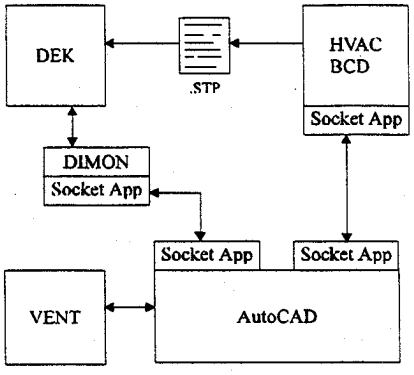

Figure 6. Communication schematic

As the diagram shows, AutoCAD communicates with both the DIMON and the BCD via a TCP/IP sockets application. The details of this communication are discussed later. The diagram also shows the communication between AutoCAD and the VENT calculational module.

Customisation of AutoCAD for the Ductwork Sizing Calculations

AutoCAD provides text based tools for creating/editing menus and dialog boxes. Using these tools, AutoCAD was customised to include two new pull-down menus shown below, giving the *Ductwork Sizing Module* interface. Fig. 7 shows the end result where two new menus have been added to the standard AutoCAD menus.

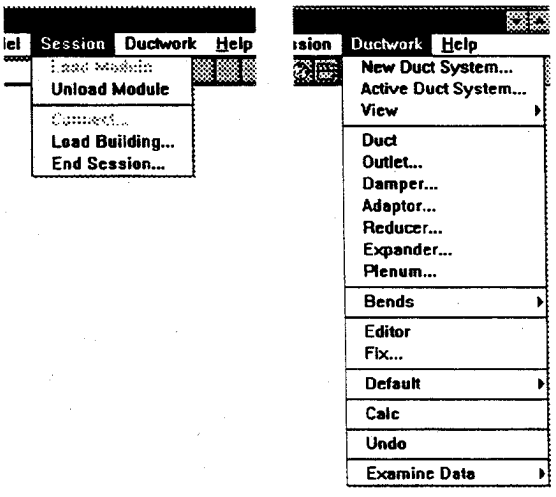

Figure 7. Customised menus in AutoCAD

The AutoCAD Development System (ADS) C library [12] provides a means of developing a set of external functions that are called when required by AutoCAD. There is an individual function corresponding to each menu item shown in the new menus which is called when that choice is selected.

This interface now allows the user to download a building floor and its corresponding ducting systems from the database and edit these systems or create new ones as necessary. The fittings listed in the ductwork menu (fig. 7) may be added to the active duct system in the AutoCAD drawing. The user can also select which ducting systems are active (displayed in a different colour), or visible, at any time. The various fittings are represented by 2-D symbols which can be independently turned on or off.

Communication Links

Sockets provide a means of software communication between any two processes through the use of the Internet Protocol Suite (IPS, generally referred to as TCP/IP). The main advantage of this approach over other means of communication (DDE etc.) is that the two applications need not be running on the same machine (i.e. distributed applications). In fact, the two applications need not necessarily be running on the same platform type. For example, two processes located on different machines running NT & UNIX, WFW & NT, etc. can all communicate with one another via a sockets protocol. The most commonly used paradigm in constructing distributed applications is the client/server model. In this scheme, client applications request services from a server application. There is a list of possible functions that can be invoked by the client and requested of the server.

The links between AutoCAD and both the DIMON and BCD have been established. If the user selects the option to create a duct in the AutoCAD menu, it calls a function in the DIMON, say *CreateDuct*. The DIMON checks to see if this is a valid request and allows/disallows the request depending on the result of its check.

The situation is similar between AutoCAD and the BCD. Here there is only one function that can be called. This function passes data to the BCD about what type of fitting is required and the BCD uses this data to suggest a list of one or more standard fittings. When the user chooses the appropriate fitting, the details of this choice are sent to the DEK in a STEP file to update the building model.

The VENT calculational module has a file based communication link with AutoCAD. This means

that AutoCAD sends VENT a file containing the data it requires and, when finished its ductwork sizing calculations, VENT returns the results to AutoCAD in a new file, allowing AutoCAD to update the model.

ACKNOWLEDGEMENTS

The authors would like to thank Mr. Derek Brady of CATERU, Mr. A. J. Baxter of HevaComp Ltd. and all of the partners in the COMBINE project, TU Delft (NL), TNO-BBI (NL), TNO-CIC (NL), SBI (DK), BRE (UK), University of Newcastle (UK), ESRU (UK), UCD (IRL), FIB (D), VTT (FIN) and CSTB (FR).

CONCLUSIONS

There exists a problem of data transfer between the various building design software packages at present. The COMBINE project attempts to overcome this problem through the development of a prototype IBDS. The result is a system where the user may:

1. Retrieve the building model from the database and perform an energy analysis on it using the energy simulation package DOE-2DX. The results may or may not be then sent back to the database.
2. Download the building model from the database into their own CAD tool (AutoCAD with the HVAC add-on in this case). The ducting system(s) can then be analysed and new ones created. The VENT module will size the ducting, based on the relevant data from the DOE-2DX run. Fittings are chosen from the Building Components Database.
3. View a centreline representation of the ducting system in AutoCAD with 2-D symbols showing the fittings.

The benefits that this system offers are:

Easily expandable to include new applications.
Reuse of applications familiar to the user (AutoCAD, DOE-2DX etc.).
Ease of use, most of the data transfer work is done in the background without the user even knowing what's going on.

The completed system was demonstrated to members of the public at a seminar/workshop on 30 & 31 May 1995 at University College Dublin, Ireland. The major objective of COMBINE (to develop the IBDS) was achieved and demonstrated. There were some elements of the work that were not finished.

VENT link to the CAD tool

- Incorporation of DOE-2DX editor/browser into the system

The background work for the VENT link was done and the actual editor/browser was developed as a standalone application.

With the project finished, the authors conclude that there are certain areas that need improvement:

- The central model (IDM) - Although strong in areas such as space & enclosure, geometry, topology, the schedule model needs improvement. This area posed particular problems for the DOE-2DX work. Another area in need of work is the performance model. This is the section which describes the data returned by all of the analysis tools. In this implementation, a generic model was not achieved and the output model of each individual tool was inserted into the IDM.
- IntKit - This is the tool that generates C++ code from EXPRESS. The resulting code is capable of reading and writing STEP files. The problem is the performance of this code. It takes too long to read in the data. For example, a two-storey domestic house with a basement could take up to 20 minutes to read in. Obviously this needs improvement.
- SDAI code - The code generated by the IntKit is SDAI compliant (contains the data structures to read and manipulate data in STEP files). However, there is no query language available for this code i.e. if the user needed to find the window in a particular wall, he would need to match the id of that window against the id of the window(s) in every wall in the building until a match is found. This prolongs development time and makes the work very tedious.

Although, there were problems and not all of the original aims were achieved, the overall impression was that the project was a success. The work does not stop there, however. Many of the teams involved are taking the most promising elements of their work and continuing development. The HVAC CAD tool will be developed further to complete the link with the VENT tool and the technology of the editor/browser is to be reused in other areas. There is also a proposal with the European Commission for funding for a follow-up project which would involve industrial partners willing to perform field tests of the software. In the UK, the Building Research Establishment (BRE, one of the project partners) are already heading a nationally funded project of this type. The input of practitioners is invaluable in the further development of these tools.

REFERENCES

[1] S. Drewer, *Myth and Reality in the Use of IT and Computer Based Technologies in Construction*, University of the West of England, Faculty of the Built Environment, Bristol, UK, 1993.

[2] G.L.M. Augenbroe, ed., *COMBINE - Computer Models for the Building Industry in Europe*, Final Report, CEC DGXII Contract no: JOUE-CT90-0060, 1993.

[3] ISO TC184/SC4, *Draft International Standard 10303, STEP Industrial Automation Systems - Product Data representation and exchange*, Parts 11, 21, 22, 1993.

[4] P.R. Wilson, *A Short History of CAD Data Transfer Standards*, IEEE Computer Graphics and Applications, 1987.

[5] E.G. Schlectendahl, ed., *Specification of a CAD*I Neutral File for Solids*, Version 2.1, Volume 1 of Research Reports ESPRIT Project 322 - CAD Interfaces (CAD*I), Springer-Verlag, 1986.

[6] K. Brauner and D. Briggs, *The Second Draft Report of the Ad-hoc Committee on the Content and Methodology of the IGES Version 3 (The second PDES Report)*, IGES Internal Report, NBS, Gaithersburg, Maryland, 1984.

[7] *Initial Graphics Exchange Specification (IGES)*, Version 3.0, NSB, Gaithersburg, Maryland, 1986.

[8] J. Kennington and P.F. Monaghan, *COMBINE: The HVAC-Design Prototype*, Building and Environment, 1993.

[9] Object Design, Inc., *ObjectStore User Guide*, Burlington, MA, USA, 1994.

[10] Acrosoft International, Inc, *Micro-DOE2 User's Guide*, 1990.

[11] Hevacomp Ltd., *HevaSketch User's Manual*, Sheffield, UK, 1993.

[12] AutoDesk, Inc., *Using AutoCAD for Windows™*, 1993.

[13] J. Flynn and P. F. Monaghan, *Integration of CAD with Simulation Programs through the use of STEP and OODB Technology*, European Conference on Energy Performance and Indoor Climate in Buildings, Lyon, France, 24-26 November 1994.