Development of an Integrated Product Information Management System

Abidemi Owolabi

Corus UK Ltd
Swinden Technology Centre
Moorgate, Rotherham
South Yorkshire S60 3AR

Centre for Innovative Construction Engineering (CICE)
Department of Civil & Building Engineering
Loughborough University
Loughborough
Leics, LE11 3TU

DEVELOPMENT OF AN INTEGRATED PRODUCT INFORMATION MANAGEMENT SYSTEM

By
Abidemi Owolabi

A dissertation thesis submitted in partial fulfilment of the requirements for the award of the degree Doctor of Engineering (EngD), at Loughborough University

September 2004

Corus UK Ltd
Swinden Technology Centre
Moorgate, Rotherham
South Yorkshire S60 3AR

Centre for Innovative Construction Engineering (CICE)
Department of Civil & Building Engineering
Loughborough University
Loughborough
Leics, LE11 3TU

ACKNOWLEDGEMENTS

I would like to thank (in chronological sequence):
1992-1998:

Dr. Ayo Oluleye of the University of Ibadan, Nigeria, for teaching me systems development in practice and for being my personal coach.

1999-2000:

Dr. Kabiru Olajire for suggesting and encouraging me to undertake an Engineering Doctorate in Innovative Construction Engineering at a time when it seemed an absurd choice and a daunting task to me.

2000-2002:

Tim Aikin and George Stevenson for teaching me the technological and business intricacies and peculiarities of the construction industry.

2000-2004:

All the people on the projects I interviewed and observed, who contributed materially to this doctoral thesis.

Dr. David Martin of Construction Applications Department, Corus UK Ltd, for seeing the project's potential and championing its sponsorship.

Colin Harper and Phillip Parkin of Construction Technology Knowledge Group, Corus UK Ltd, for all the days and nights we spent together on the development of PROMIS and for their immeasurable knowledge input that saw me through to the completion of this programme.

My academic supervisors, Prof. Chimay Anumba and Dr. A. El-Hamalawi, for their insight on the subjects at hand, for repeated careful reading of draft texts, for guidance on relevant literature and direction of the research, and for their unflinching support throughout the programme.

I am grateful to Dept. of Civil and Building Engineering, Loughborough University and Corus (UK) Limited for sponsoring this programme.

I would like to appreciate my mum, Mrs. Mary Owolabi, and friends, Jean and John Hodgkinson, for keeping my boys, Toluwa and Tomisin, away while I worked on this thesis. Finally, I would like to thank my wife, Dr. Idayat Bolanle Owolabi, for all the encouragement, the interest, the sacrifice and for everything else that made the task just that little bit easier. To God indeed, be all the glory.

ABSTRACT

This thesis reports on a research project undertaken over a four year period investigating and developing a software framework and application for integrating and managing building product information for construction engineering. The research involved extensive literature research, observation of the industry practices and interviews with construction industry practitioners and systems implementers to determine how best to represent and present product information to support the construction process. Applicable product models for information representation were reviewed and evaluated to determine present suitability. The IFC product model was found to be the most applicable. Investigations of technologies supporting the product model led to the development of a software tool, the IFC Assembly Viewer, which aided further investigations into the suitability of the product model (in its current state) for the exchange and sharing of product information. A software framework, or reusable software design and application, called PROduct Information Management System (PROMIS), was developed based on a non-standard product model but with flexibility to work with the IFC product model when sufficiently mature. The software comprises three subsystems namely: ProductWeb, ModelManager.NET and Product/Project Service (or P2Service). The key features of this system were shared project databases, parametric product specification, integration of product information sources, and application interaction and integration through interface components. PROMIS was applied to and tested with a modular construction business for the management of product information and for integration of product and project information through the design and construction (production) process.

KEY WORDS

Product libraries, project information, data sharing, application integration, product modelling, IFC

PREFACE

The Centre for Innovative Construction Engineering (CICE) is a unique university centre dedicated to industry-focused research, innovation and training. It was set up to address the technical and business needs of the construction industry through a targeted, high quality training programme - the Engineering Doctorate (EngD) programme. The EngD programme provides the candidates with a good balance of academic training, industry experience and high level research. In return, it requires the candidates to apply the knowledge and expertise acquired in a business environment and to demonstrate innovation in that application. This thesis demonstrates the actualisation of the goals of CICE, the EngD programme and, not least, that of the candidate, me!

Upon completion of a Masters Degree programme in Computer Science - Advanced Computer Systems, the opportunity to apply the knowledge gained came through enrolment on the EngD programme, but in Construction Engineering. This was a daunting undertaking given I had no previous exposure to the field. It was no doubt a path laden with challenges, unprecedented in my career. For me, it was like a long distance hurdle race against time.

If indeed this was a hurdle race, I did not compete fairly. There were people all along the way removing hurdles to make the race easier. From supervisors who worked weekends to colleagues who were untiring in explaining construction techniques from piles to façades. I also had help at the home front. I hope someday my 2-year old would understand why I did not appreciate his attempts to put in a few “words” into my thesis.

On the actual research, the choice of which track to run was a result of discussions between me and my supervisors, both industrial and academic. Research into development of product libraries and management of product information was agreeably one that promised to be both stimulating and relevant. With intellectual resources including personal friends, colleagues and ex-colleagues like Michael Ward, Kirti Ruikar, Isao Matsumoto, Tim Aikin and Phillip Parkin, and the project supervisors including Chimay Anumba, Ashraf El-Hamalawi, and Colin Harper, there was no doubt I had enough resource capacity to execute the project. The contributions of these people and my able wife/personal secretary, Idayat Owolabi cannot be over-emphasised.

This thesis explains why and how this project was carried out. It describes what was done and the findings from the project. However, it tells only a part of the story. The other part is the immense contributions of the people that made it possible. To these people, I am most indebted. And to God, I am thankful for grace, life and strength to finish.

Abidemi Abiodun Owolabi, September 2004

TABLE OF CONTENT

Acknowledgements .... **viii**

Abstract .... **ix**

Key Words .... **ix**

Preface .... **x**

Table of Content .... **xi**

List of Figures .... **xiii**

List of Tables .... **xiv**

1 Background to the Research **15**
1.1 Introduction 15
1.2 Background to the Research 15
1.2.1 EngD Project's History 16
1.3 Industrial Sponsor 17
1.4 Thesis Structure 20
1.5 Summary 20

2 Overall Aim and Objectives **21**
2.1 Introduction 21
2.2 Aim and Objectives 21
2.2.1 Carry Out Preliminary Studies 22
2.2.2 Evaluate Applicable Technologies 22
2.2.3 Develop Prototype Software System 23
2.2.4 Evaluate Developed System 24
2.3 Summary 24

3 Adopted Methodology **27**
3.1 Introduction 27
3.2 Methodological Considerations 27
3.2.1 The Adopted Methodology 28
3.2.2 Justification for Adopted Methodology 29
3.3 Methods and Tools Used 32
3.3.1 Preliminary Studies 32
3.3.2 Evaluation of Technologies 32
3.3.3 Systems Development 33
3.3.4 Systems Evaluation 34
3.4 Summary 34

4 Research Undertaken **37**
4.1 Introduction 37
4.2 Preliminary Studies 37
4.3 Evaluation of Technologies 38
4.4 Systems Development 40
4.5 Systems Evaluation 41
4.6 Summary 42

5 Findings and Implications **43**

5.1 Introduction....43
5.2 Research Findings....43
5.2.1 Existing Methods of Product Information Delivery....43
5.2.2 Problems with Existing Methods....44
5.2.3 Industry's Effort towards Improvement....45
5.2.4 Product Modelling the IFC Way....45
5.2.5 Issues with IFC....46
5.2.6 ComPROMISing on Standard....48
5.3 Implications/Impact of the EngD Project....50
5.3.1 Integration of Modular Product Development Project Teams....51
5.3.2 Better Quality Products....51
5.3.3 Product Development....51
5.3.4 Supply Chain Management....52
5.3.5 Modular product Development Project Implementation....52
5.3.6 The Construction Industry....52
5.4 Critical Evaluation of the EngD Project....52
5.4.1 IFC Assembly Viewer....53
5.4.2 PROMIS....53
5.5 Recommendations for further research....55
5.6 Concluding Notes....56

6 References....57

Appendix A Paper 1....61

Appendix B Paper 2....85

Appendix C Paper 3....107

Appendix D Paper 4....125

LIST OF FIGURES

Figure 1: Traditional Design and Construction Process: The over-the-wall approach... 16
Figure 2: Construction Parts Interface ... 17
Figure 3: Universal Building Model ... 18
Figure 4: Corus Organisational and Divisional Structure ... 18
Figure 5: Corus Markets and Estimated Share of Turnover ... 19
Figure 6: Corus RD&T Organisational Structure ... 20
Figure 7: EngD Work Breakdown and Outputs ... 25
Figure 8: Approaches to Systems Development Research ... 28
Figure 9: Activities in the Rational Unified Process ... 30
Figure 10: Extreme Programming Methodology ... 31
Figure 11: The Spiral Model and the Integrated Waterfall Model ... 31
Figure 12: IfcWall Definition in Ifc version 2x ... 40
Figure 13: The EngD Project Timeline ... 42
Figure 14: IfcOwnerHistory – IFC Object Auditing Attribute ... 47
Figure 15: IAI Shared Project Model ... 49
Figure 16: UBM Interaction Model ... 49
Figure 17: Current Status of PROMIS in the context of the UBM ... 50

LIST OF TABLES

Table 1: Research Map: Project Objectives, Tasks and Methods Adopted 35

1 BACKGROUND TO THE RESEARCH

1.1 INTRODUCTION

This thesis describes a research project that is focused on construction product information management. The project's background, its aims and objectives, and research methodologies adopted are discussed as well as its achievements and findings. It further highlights the implications of the findings to the project sponsor and the construction industry as a whole. This chapter introduces the research background and the project's industrial context.

1.2 BACKGROUND TO THE RESEARCH

The effects of the fragmentation of the construction industry are well recognised and documented. Client dissatisfaction, low productivity, cost and time overruns, waste and defects, and less than desirable safety records are some of the identified consequences (Harper, 2003). A decade ago, Sir Michael Latham challenged the industry to use information technology to improve operations and save up to 30% of building project costs (Latham, 1994). In 1995, the Construction Sponsorship Directorate of the then Department of Environment, with industry representatives, carried out a study, which established the information requirements for the UK construction industry. Product information was recognised as principal to all industry participants and the requirements for a library of design objects that correspond to actual building products were laid out (DoE, 1996).

Construction product information is needed throughout the life cycle of a construction project, from the feasibility study to demolition planning. Industry practitioners require complete and up-to-date information to assist in comparative evaluation and selection of products as well as for specification, installation, maintenance and disposal. Currently most product information created and used within a construction project exists in electronic form at some stage. The discrete nature of ICT tools employed in the industry means that irrespective of the *source* of the electronic product information, it is destined for a *sink* somewhere in the construction process. Whether from the supplier who produces and disseminates paper catalogues to specifiers or the contractor handing over documents to clients at completion, product information is soon converted into a form that is not easily shared or updated. The result is information is lost and the consequences are repetitions, confusion, misunderstandings, errors, delays and time-consuming litigations.

Building on Latham (1994), Sir John Egan in *Rethinking Construction* (Egan, 1998) recognised information and communication technology (ICT) as a tool for enabling improvement. Egan (1998) however cautioned that ICT is not the panacea for greater efficiency and quality in construction rather it should be regarded as a tool to support cultural and process improvements. On process improvements, Egan (1998) suggested the industry could benefit from rethinking construction as a repeated process because many buildings are essentially repeat products and the construction process is largely a repeat process from one project to another. In contrast to the existing sequential, *over-the-wall* approach (Evbuomwan & Anumba, 1998) (see Figure 1), Egan (1998) recommended an integrated project process focused on delivering value to customers through product development, project implementation, partnering the supply chain, and production of components. This Engineering Doctorate (EngD) project can be viewed as an effort aimed at providing IT tools to support the Rethinking Construction agenda for change toward a reduction in construction *time*, *cost* and *defects*.

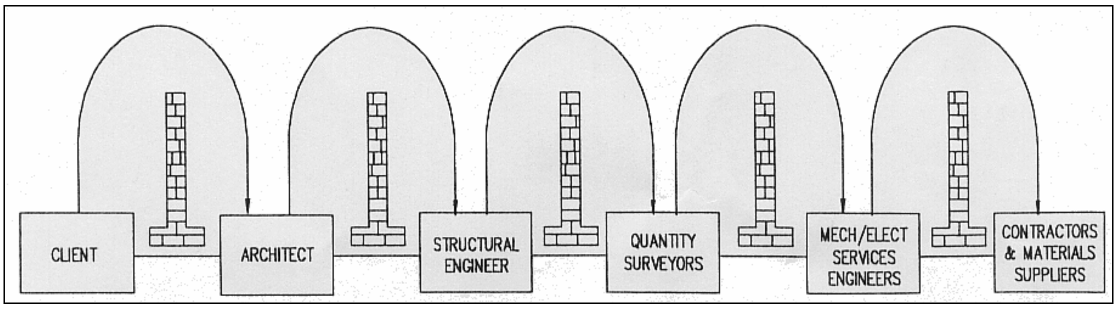

CLIENT
ARCHITECT
STRUCTURAL ENGINEER
QUANTITY SURVEYORS
MECH/ELECT SERVICES ENGINEERS
CONTRACTORS & MATERIALS SUPPLIERS

Figure 1: Traditional Design and Construction Process: The over-the-wall approach

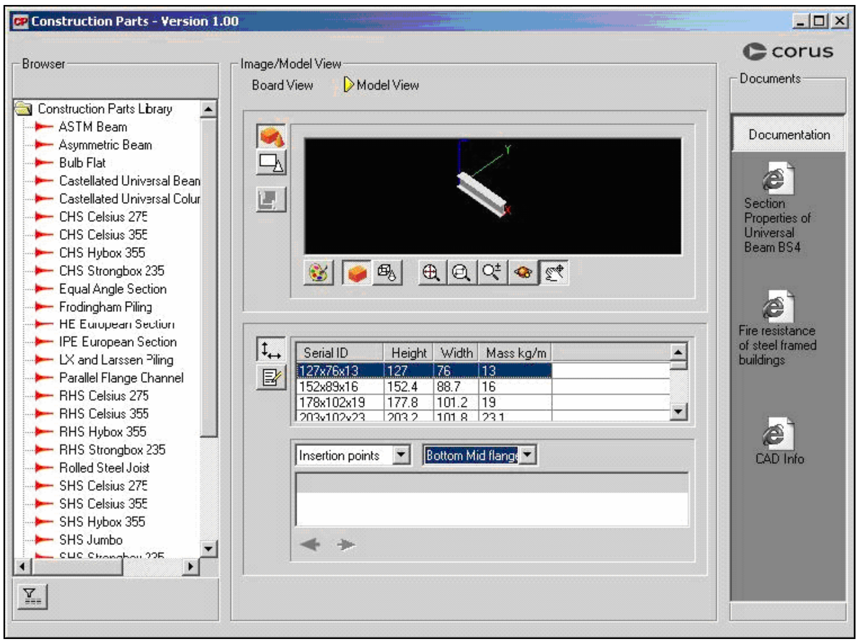
1.2.1 ENGD PROJECT'S HISTORY

The EngD project's industrial sponsor, in 2001, developed a building object library populated with real products and components for use by architects, engineers and other construction industry professionals. The software, named *Construction Parts*, was built on the idea of manufacturing *Part Libraries* and parametric object specification. It was distributed on CD for use as a standalone product catalogue or CAD library integrated with common industry CAD software. Figure 2 shows a screenshot of the application's interface. Construction Parts had some success in the construction industry but its use was limited to design and specification.

Another initiative by the sponsor was the Construction On-Line Information Network (COLIN), which was a Web-based application for managing and disseminating product information. It provided more (up-to-date) information, was easier to maintain and had better reach than Construction Parts. These developments were part of a vision to manage product information more efficiently, to disseminate the same widely, and to support the construction project teams with product information throughout the construction process.

The EngD researcher, in September 2000, started investigating current approaches to the development of product libraries with a view to improving on existing methods and tools used for the representation and presentation of building product libraries. The EngD project was then entitled *"Development of Intelligent Product Libraries for Construction"*. This was based on the proposition that product libraries should not only involve the creation and acquisition of product information but also the transfer and modification of the same to reflect new *knowledge and insight*. Preliminary investigations revealed that a more practical problem with far reaching implications to the industry is that of integration of information sources. Moreover, there was (and still is) considerable controversy surrounding the meaning of *intelligence* in software applications (Paprzycki & Abraham, 2003). The project title was thus changed to *"Development of Integrated Product Libraries for Construction"*.

The researcher joined the industrial sponsor's research team working in a similar area in September 2002 and the project was redirected at construction product life cycle information management. The project was subsequently re-titled *"Development of an Integrated Product Information Management System"* to reflect the move from product libraries to a *system* that is aimed more at achieving the best utilisation of existing sources of product information.


Construction Parts - Version 1.00
corus
Browser
Image/Model View
Board View
Model View
Documents
Documentation
Section Properties of Universal Beam BS4
Fire resistance of steel framed buildings
CAD Info
Insertion points
Bottom Mid flang

Construction Parts Library
- ASTM Beam
- Asymmetric Beam
- Bulb Flat
- Castellated Universal Bean
- Castellated Universal Colur
- CHS Celsius 275
- CHS Celsius 355
- CHS Hybox 355
- CHS Strongbox 235
- Equal Angle Section
- Frodingham Piling
- HE European Section
- IPE European Section
- LX and Larssen Piling
- Parallel Flange Channel
- RHS Celsius 275
- RHS Celsius 355
- RHS Hybox 355
- RHS Strongbox 235
- Rolled Steel Joist
- SHS Celsius 275
- SHS Celsius 355
- SHS Hybox 355
- SHS Jumbo
- SHS Strongbox 235

Serial ID	Height	Width	Mass kg/m
127x76x13	127	76	13
152x89x16	152.4	88.7	16
178x102x19	177.8	101.2	19
203x102x23	203.2	101.8	23.1

Figure 2: Construction Parts Interface

This EngD project is focused on the management of construction product information through the building life cycle and the integration of software applications commonly employed in the industry. It aims to facilitate sharing of information within the construction project team. The research work was initiated to establish requirements and to develop an IT framework and system for managing product information for construction projects. It builds on a conceptual integration framework, termed the *Universal Building Model*, and the existing research and development carried out within the sponsoring organisation's research establishment.
The concept of a Universal Building Model (UBM) was born out of the need to manage information throughout the life cycle of a building project. Based on a universally described building information model, the UBM is envisioned as a secure environment and central repository for product information managed over the lifetime of the building (Harper, 2003). Industry applications would interact with this repository to present various views of the information. Figure 3 shows the UBM concept. This EngD project contributes to the implementation of the UBM vision by providing a software implementation which was deployed and tested within a business unit in the sponsoring organisation.

1.3 INDUSTRIAL SPONSOR

The industrial sponsor, Corus, was formed in October 1999 through the merger of British Steel and *Koninklijke Hoogovens*. It is one of the world's largest metal manufacturing and service organisations employing just under 50,000 people in more than 10 countries including the United Kingdom, Netherlands, Germany, France, Norway and the USA. Corus has 20 major business units, organised into four divisions and a *Speciality Portfolio* as shown in Figure 4. As shown in Figure 5, Corus' major markets include construction, oil and gas, automotive, mechanical and electrical engineering industries.

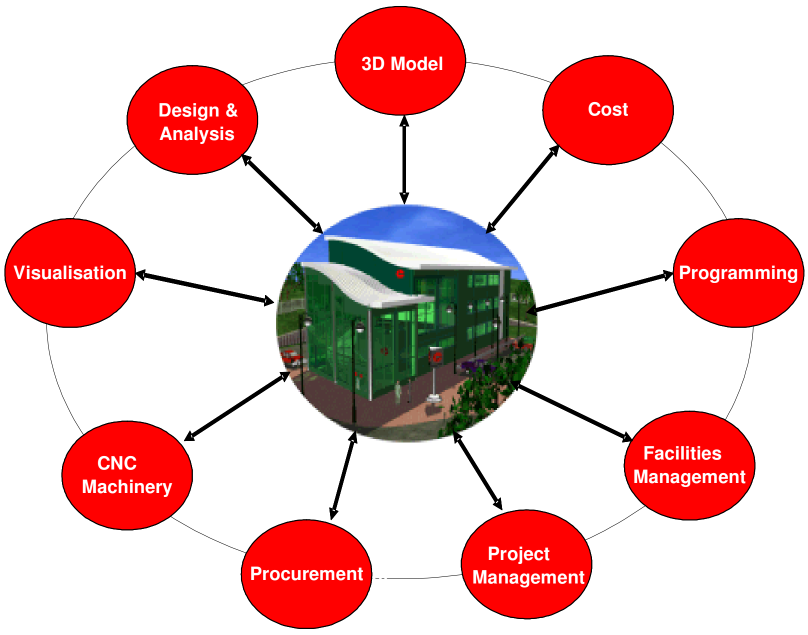
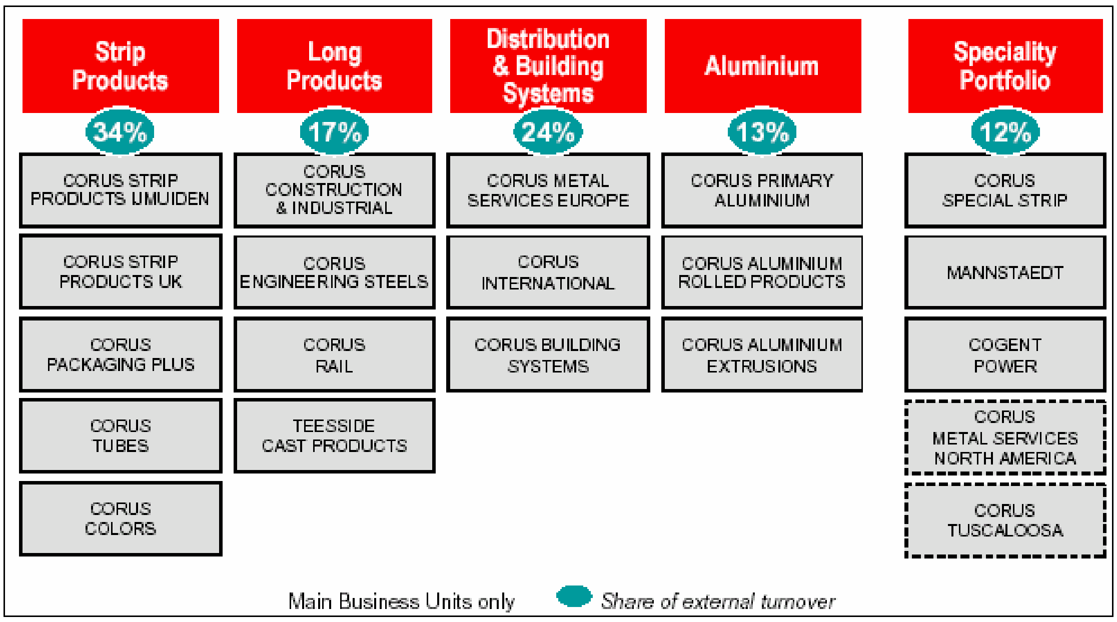


Figure 3: Universal Building Model



Strip Products 34%	Long Products 17%	Distribution & Building Systems 24%	Aluminium 13%	Speciality Portfolio 12%
CORUS STRIP PRODUCTS IJMUIDEN	CORUS CONSTRUCTION & INDUSTRIAL	CORUS METAL SERVICES EUROPE	CORUS PRIMARY ALUMINIUM	CORUS SPECIAL STRIP
CORUS STRIP PRODUCTS UK	CORUS ENGINEERING STEELS	CORUS INTERNATIONAL	CORUS ALUMINIUM ROLLED PRODUCTS	MANNSTAEDT
CORUS PACKAGING PLUS	CORUS RAIL	CORUS BUILDING SYSTEMS	CORUS ALUMINIUM EXTRUSIONS	COGENT POWER
CORUS TUBES	TEESSIDE CAST PRODUCTS			CORUS METAL SERVICES NORTH AMERICA
CORUS COLORS				CORUS TUSCALOOSA

Main Business Units only ● *Share of external turnover*

Figure 4: Corus Organisational and Divisional Structure

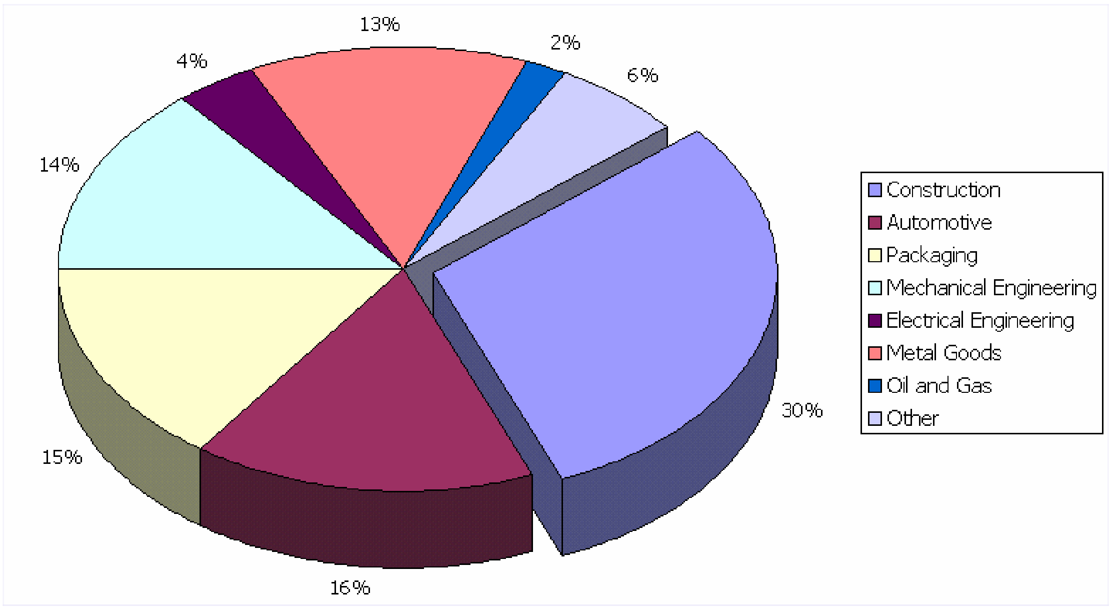


Figure 5: Corus Markets and Estimated Share of Turnover

Corus Research, Development and Technology (Corus RD&T) is an organisation, within the Corus Group, dedicated to finding the best possible metals-based solutions for Corus' major markets and continuous improvement of production processes with respect to quality, cost and protection of the environment. Corus RD&T has 14 departments divided up into Knowledge Groups with each knowledge group dedicated to a specific area of expertise. Figure 6 shows the organisational structure for Corus RD&T highlighting the Construction Technology Knowledge Group of the Construction Applications Department where this research was carried out.

As shown in Figure 5, Construction is Corus' biggest market accounting for about 30% of the total turnover. The Construction Applications department is therefore dedicated to developing innovative new products and systems for the construction sector. The Construction Technology Knowledge Group is tasked with supporting the business units trading within the construction industry, through innovative research, development and application of technology as demonstrated by this EngD project.

The software application developed as part of this EngD project is applicable to a number of business units within Corus although it is currently being deployed as the core system for a new business unit. This business unit, hereinafter referred to as the *target business unit,* designs, manufactures and delivers a range of pre-engineering building systems for affordable housing. It seeks to eliminate the most common barriers to the adoption of modular construction techniques in the building industry by:

- Offering expertise, quality, choice, flexibility and value; and
- Introducing automotive factory principles with a view to reducing waste, expense, on-site delay and building time.

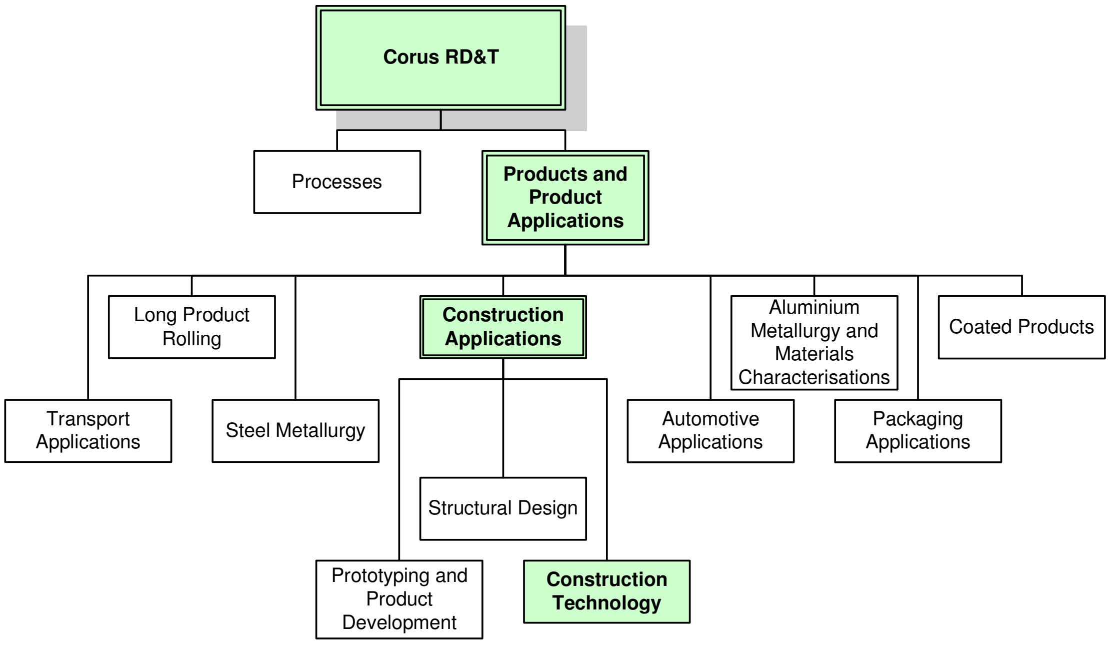


Figure 6: Corus RD&T Organisational Structure

1.4 THESIS STRUCTURE

The rest of this thesis is structured as follows:

Chapter Two contains a discussion of the EngD project's aim, its specific objectives, and task breakdown towards achieving the project objectives;

Chapter Three presents the adopted research methodology including an introduction and justification for its adoption. It also describes the methods used to achieve specific objectives;

Chapter Four describes the research undertaken with reference to the set aims and objectives; and

Chapter Five discusses the findings of the research and their impacts and implications for the project sponsor and similar organisations within the wider construction industry. It also contains a critical evaluation of the IT systems developed and makes recommendations for further development.

1.5 SUMMARY

This chapter has introduced this EngD research project. It presented the research context by highlighting the positioning of the research within the construction industry and sponsoring organisation's research efforts. This was followed by a brief justification for and a chronicle of the EngD project. Finally, the project's industrial sponsor and its specific stakeholders, including the participating research group and business unit, were introduced.

2 OVERALL AIM AND OBJECTIVES

2.1 INTRODUCTION

As discussed in Chapter 1, the title of the EngD project changed during the course of the programme to reflect new knowledge and business demands. Although the project was focused on product information throughout, the aim and some of the objectives changed with the change in project sponsor. The initial sponsor provides construction project information management as a Web-based application service provider (ASP) and sought to incorporate manufacturer product information in that Web environment. The new sponsor is a product manufacturer seeking to manage product information more effectively and to disseminate the same widely in the construction industry. While global reach and acceptability remained core requirement, there was a significant difference in the deployment environment which had an affect on the developed application architecture and scope.
Progress on the EngD project and achieved objectives were however not adversely affected by changes to the project as the changes occurred early in the project well before the commencement of major implementation tasks. A number of tasks were revisited to reflect the change in direction and project deliverables. This chapter presents the eventual aim and objectives of the EngD project as well as the justification for the objectives and the work breakdown.

2.2 AIM AND OBJECTIVES

The overarching aim of this EngD project was:

"To investigate and develop a software framework and application for integrated building product life cycle information management"

In this context, *a software framework is defined as a reusable design for all or part of a software system* (Johnson, 1992). In order to achieve the project's aim, four specific objectives were set including:

1. To investigate current methods of product information delivery and identify areas for improvement;
2. To determine the suitability of existing international product models for the implementation of an integrated construction product information management system;
3. To design and develop a prototype software system product information management and to integrate it with commonly used industry software applications; and
4. To evaluate the usability of the developed systems within the target business unit.

These four objectives can be defined more succinctly as follows:

1. To carry out preliminary studies;
2. To evaluate applicable technologies;
3. To develop prototype software system; and

4. To evaluate the developed system.

The following subsections describe these objectives including their justification, deliverables and task breakdown.

2.2.1 CARRY OUT PRELIMINARY STUDIES

The execution of this EngD project was approached from the perspective of an IT system developer seeking to automate some process in an unfamiliar subject domain. The researcher has extensive training in computer science and advanced system development and over five years experience in practical systems development. However, it was essential for the researcher to gain a good understanding of the construction industry including what constitutes building product information, existing sources, delivery methods, how the information is used in the industry, the stakeholders and their information needs. This knowledge was identified as critical to the rest of the research and four tasks were identified as necessary to achieve this objective. The identified tasks are as follows:

Task 1 - Examine product information and information sharing in the construction industry;

Task 2 - Investigate software technologies and information standards employed for the exchange and sharing of product information;

Task 3 - Review related research projects and their approaches towards improving product information delivery; and

Task 4 - Identify and recommend ways of improving the implementations of product information sources.

These initial investigation tasks were carried out through software surveys and extensive literature review including white papers, reports of feasibility studies, manuals, books, and standards.

2.2.2 EVALUATE APPLICABLE TECHNOLOGIES

The results from Objective 1 were fed into Objective 2. It was found from Task 2 that the Industry Foundation Classes (IFC) product model, defined by the International Alliance for Interoperability (IAI) is the most applicable to the construction industry. Its scope, software support and currency made the IFC the choice model for the implementation. Given that the standard is still under development it was necessary to determine its suitability, in its current state, for the planned integrated system implementation.
The target business unit undertakes construction projects with considerable involvement throughout the design and construction process. In addition, it deals with an extensive product (information) supply chain. This made it a typical environment for identifying and evaluating the technologies.
The following tasks were executed to achieve this objective:

Task 5 - Analyse product information requirements within construction projects;

Task 6 - Determine the suitability of the IFC for the planned implementation.

Task 7 - Design and develop software tool to aid understanding and implementation of IFC models; and

Task 8 - Develop a minimal information model for the planned implementation.

Task 5 was carried out through interviews, observations, and examination of product data sheets and publications. In executing Task 6, attendance at workshops and courses on IFC implementation provided the necessary understanding of the model's architecture but implementation requirements remained unclear due to inadequate software tool support. Task 7 was therefore introduced to aid the evaluation of the IFC model's suitability. From the evaluation, it was concluded that the IFC product model was not mature enough for the planned implementation. Task 8 was thus added with the additional requirement that it should be possible to integrate or convert the derived information model into the IFC in the future.
It should be noted that the change in the EngD project's sponsorship occurred while executing Tasks 5 and 6 and caused both tasks to be revisited. Task 5 was affected by the change because the initial focus was on product information requirement for the global construction industry with a view to supporting the wider product information suppliers and the construction industry practitioners or users of product information. The scope was narrowed down to look at the product information requirements with modular construction projects as a case study. The "*planned implementation*" in Task 6 was changed to reflect the new business requirements and the IFC was evaluated based on the product information requirement of the target business unit.

2.2.3 DEVELOP PROTOTYPE SOFTWARE SYSTEM

Realising this objective required prototype software development using appropriate methodologies. A number of software development methodologies were critically considered including Extreme Programming (XP), Rational Unified Process (RUP) and Reflexive Systems Development (RSD). RSD was eventually selected for the development. The next chapter discusses the considerations that led to this choice.
The following tasks were identified as necessary to achieve this objective:

Task 9 - Design and verify database to support identified information requirements;

Task 10 - Produce the functional requirements for the prototype system;

Task 11 - Produce a development plan for the prototype system;

Task 12 - Derive an architecture for the prototype system; and

Task 13 - Implement the prototype system.

The analysis of product information requirements (Task 5) revealed the need for a number of databases. These included product database, document database, supplier database, project database, user/organisation database, materials database, and media library. There was a need to design the databases and to verify them against the identified information requirements, hence Task 9. A number of potential requirements for the integrated systems have been captured during the course of the EngD project but these had to be interpreted, analysed, modelled and validated to ensure that a fairly complete set of requirements had been collected for the system. These requirements were processed in Task 10, and Task 11 produced a plan for delivering the prototype software that actualises them (including planned development phases and iterations). Task 12 executed this plan.

2.2.4 EVALUATE DEVELOPED SYSTEM

As with any software development project, it is necessary to continuously test and verify that the right system is being built right. This would normally involve a rigorous testing regime, enforcement of coding and documentation standards and management of change throughout the development process. These activities were instituted to ensure the system was built right. Close involvement of the potential end-users through regular user testing, requirements review and feedback were used to ensure that the right system was built. The tasks for this objective were therefore:

Task 14 - Carry out system testing for the implemented system;

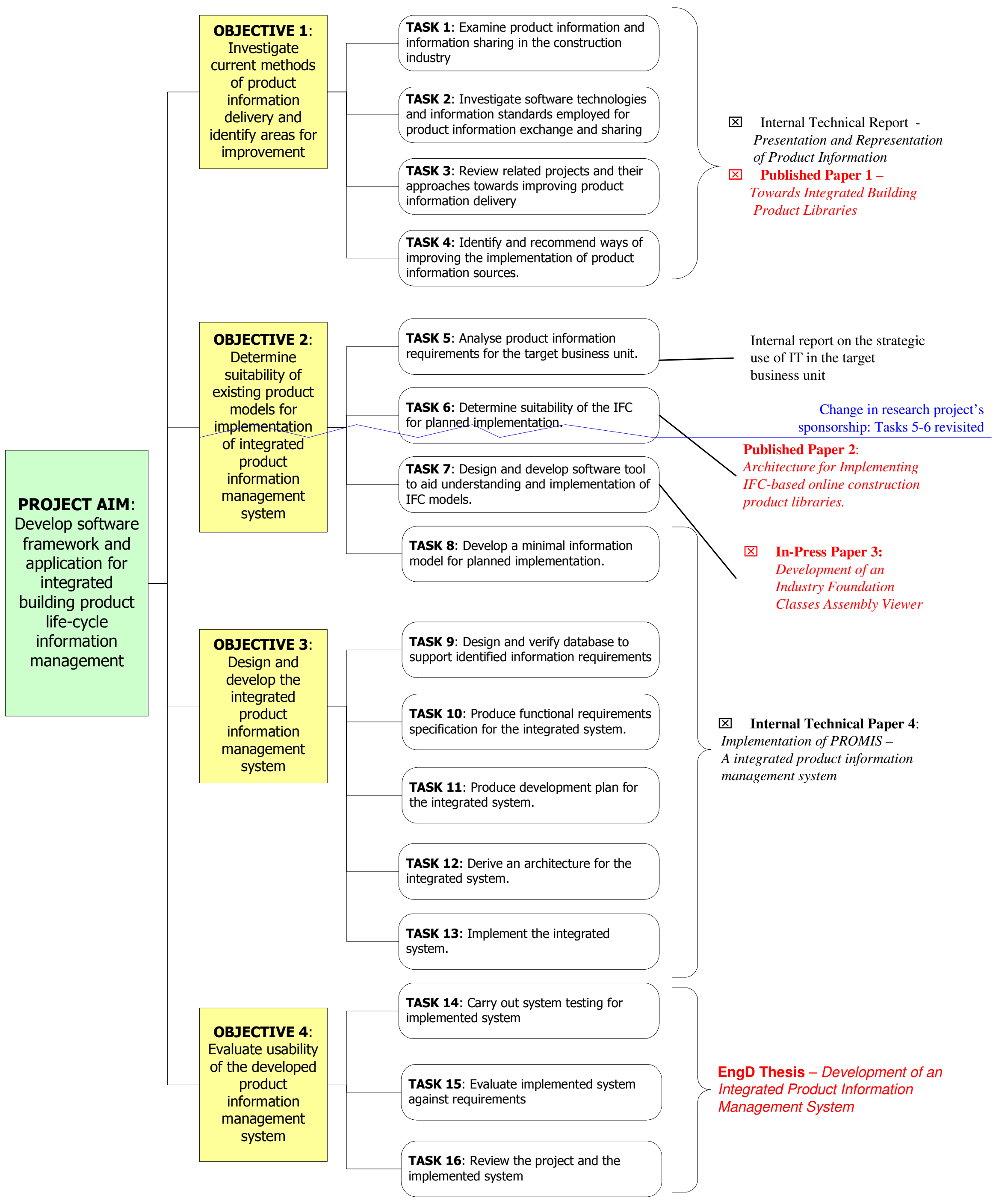
Task 15 - Evaluate the implemented system against the identified requirements; and

Task 16 - Review the EngD project and the implemented system.

It should be noted at this point that Tasks 8-15 were carried out iteratively throughout the development process with two or more tasks running concurrently at any one time. A final review workshop was organised to conclude the prototype development. This was used to identify and document outstanding issues and areas for future development.

2.3 SUMMARY

This chapter has presented the EngD project's aim and objectives as well as the work breakdown including its justification and evolution through the project. Four objectives were identified for the project. These were divided into 16 interrelated tasks some of which were carried out concurrently. Figure 7 summarises this and highlights the outputs from the tasks. Externalised outputs (mainly reports, technical papers, and the EngD thesis) are highlighted (in red), with the technical papers included in the Appendix to this thesis.



PROJECT AIM: Develop software framework and application for integrated building product life-cycle information management
OBJECTIVE 1: Investigate current methods of product information delivery and identify areas for improvement
TASK 1: Examine product information and information sharing in the construction industry
TASK 2: Investigate software technologies and information standards employed for product information exchange and sharing
TASK 3: Review related projects and their approaches towards improving product information delivery
TASK 4: Identify and recommend ways of improving the implementation of product information sources.
☒ Internal Technical Report - Presentation and Representation of Product Information
☒ Published Paper 1 – Towards Integrated Building Product Libraries
OBJECTIVE 2: Determine suitability of existing product models for implementation of integrated product information management system
TASK 5: Analyse product information requirements for the target business unit.
Internal report on the strategic use of IT in the target business unit
TASK 6: Determine suitability of the IFC for planned implementation.
Change in research project's sponsorship: Tasks 5-6 revisited
Published Paper 2: Architecture for Implementing IFC-based online construction product libraries.
TASK 7: Design and develop software tool to aid understanding and implementation of IFC models.
☒ In-Press Paper 3: Development of an Industry Foundation Classes Assembly Viewer
TASK 8: Develop a minimal information model for planned implementation.
OBJECTIVE 3: Design and develop the integrated product information management system
TASK 9: Design and verify database to support identified information requirements
TASK 10: Produce functional requirements specification for the integrated system.
TASK 11: Produce development plan for the integrated system.
TASK 12: Derive an architecture for the integrated system.
TASK 13: Implement the integrated system.
☒ Internal Technical Paper 4: Implementation of PROMIS – A integrated product information management system
OBJECTIVE 4: Evaluate usability of the developed product information management system
TASK 14: Carry out system testing for implemented system
TASK 15: Evaluate implemented system against requirements
TASK 16: Review the project and the implemented system
EngD Thesis – Development of an Integrated Product Information Management System

Figure 7: EngD Work Breakdown and Outputs

3 ADOPTED METHODOLOGY

3.1 INTRODUCTION

This EngD research project can be described as an information systems development project. Systems development is a specific kind of social activity which aims to change organisations through the use of information technology (IT). It is an important practice and research area that takes place under different economical, organisational, and cultural conditions (Dittrich, 2000). Systems development activities include analysis, design, programming, implementation, and maintenance, as well as project management, quality assurance and software process improvement (Mathiassen, 1998). The diversity of activities and varied contexts in which this type of research is usually undertaken suggests that some careful considerations are required in choosing and applying a research methodology.
Clarke (2000) defined research methodology as the study of research methods; research method as the manner (steps and procedures) in which a particular research task is undertaken; and research technique as an approach or tool-and-its-use whereby data is gathered and analysed, and inferences are drawn. More specifically, Avison & Fitzgerald (2003) define systems development methodology as:

> *"a recommended means to achieve the development, or part of the development, of information systems based on a set of rationales and an underlying philosophy that supports, justifies, and makes coherent such a recommendation for a particular context."*

This chapter discusses the methodological considerations for this EngD project as well as the adopted methodology and methods used to achieve the project objectives.

3.2 METHODOLOGICAL CONSIDERATIONS

Research methods can be categorised in various way. One of the most common distinctions is between quantitative and qualitative methodologies. This distinction is based on the type of data collected and tools used. Quantitative research methods (sometimes referred to as scientific or posivitist) originates from natural sciences and uses empirical approaches including survey methods, laboratory experiments, formal methods (e.g. econometrics) and numerical methods such as mathematical modelling. On the other hand, qualitative research (sometimes called interpretivist or humanistic) originates from social sciences and uses qualitative techniques including action research, case study research and ethnography. Here, the researcher's impressions and reactions play a key role.
Since systems development is a social activity, one could argue that social science research approaches would be most appropriate. There is however several factors that makes systems development rather peculiar and militates against a consensus in methods applied. Some of these factors include the experience and competence of the development group, the considered information systems, the dynamics of the objectives, and the social and technological environments (Lyytinen, 1987).
A variety of research methods has been explored by systems development researchers each being appropriate for different aspects of research study depending on the domain and philosophical positions of the researcher. Nunamaker et al. (1991) categorises a

number of these approaches as combinations of action research, experiments and practice studies. Figure 8 presents this as the multi-method approach.

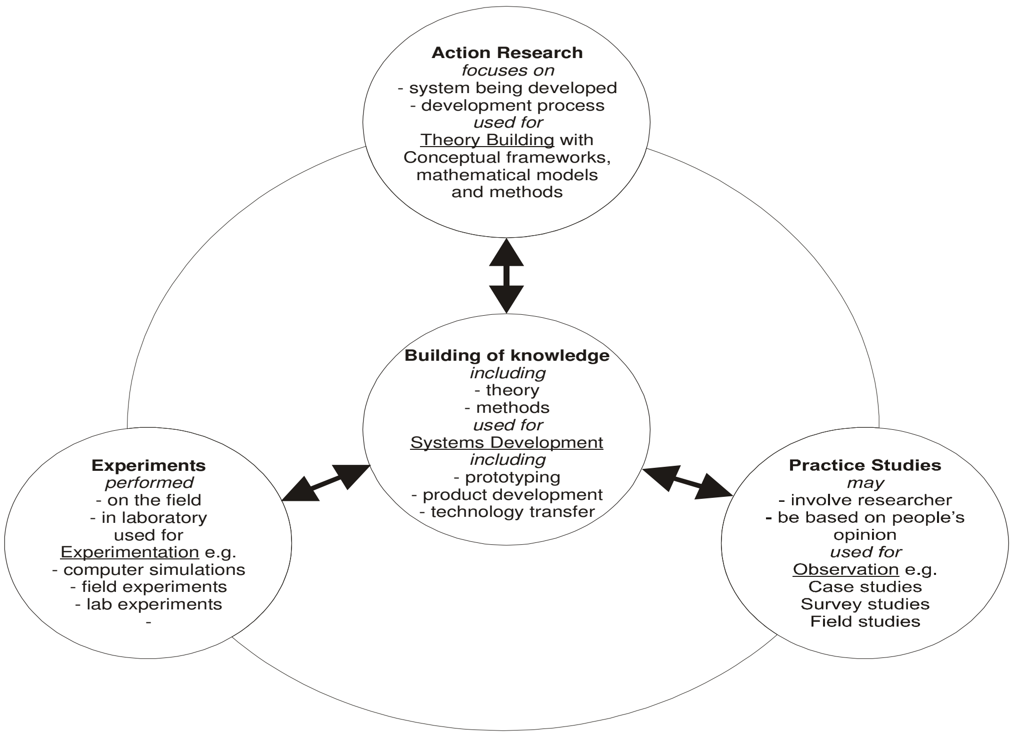


Figure 8: Approaches to Systems Development Research (adapted from Nunamaker et al. (1991))

Other researchers have suggested multi-method, or *triangulation*, approaches for systems development. Triangulation is defined as "the combination of methodologies in the study of the same phenomena" (Ghauri & Gronhaug, 2002; Gable, 1994). Another research typology that combines approaches was proposed by Wilson (2002). Wilson (2002) suggested that *observation is the starting point of all research* and derived other classifications from that root. Suggested classifications included *direct* or *indirect* observations which may be undertaken with *imposed* or *emergent* structures.

3.2.1 THE ADOPTED METHODOLOGY

This EngD project adopts a multi-method approach called *Reflective Systems Development* (RSD) (Mathiassen, 1998; Schon, 1995). RSD is a subjective approach which intertwines research and practice. It is based on the notion that information systems researchers should engage in reflections and dialogues to generate the necessary insights into the situation at hand. Reflection is the practice of periodically stepping back to ponder on the actions of oneself and others in one's immediate environment (Raelin, 2001). Heiskanen (2004) suggested three objects of reflection, including contents (how a practical problem was solved), process (the procedure and sequences of events) and premise (presuppositions attending to the problem), and three timings of reflection, including anticipatory, contemporaneous, or retrospective. The goal of RSD is to facilitate the development of computer-based information system as part of an on-going transformation of organisation and society through a process of *reflection-in-action*.

3.2.1.1 Reflection-in-Action

Problematic situations involving complexity, uncertainty, instability, uniqueness and value-conflict are quite common in systems development and vary depending on the type of project. Reflection-in-action is a process for dealing with these situations effectively through problem-framing, implementation and improvisation activities. Beyond theories and methods, reflection-in-action emphasises the role of knowledge and its relation to actions in systems development practice. It suggests a combination of analytical (specifications) and experimental (prototyping) approaches on the ground that:

> *"Relying on an analytical mode of operation to reduce complexity introduces new sources of uncertainty requiring experimental countermeasures. Correspondingly, relying on an experimental mode of operation to reduce uncertainty introduces new sources of complexity requiring analytical countermeasures. (Mathiassen & Stage, 1992)"*

Mathiassen et al. (1995) report on the comparative success of this mixed approach through some qualitative experiments. RSD recommends the use of diaries as an empirical research technique for management of information systems development (Jepsen et al., 1989). Diaries would contain notes on participants' reflections including understanding of the situations, problems, and conflicts as well as options, events, decisions, and future plans.

3.2.2 JUSTIFICATION FOR ADOPTED METHODOLOGY

More than a dozen systems development methodologies are available today and deciding on which to adopt is not a trivial task. Avison & Fitzgerald (2003) identifies more than twenty different systems development methodologies and their variations, and suggests seven elements for comparing them including their philosophy, model, techniques and tools, scope, outputs, practice and product. For this project, most of these methodologies were easily eliminated on the basis of their application domain, systems paradigm, objectives of the methodology and target systems. Here are some examples:

Application Domain – Knowledge Analysis and Design System (KADS) is for expert systems development; Welti ERP and Accelerated SAP are for Enterprise Resource Planning (ERP) projects; and Web Information Systems Development Methodology (WISDM) is for Web-based applications.

Systems Paradigm – Effective Technical and Human Implementation of Computer-Based Systems (ETHICS) adopt a socio-technical view. With ETHICS, requirements engineering takes far too long and requires considerable resources.

Objectives – Structured Analysis, Design and Implementation of Information Systems (STRADIS) is too general purpose with objectives beyond computerised systems; and ETHICS seek to enhance the job satisfaction of users.

Target Systems – STRADIS, Merise, and Structured Systems Analysis and Design Method (SSADM) are all primarily for large organisations and systems.

A detailed description of the methodologies mentioned above can be found in Avison & Fitzgerald (2003). The main consideration was eventually between three methodologies:

Rational Unified Process (RUP) (Kruchten, 2000), Extreme Programming (XP) (Beck, 1999) and the less well-known RSD.

RUP is a commercial product of Rational Software (*www.rational.com*). It is described as "use-case driven, architecture-centric, iterative and incremental". Its iterative approach was appealing to the researcher because it helps to minimise risks and reduce the chance that the system would not meet its goals at the conclusion. It employs a number of cycles (or iterations) each consisting of four phases: inception, elaboration, construction and transition as shown in Figure 9. This methodology was initially adopted but later abandoned because it was too complex and required extensive production and maintenance of documents - an activity which can be rather time consuming, resource intensive, and distracting.

A more agile development methodology was sought in the form of Extreme Programming (XP) (West, 2000). XP is a lightweight methodology with values for "simplicity, communication, feedback and courage." Its phases include planning, designing, coding, and testing as shown in Figure 10. With XP, documentation is of little importance and the software is the primary goal. It requires pair programming (i.e. two people working together at a single computer) to compensate for the documentation deficiencies. This fundamental requirement made the methodology unsuitable for this project.

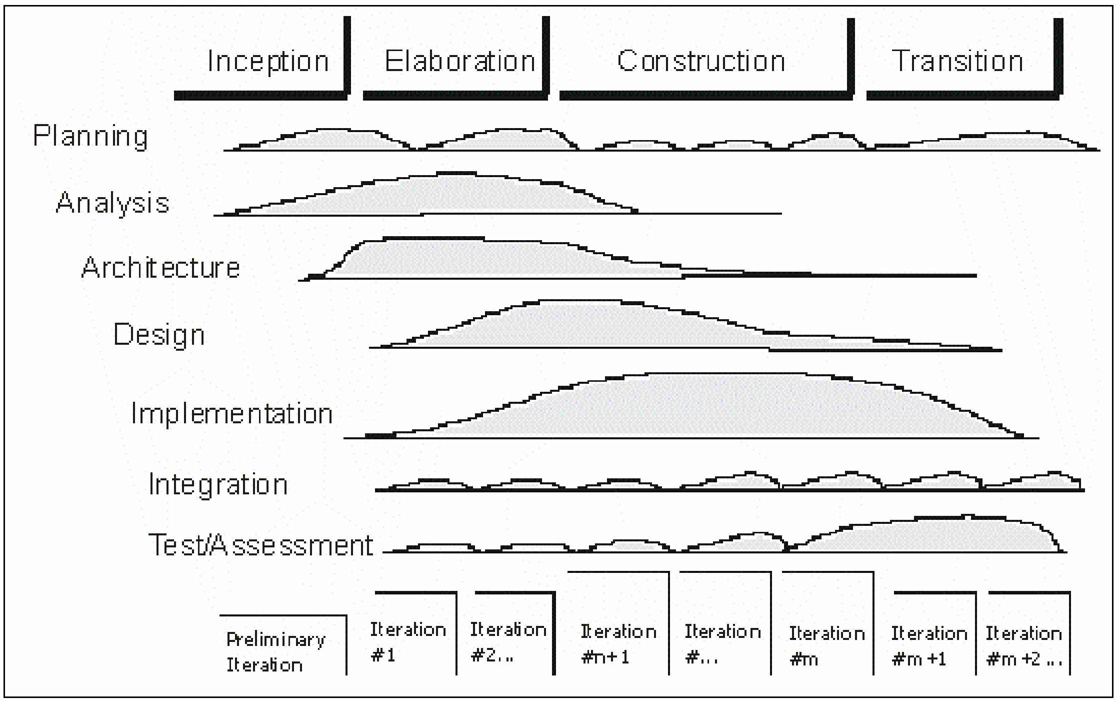


Figure 9: Activities in the Rational Unified Process (adapted from Kruchten (1996))

With flexibility, ease of use and learning as criteria, RSD was chosen by a process of elimination. Its application domain includes all information systems, it employs an applied science paradigm, its objective is specifically to develop computerised information systems, and its target system includes small teams and systems. As a way of managing project risk, it embraces iterations and evolution through the Spiral Model (Boehm, 1988).

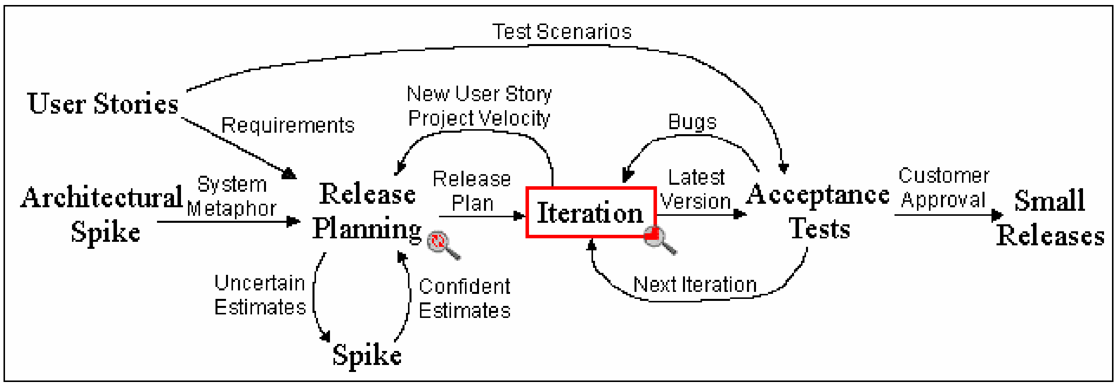


Figure 10: Extreme Programming Methodology (adapted from West (2000))

The Spiral Model is a systematic way of mixing specifying and prototyping approaches with a view to managing project risks, particularly complexity and uncertainty. As shown in Figure 11, the cycles comprise spiral movements with the radial dimension representing the cumulative costs of the activities carried out and the angular dimension representing the progress made in each cycle. Each cycle of the spiral starts with a re-evaluation of risks which may lead to some specifying or prototyping and subsequently to system development. The development tasks were executed with the traditional *waterfall* model also shown in Figure 11.

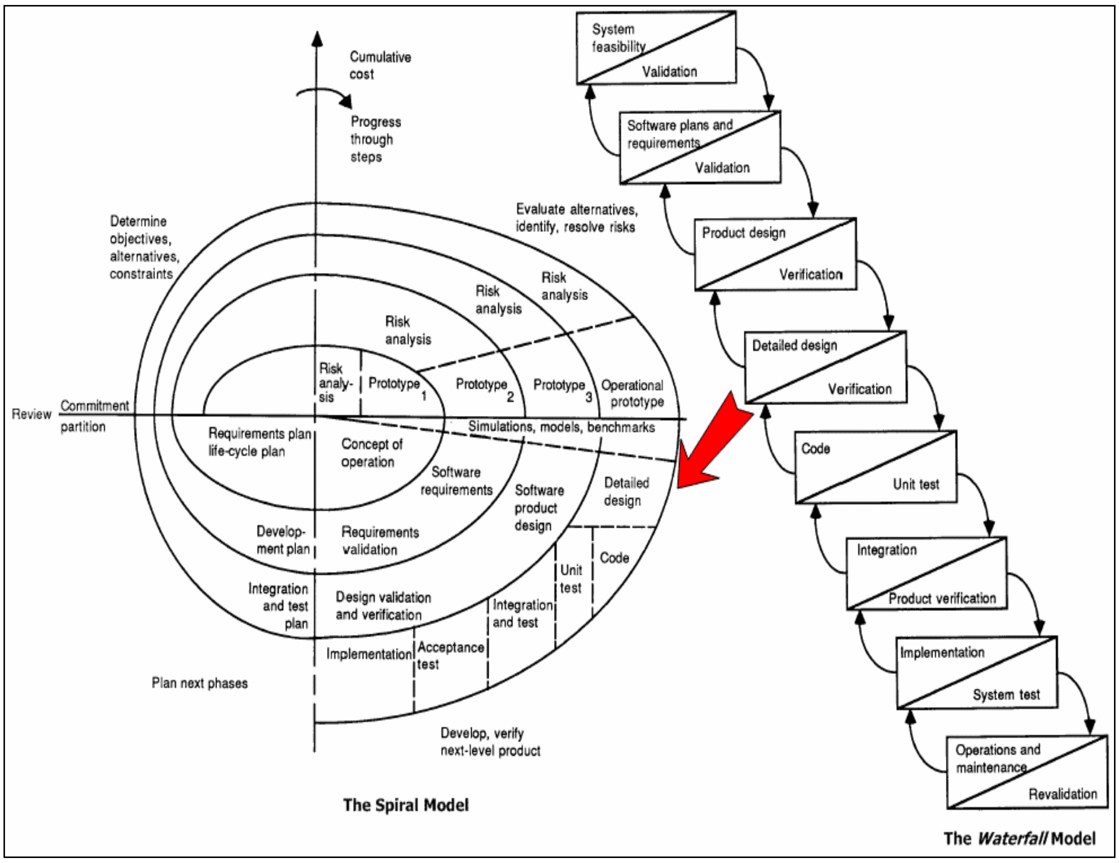


Figure 11: The Spiral Model and the Integrated Waterfall Model

3.3 METHODS AND TOOLS USED

The purpose of this section is to discuss the methods and tools that were applied to achieve the project objectives as discussed in Chapter 2. These are reiterated here:

1. Preliminary studies;
2. Evaluation of Technologies;
3. Systems Development; and
4. Systems Evaluation.

3.3.1 PRELIMINARY STUDIES

The author, at the commencement of the programme, had expert knowledge of computer systems but no previous knowledge of the construction industry. Hence, the preliminary study included obtaining working knowledge of the global and (the peculiarities of) the UK construction industry.
Arguably, observation is the foundation for all information systems research (Wilson, 2002). This objective was achieved through observation either directly, with the author situated in relevant environments, or indirectly through archival analysis of documents. The direct observation took the form of an ethnographic study which was carried out in a firm that delivers technology for Web-based project collaboration.

3.3.2 EVALUATION OF TECHNOLOGIES

The process for identifying and accessing technologies for systems implementation is described by Wesley-Tanaskovic et al. (1992) as consisting of four phases:

1. *Scouting* and identification of requirements specifications for candidate technologies;
2. *Authoritative information documentation* concerning technological, economic, and societal need for, and feasibility of, the technologies;
3. *Assessment* and evaluation of the technologies; and
4. *Selection* of appropriate technologies for initial development and implementation.

Establishing the requirements for product information management was initially carried out for the construction industry through informal interviews with industry professionals and extensive literature review. The report of the scouting, information documentation, assessment and selection of technologies is documented in Paper 2 (see the Appendix). Some of the requirements for the target business unit had been established by the time the author joined the sponsoring organisation. The other requirements were elicited as the research project progressed through archival analysis, informal interviews and ethnographic studies looking at how product information is used within the organisation. The subsequent scouting, documentation, assessment and selection exercise was thus targeted to the specific planned implementation.
It was necessary to develop a software tool during the assessment exercise. The choice of information exchange standard is an architectural decision and very critical to the development project hence no expense was spared in ensuring the right decision was made. The author attended a workshop on system implementation using the IFC

standard and an intensive training course on the use of a leading software that is supposedly well-suited to its implementation. There were, however, outstanding knowledge gaps that hampered a definitive assessment of the technology. The development of a software tool to fill in the gaps and aid understanding of the technology was therefore commissioned. As the requirement specification was clear and the outcome predictable, the development process adopted for this task was the *waterfall* aspect of the Spiral Model. Entity-Relationship (ER) modelling, supported by software tools including Visual UML™ and Microsoft Visio™, was used to capture the information requirements of the target business unit. The information requirements shaped the evaluation and eventual selection of technologies.

3.3.3 SYSTEMS DEVELOPMENT

This EngD project proposed the development of a new software system with a considerable degree of uncertainty. The typical construction user is sceptical of new technologies. The proposed system would require a major shift in current approaches to the use and management of product information. A number of new, volatile and competitive technologies was required, and there were legal implications to be considered. RSD provided a useful framework for managing these risks as well as complexity. Cultural issues and user scepticism were addressed by gaining user buy-in early on and involving the potential users throughout the development process. Frequent development cycles and project meetings to clarify requirements and obtain feedback were undertaken. Not only did the development team reflect before, during and after each cycle, the users were also invited to do the same as they used the unfinished system. Prototyping and specifying were key parts of each cycle.

3.3.3.1 Prototyping

Prototyping is widely recognised as a potent and cost effective approach to systems development that can lead to the production of systems with a high level of user-acceptability more rapidly and more cheaply (Law & Longworth, 1987; Lejk & Deeks, 2002). It is particularly useful when system concepts and requirements are unclear; application is fairly complex, users communities are geographically-dispersed; and rapid response to external changes are required. These factors were present in varying degrees throughout the development task making it a suitable candidate for the prototyping approach. A number of prototypes were developed during the course of the project to elicit the social and organisational requirements and to test new technologies.
This prototyping approach was combined effectively with ethnographic studies, project meetings, coffee room discussions and interviews. It provided very useful information about how people actually work rather than how process definitions suggested they worked and more importantly, it provided the development team with priceless information about how the application could be improved to ensure user satisfaction. Conflicts detected between the different communities of users were resolved in project meetings.

3.3.3.2 Specifying

Specifying is an analytical approach to handling systems design complexity through abstraction and decomposition. Abstractions provide clarity through selection and structuring of relevant information to aid communication with users, design and implementation, measurement of progress, and division of labour among systems developers. The use of specifications was rather limited on this development project.

Where produced, the specifications used free-text and diagrams, and were just detailed enough to aid communication between the development teams. This approach was adopted for the following reasons:

- Requirements were rarely understood enough to specify unambiguously;
- The requirements changed consistently during the development process;
- The implementation process revealed more information which invalidated previous design decisions; and
- Many design decisions were made based on intuition rather than *good* reasons to facilitate experimentation.

The general limitations of specifying are well outlined in Parnas & Clements (1986) and Mathiassen et al. (1995). It is well recognised that specifications play an important role in verification and ensuring future continuity of development projects. Concerted effort was therefore directed at producing and revising the specifications *after* the program codes were largely developed.

3.3.4 SYSTEMS EVALUATION

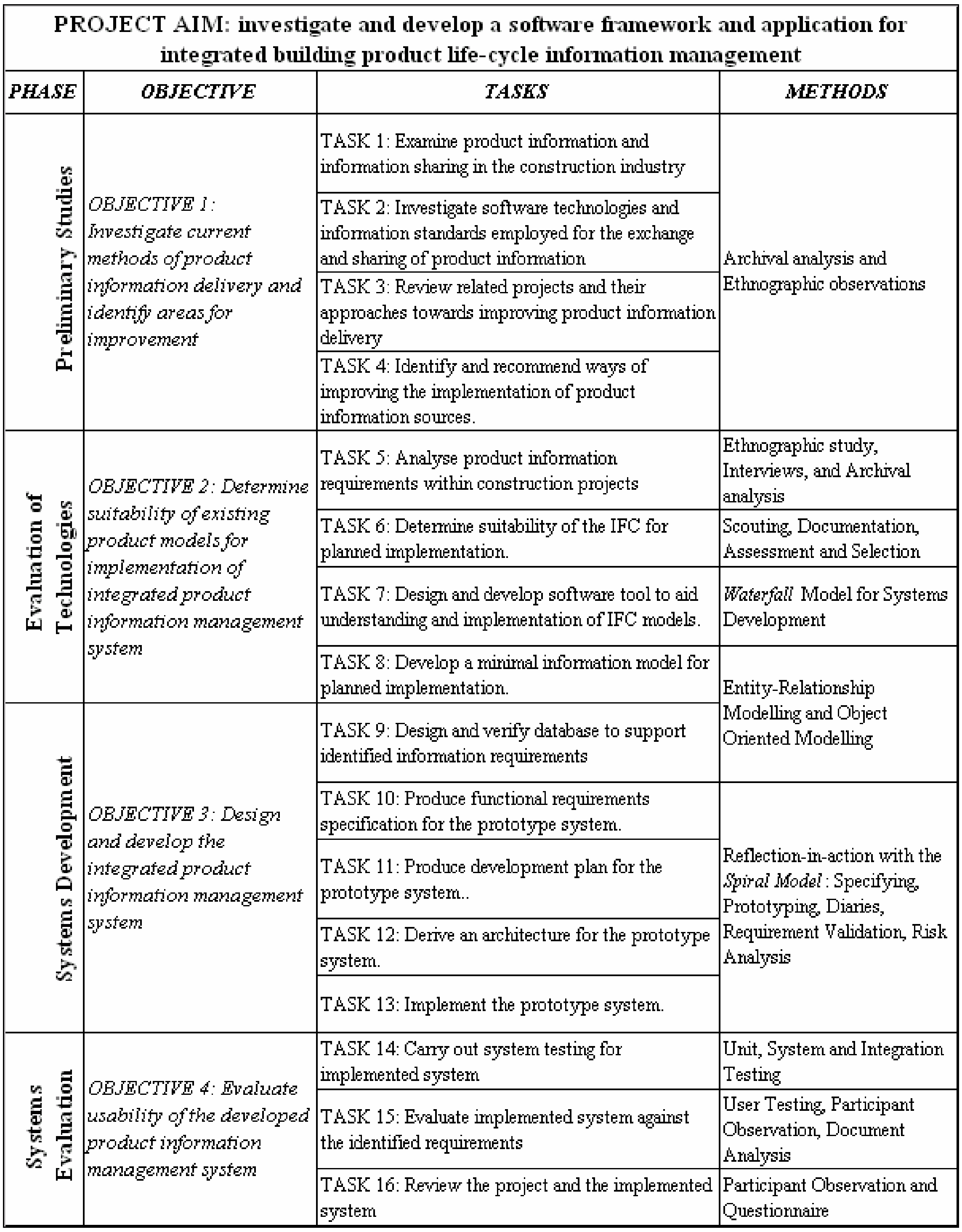
The evaluation of the developed system was an integral task of the development process. The system was evaluated periodically by potential end-users selected from within the target business unit and the industrial sponsor's research establishment. It was evaluated for functionality, robustness, ease of use, and ease of learning. Comments, concerns and errors were either logged remotely or communicated directly by various means. This feedback was analysed and resolved or noted for further actions by an appropriate developer who may need to interview the affected user(s) for further clarification. Logging facilities were also used to record and manage the development diary entries.

Regularly, the researcher observed the users while using the system in order to identify breakdowns or conflicts between what the users assumed or expected to happen and what actually happened. This participant observation provided useful information particularly for the user interface design. The final evaluation was carried out in similar fashion with participant observation.

3.4 SUMMARY

This chapter introduced the philosophical approach of the adopted methodology, the Reflective Systems Development (RSD) methodology. It provided a justification for the adoption of RSD in contrast to other seemingly applicable methodologies. A discussion of processes and specific methods used were also presented along with the considerations that influenced their selection. This discussion was structured according to the project objectives identified in Chapter 2. Table 1 summarises the methods applied for specific tasks toward achieving each of the project objectives.

Table 1: Research Map: Project Objectives, Tasks and Methods Adopted



PROJECT AIM: investigate and develop a software framework and application for integrated building product life-cycle information management

PHASE	*OBJECTIVE*	*TASKS*	*METHODS*
Preliminary Studies	*OBJECTIVE 1: Investigate current methods of product information delivery and identify areas for improvement*	TASK 1: Examine product information and information sharing in the construction industry	Archival analysis and Ethnographic observations
		TASK 2: Investigate software technologies and information standards employed for the exchange and sharing of product information	
		TASK 3: Review related projects and their approaches towards improving product information delivery	
		TASK 4: Identify and recommend ways of improving the implementation of product information sources.	
Evaluation of Technologies	*OBJECTIVE 2: Determine suitability of existing product models for implementation of integrated product information management system*	TASK 5: Analyse product information requirements within construction projects	Ethnographic study, Interviews, and Archival analysis
		TASK 6: Determine suitability of the IFC for planned implementation.	Scouting, Documentation, Assessment and Selection
		TASK 7: Design and develop software tool to aid understanding and implementation of IFC models.	*Waterfall* Model for Systems Development
		TASK 8: Develop a minimal information model for planned implementation.	Entity-Relationship Modelling and Object Oriented Modelling
Systems Development	*OBJECTIVE 3: Design and develop the integrated product information management system*	TASK 9: Design and verify database to support identified information requirements	
		TASK 10: Produce functional requirements specification for the prototype system.	Reflection-in-action with the *Spiral Model*: Specifying, Prototyping, Diaries, Requirement Validation, Risk Analysis
		TASK 11: Produce development plan for the prototype system..	
		TASK 12: Derive an architecture for the prototype system.	
		TASK 13: Implement the prototype system.	
Systems Evaluation	*OBJECTIVE 4: Evaluate usability of the developed product information management system*	TASK 14: Carry out system testing for implemented system	Unit, System and Integration Testing
		TASK 15: Evaluate implemented system against the identified requirements	User Testing, Participant Observation, Document Analysis
		TASK 16: Review the project and the implemented system	Participant Observation and Questionnaire

4 RESEARCH UNDERTAKEN

4.1 INTRODUCTION

Previous chapters of this thesis have proffered answers to the questions of *why* and *how* the EngD project was carried out. An introduction to the project was presented in Chapter 1. Chapter 2 highlighted the aims and objectives and Chapter 3 discussed the methodological approach to the project. This chapter describes the actual research carried out. It highlights the activities involved in achieving each of the set objectives and the overall project aim. The discussion is organised according to the project objectives.

4.2 PRELIMINARY STUDIES

The preliminary studies carried out provided the author with adequate knowledge of the use of product information within the global and UK construction industries, as well as the technologies and approaches adopted by relevant past and on-going research projects.

The first three months of the programme was dedicated to information gathering through attendance at lectures and workshops and extensive literature search and review. This afforded the author an historical appreciation of the industry and provided information on recent cultural and technological trends in the industry. The author observed that there is considerable impetus, within the construction industry, toward better integration of the construction project team. Published documents on relevant past and ongoing research projects were reviewed. Some of the researchers involved were contacted through emails and where necessary informal telephone interviews were conducted to clarify issues.

It was observed that the industry was approaching electronic information sharing largely through shared databases, distributed objects and file-based or dynamic application-based data exchange. File-based exchange and shared databases, accessible via the Internet, were found to be in commercial, production-level deployment. The prevalent method for information exchange in the industry is file transfer and translations but this method is fraught with inefficiencies. Capabilities of distributed objects and dynamic (application) data exchange, which allowed software components to work together, have been demonstrated for construction integration although commercial implementations are non-existent.

In a bid to study the intricacies of the information structures and exchanges in shared project databases, a *background* ethnographical observation of the processes and supporting technologies was undertaken. The complexity of information exchanges on construction projects of varying sizes was observed by examining how a commercial Web-based project collaboration tool was utilised by members of construction project teams. These observations were carried out over sixteen months during which the author was also involved with the development of a software tool for batched uploading (or publishing) of project information to the shared database. This involvement provided an even better understanding of information flow within construction project teams.

Following an appreciation of information exchanges in the industry, the next stage of the work focused on examining product information in particular. Various sources including printed catalogues and their electronic versions in CD-ROMs and DVDs were acquired and studied for information representation and presentation. Usability of Web-enabled product information sources was examined from the perspectives of both consumers and producers of product information.

It became quite apparent that there was a lack of agreement in product information representation and this greatly hampered reuse and exchange of the information. Archival document analysis was therefore carried out to find out how the problem was being addressed in the construction industry and other industries, especially manufacturing. The study found that manufacturing industry tackled the problem through a global standardisation effort, the STEP standard (ISO, 1994) and the construction industry was taking similar route through a standardisation body, the International Alliance for Interoperability (IAI). The IAI defines data structures, called Industry Foundation Classes (IFCs) to support a shared project model for data sharing across applications for the AEC/FM industry. A face-to-face interview was held with the Technical Coordinator for IAI UK Chapter to obtain his views on the feasibility and readiness of the standard for construction integration. He was of the view that the standard is ready for a number of application areas and the IAI is actively working on the development of specification and property sets for electronic product library applications.

The findings from these studies and interactions were collated, analysed and documented in an internal technical report which was later distilled into Paper 1 (see the Appendix). A summary of the main findings is provided in the next chapter.

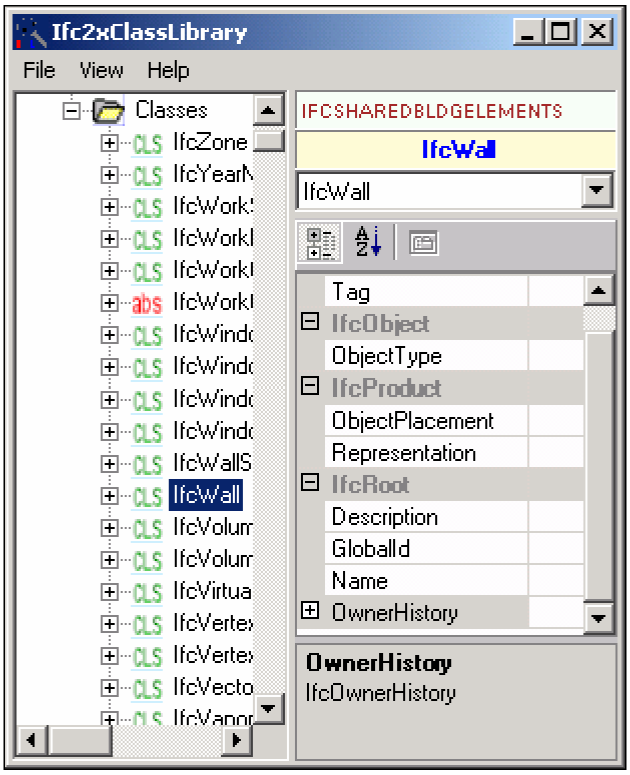
4.3 EVALUATION OF TECHNOLOGIES

During the preliminary studies, the problems of product information delivery and sharing became clearer and candidate technologies also began to emerge. The entire construction industry was the initial focus of the research and the search for a solution was constrained by that scope. In addition, it was envisaged that the solution would be deployed as an application service alongside the Web-based shared project database system mentioned in the previous section. An ethnographic study of product information acquisition and transfer between construction project teams on different projects was carried out to establish the product information requirements within construction projects. Published documents were analysis and informal interviews were carried out with industry professionals to validate the observations from the ethnographic study.

The requirements obtained aided in technology scouting whereby candidate technologies for information representation and application integration were identified and their capabilities were documented. The assessment of these capabilities led to the selection of the IFC model for information representation and Microsoft.NET development environment for application integration. A vision document was subsequently produced detailing the high-level requirements and features of an industry-wide product library application. The document identified the stakeholders and target users and the capabilities they require with justifications for the needs.

As part of the assessment exercise of the IFC, which is defined with EXPRESS information modelling language, software tools for working with the EXPRESS information model were acquired and evaluated. Two key developers, one from each, of the two leading complete EXPRESS solutions providers, were interviewed to gain a better understanding of their application architectures and peculiarities. IFC implementation workshops and EXPRESS implementation training course were

attended. Two project managers in the defence industry, who were also implementing or considering the STEP (ISO 10303) EXPRESS-based information model, were also interviewed to find out about their experience including the risks and challenges they were facing on the projects.
The study of product information requirement on construction projects produced five usage scenarios or levels of interaction which an integrated product library implementation need to support to satisfy industry requirements. These are documented in Paper 2 (see the Appendix) along with the software architecture to support all the levels of interaction. The derived architecture specified service-oriented implementation in a multi-tier client/server environment and addressed identified presentation, integration, persistence, and cooperation and configuration issues. Three candidate development environments were identified including Microsoft .NET, Enterprise Java and CORBA Component Model (CCM) and Microsoft .NET was selected. The architecture was however not implemented because of the change in project sponsorship.
The change in the project's sponsor resulted in a switch to a case study approach. The project was refocused to consider product information requirements for a specific business unit and to evaluate applicable technologies for the system implementation.
Egan (1998) suggested that the construction industry could learn from the manufacturing industry and that modularisation could deliver considerable benefits. A modular, manufacturing-type, construction business was selected as the target business unit as this would derive the most benefit because of the repeatability of its processes and the nature of its products.
As before, the IFC was the prime candidate for the information models and XML for data exchange. ifcXML, a subset of XML for IFC development had been developed making the IFC even more appealing. Evaluating the IFC was however more tasking than envisaged principally because of inadequate software tool support. This prompted the author to examine how the information model could be made more understandable and accessible to developers. The requirements for a software tool to bridge the gap between EXPRESS information modellers and software developers was set out and a development project was started to actualise the requirements. The end result of this development was the IFC Assembly Viewer software described in Paper 3 (see the Appendix).
The assessment exercise aided by the IFC Assembly Viewer, led to the conclusion that the IFCs were not sufficiently mature to cater for the needs of the target business unit. There were suggestions that the IFC *property sets* could be used to augment the standard for the specific use. For example, the basic *IfcWall* definition shown in Figure 12 could serve as the base for some detailed wall description. This would however require considerable development and could possibly lead to the development of another non-standard information model. In addition, managing the information overhead would require substantial development effort without any visible value added to the target business unit. It was therefore decided to discontinue the implementation of the IFC and to develop a minimal but incremental information model to support the planned development for the target business unit. It was also agreed that the development should be loosely-coupled to the information model to enable quick changeover when the IFCs become suitable. The development of the information model was aided by the Visual UML™ modelling tool which enabled automatic generation of database designs, program source code and design documentations.



OwnerHistory
IfcOwnerHistory

Figure 12: IfcWall Definition in Ifc version 2x

4.4 SYSTEMS DEVELOPMENT

The systems development task was 3-phronged was divided into three work streams namely Prototyping, Development and Release. The prototyping work stream was used to clarify requirements, test assumptions, analyse risks, evaluate technologies and identify potential solutions as prescribed by the Spiral Model of the adopted RSD methodology. The outputs from the prototype stream were reverse engineered to produce the detailed design specification which were translated into application code and integrated into the development work stream using the Waterfall model as discussed in Chapter 3. The application design changes during the development process hence final design documentation is produced through reverse engineering before the application is released for user testing. Each release application is transferred into the Release work stream for minor bug fixes.
This process has the potential to create *phenomena of parallel changes* (Perry et al., 1998), and demanded an efficient management of the change process to avoid problems such as lost changes, reappearing bugs, application instability, etc. The change process was managed through application source code control with *branching and merging*. Microsoft Visual SourceSafe™ was used to manage the code branching, which allowed development to take place along more than one *path* for a particular file or directory, and Araxis Merge™ was used to merge changes to files and directories within the different work streams.
The development work on each stream was categorised according to three major functionalities that the development seeks to implement. There were:

1. Management of product information through Web-based product libraries;

2. Management and integration of product information within construction projects through project databases; and
3. Integration of product information sources and project information management through information brokerage.

These three functionalities were implemented in three subsystems, ProductWeb, ModelManager.NET and P2Service respectively, and encapsulated in the **PRO**duct **M**anagement **I**nformation **S**ystem, (PROMIS). The PROMIS architecture is described in detail in Paper 4 (see the Appendix). For ease of maintenance, the components of each of the subsystems were functionally decomposed into *infrastructure*, *application domain*, and *user interface*. *Infrastructure* components generally provide framework support and enables flexibility in the application deployment. *User interface* manages interaction with users capturing inputs and providing reports and views. The *application domain* components capture and codify domain knowledge largely provided by subject matter experts (SME) within Corus.

4.5 SYSTEMS EVALUATION

A successful system should perform to specification, be implemented on time and to budget, be reliable, be acceptable to its users and above all help the organisation to achieve its objectives (Law & Longworth, 1987).

The evaluation of the developed system was an integral task of the development process. As described in the previous section, formative evaluation was carried out periodically with potential end-users for functionality, robustness, clarity, ease of use, and ease of learning. The evaluations provided a good understanding of what the users wanted and what problems they were experiencing. To support this process, a software tool, called *Log Manager* was developed and incorporated into the system to enable remote logging of errors, requests, comments and concerns.

PROMIS is a complex and new system. Experience with the system, has shown that new users require 2-3 days of training depending on the user's interest and domain knowledge. Users had to be trained on the concepts of modularisation of design, parametric product specification, composition and use of templates for complex building objects, interaction with external applications, and generation of reports.

Substantial efforts has been invested into ensuring consistency of the interface using intuitive tree structures, lists, menus, toolbars, tool tips, drag & drop facilities, etc. A user, with previous exposure to the Windows Operating System would have no problems using the interfaces. A summative evaluation with existing users was carried out for PROMIS version 1.2, released June 2004. This involved direct observation while doing their normal work with the author taking notes. The users were unaware they were being observed so as not to alter their behaviour or performance levels. The evaluation revealed some bugs in the processing of some complex object. Bugs and areas for improvement were also identified in a number of application features and the user interface. Some of the issues raised during the evaluation have being addressed through the usual refinement process. Outstanding issues are highlighted in the next chapter.

4.6 SUMMARY

This chapter has described the research undertaken towards achieving the EngD project objectives. It describes the work done during each of the project phases including preliminary studies, evaluation of technologies, systems development and systems evaluation. It highlighted the role played by the author during the systems development, which involved working closely with Corus staff. Figure 13 presents the project timeline running from October 2000 to June 2004. While there is no definite end date to any of the phases and there were general overlaps, the timeline shows the time interval where the tasks were largely concentrated. Milestones have been marked to highlight key deliverables throughout the project.

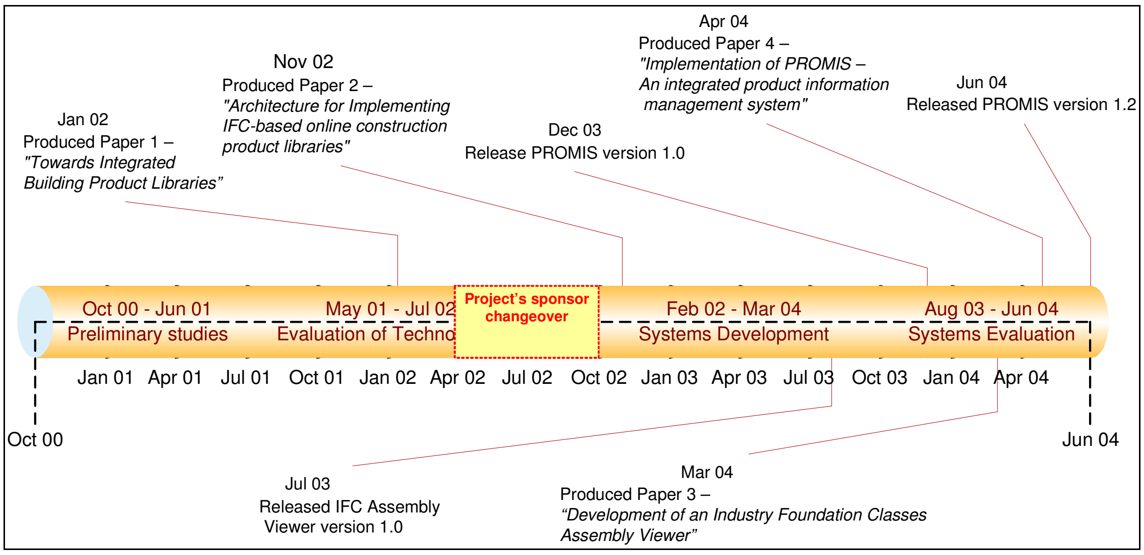

Apr 04
Produced Paper 4 –
"Implementation of PROMIS – An integrated product information management system"
Nov 02
Produced Paper 2 –
"Architecture for Implementing IFC-based online construction product libraries"
Jun 04
Released PROMIS version 1.2
Jan 02
Produced Paper 1 –
"Towards Integrated Building Product Libraries"
Dec 03
Release PROMIS version 1.0
Oct 00 - Jun 01
Preliminary studies
May 01 - Jul 02
Evaluation of Techno
Project's sponsor changeover
Feb 02 - Mar 04
Systems Development
Aug 03 - Jun 04
Systems Evaluation
Jan 01
Apr 01
Jul 01
Oct 01
Jan 02
Apr 02
Jul 02
Oct 02
Jan 03
Apr 03
Jul 03
Oct 03
Jan 04
Apr 04
Oct 00
Jun 04
Jul 03
Released IFC Assembly Viewer version 1.0
Mar 04
Produced Paper 3 –
"Development of an Industry Foundation Classes Assembly Viewer"

Figure 13: The EngD Project Timeline

5 FINDINGS AND IMPLICATIONS

5.1 INTRODUCTION

The purpose of this chapter is to present the findings of this EngD project and to set out the implications of the project findings to the construction industry, and the project sponsor in particular. The research findings are highlighted in Section 5.2 followed by the implications in Section 5.3. A critical evaluation of the EngD project is provided in Section 5.4 and recommendations for further research are in Section 5.5. This chapter concludes in Section 5.6 by summarising the achievements of the project.

5.2 RESEARCH FINDINGS

Preliminary observations of the construction industry revealed four needs with regard to the management of product information. These are as follows:

1. Improvements in the implementation of product libraries;
2. Integration of product libraries;
3. Integration of project information; and
4. Integration of product and project information.

Every finding of this EngD project can be associated with addressing one or more of these needs. One of the outputs of this EngD project is a software tool that addresses all these needs for a specific business and perhaps for similar businesses. It remains to be proved that the solution can be extended to generic construction business but suggestions are offered for further research to extend the developed framework and application to the wider construction industry.

5.2.1 EXISTING METHODS OF PRODUCT INFORMATION DELIVERY

Product information is created by manufacturers, suppliers, fabricators and distributors. External researchers occasionally add performance data and other users add feedback based on their experience with the specific products. This suggests that product information is modified throughout the life of the product. In a construction project, product information is needed in every phase from feasibility studies to facility management, and possibly demolition. Product information is commonly sourced through information brokers who aggregate information from several sources in printed catalogues and brochures and/or in electronic formats on CD-ROMs, DVDs and online websites. Many manufacturers provide these services directly to their target users. The structure and content of product information in these media varies widely.
The primary motivation for publishing product information is to increase the product's market share by getting the product specified in the hope that it would eventually be procured. Information provided by manufacturers is geared toward this purpose. Paper catalogues and most electronic versions are purely presentation-oriented with text and images. Some electronic sources go beyond presentation to provide information in some reusable format – specifically, CAD drawings for inclusion in designs. Corus Construction Parts is an example of such sources delivered on CD-ROM. BricsNet Product Center (*www.arcat.com*) and Sweet's Catalogue (*www.sweets.com*) are Web-based examples.

Given the amount of information contained in a typical product library, the organisation of the information critically affects its usability and accessibility. In electronic product information sources, tree-structured classification systems are common as well as grouping by manufacturers and product application. These are usually complemented with free-text search for faster access. Other search methods including parametric search, performance-based search and knowledge-based search are being researched but there is currently no commercial implementation in the market. Paper 1 describes product information delivery methods in more detail and further discussions on search methods are contained in Paper 2.

5.2.2 PROBLEMS WITH EXISTING METHODS

There are a number of problems associated with current delivery methods. Some of these are highlighted below:

Problem 1 – *Product information is not readily accessible*. There are too many disjointed sources sometimes providing copies of the same information.

Problem 2 – *Product information is rarely complete*. All product characteristics necessary for complete description or modelling including shape, properties, behaviour data, performance data, use conditions, transport requirements, assembly requirements, disposal requirements, etc., are not captured.

Problem 3 – *There is no agreement on the information content, structure, and format for presentation* as individual brokers implement their preferred representations. This becomes a problem when attempting automated information sourcing and reuse.

Problem 4 – *Classification systems vary widely* depending on the locality and target users. Once again, this is only a problem when integrating systems.

Problem 5 – *There is lack of continuity* or persistent access to archival information. It is impractical to keep paper catalogues through the life of a construction project, CD-ROMs require considerable care and Web-based information is transient by nature. Direct contact with manufacturers may not be possible in the uncertain business world. This inability to access archival information means product information, in most cases, is lost before the product falls out of use.

Problem 6 – *Facility for two-way communication are seldom provided*. Examples of communication here would include feedback to supplier, critical updates to users, and supplier responses to users requests, such as RFI (Request for Information), RFQ (Request for Quotation), RFP (Request for Proposal).

Problem 7 – *Search methods are limited, slow and rather unintuitive*. They are limited because the lack of agreement on the description of things hampers intelligent processing and matching; *slow* because the existing text comparison is computationally intensive given the amount of information involved; and *unintuitive* because domain knowledge is not utilised.

As a result of these technological limitations and the more ingrained resistance to change in the industry, there is limited use of product libraries. Users resort to reusing *time-tested* products with a view to limiting their risks with *untried* products. This may lead to repeated bad choices and high opportunity cost.

In addition, the prevalent contracting forms and fragmentation of the construction industry hamper information sharing, not least product information. There is inadequate investment in research and development activities which implies that change is not being effected as fast as is achievable. There is little or no disagreement on the need for, and potential benefit of, information sharing through integrated systems. Paper 2 substantiates the problems identified here and presents the industry's efforts aimed at addressing them.

5.2.3 INDUSTRY'S EFFORT TOWARDS IMPROVEMENT

The construction industry has invested considerable effort into improving efficiency and productivity, particularly in the United Kingdom. Integration of construction information is one of the recognised efforts. This has been the focus of Government institutions, research establishments, academia and major industry players for well over two decades. Examining the complexity of communications within a typical construction project team (Anumba et al., 2000) would affirm that the integration task is not trivial. Apart from the technical problems to be solved, there are mitigating cultural issues that needs to be fully understood and resolved.

Unlike project information, integration of product information sources has received limited research and development attention in the United Kingdom. Following the Latham report (Latham, 1994), the Construction Sponsorship Directorate (CD), of the then Department of Environment, with representatives of the construction industry, developed an information technology strategy aimed at increasing the efficiency of the construction industry. The need for an industry-wide library of design objects, which correspond to actual building products, was established (DoE, 1996). The Building Research Establishment (BRE), Visual Technology Ltd and Department of Environment again built on the strategic study findings and results of past projects, particularly COMBINE (Augenbroe, 1994) through the Advanced Reusable and Robust Object Warehouse (ARROW). ARROW demonstrated integration with distributed manufacturer databases and CAD systems (Amor & Newnham, 1999). There were other European research projects including Product Catalogue for Global Engineering Network Intelligent Access Library (PROCAT-GEN) (Radeke, 1999) and RINET (Hannus & Pirhonen, 2001). PROCAT-GEN sought to establish market potentials for electronic delivery of product catalogues and defined a Web-based application service infrastructure to support it. RINET (Hannus & Pirhonen, 2001), by VTT Building Technology in Finland, demonstrated feasibility of an Internet-based, single point of entry system to distributed manufacturer databases. Paper 1 (see the Appendix) provides a more detailed description of these efforts.

Both ARROW and RINET recognised the need to have a common understanding of the meaning of information (that is, its semantics) and embraced the IFC standard for data modelling and exchange. They also recognised the need for improved methods of information retrieval through classification and search methods.

5.2.4 PRODUCT MODELLING THE IFC WAY

Information modelling is widely accepted as a plausible approach to information exchange and sharing within many industry sectors including AEC/FM (Liebich & Wix, 2002). The IFC can play a part in addressing every one of the seven problems associated with existing product information delivery methods as enumerated in Section 5.2.2. The following explains how:

Problem 1 – Both ARROW and RINET projects have already demonstrated that single point of entry, or at least fewer points of entry, can be achieved through aggregation or integration of manufacturer databases. This can potentially eliminate the problem of unnecessary duplication of product information.

Problem 2 – The essence of product modelling is completeness (Augenbroe, 1998). Beyond presentation, product modelling approaches seek to provide a mechanism for describing product data throughout the product's life cycle.

Problem 3 – Product modelling provides definitions for objects in a domain along with conformance specification to ensure data (content) validity. The data structure is also intrinsically contained in the definitions. Text-based data formats such as STEP Part 21 and ifcXML can be used.

Problem 4 – ARROW implemented two classification systems, IFC and Uniclass (CPIC, 1997) thus demonstrating feasibility of using IFC and combining it with a local classification system.

Problem 5 – Every object in IFC has *OwnerHistory* attribute defined (See Figure 14). This can capture extensive auditing information necessary for version control implementation.

Problem 6 – Two-way electronic communication requires well defined message standard. IFC can play a role in the message body.

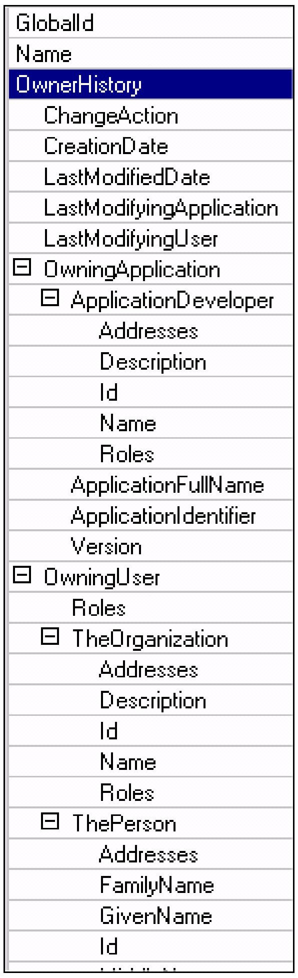
Problem 7 – Advanced search methods like parametric search and knowledge-based search can best be implemented where product metadata, in the form of IFC definitions, is available.

These are the potentials of an industry-wide product modelling approach as represented by the IFC effort. The actualisation of these depends on the readiness of the IFC to support product definitions. Further discussions on this can be found in Paper 2 which concluded:

The IFC model is the foundation for interoperability in the AEC/FM industry. A well-engineered application architecture and implementation based on the model can meet the industry's present and future requirements for reliable, flexible and available product information, which is complete and reusable in software applications employed in the industry.

5.2.5 ISSUES WITH IFC

The potential of the IFC is not in doubt but the question that faced the author at a point during this research was *"Can we use the IFC for product library implementation now!?"* The answer to this was negative given the time afforded the project. There were two major issues with the IFC model that informed this conclusion:
The IFC specification is currently incomplete for product library implementation. Although the IFC is fast evolving and user extensible through property sets, it was observed that the amount of work that would be required in extending the standard to capture the sponsor organisation's products alone is extensive. This would involve considerable time and effort and possibly detract from more pressing implementation issues; and



```
GlobalId
Name
OwnerHistory
   ChangeAction
   CreationDate
   LastModifiedDate
   LastModifyingApplication
   LastModifyingUser
⊟ OwningApplication
   ⊟ ApplicationDeveloper
         Addresses
         Description
         Id
         Name
         Roles
      ApplicationFullName
      ApplicationIdentifier
      Version
⊟ OwningUser
      Roles
   ⊟ TheOrganization
         Addresses
         Description
         Id
         Name
         Roles
   ⊟ ThePerson
         Addresses
         FamilyName
         GivenName
         Id
```

Figure 14: IfcOwnerHistory – IFC Object Auditing Attribute

IFC is specified with EXPRESS modelling language. EXPRESS is perhaps the most expressive and unambiguous modelling language available but it is not the easiest to understand and tool availability is comparatively very limited. Paper 3 reviews tool support for EXPRESS-based implementations. Unfortunately, the limitations of EXPRESS are inherent in IFC.

IFC version 1.0 was announced in June 1996, less than a year after the formation of the International Alliance for Interoperability (IAI). Since then, five other releases have been published including version 1.5, 1.5.1, 2.0, 2x and 2x2. Yet, a survey of IFC-compliant applications and development tools (IAI-ISG, 2003) would suggest that either the software vendors are not as eager as the modellers (perhaps for fear of losing market share) or the implementation tasks are considerably difficult. Whichever is the case, the crux of the matter was that *there was no tool that suited our chosen development platform – .NET platform*. The .NET platform was selected because it offered a controlled, multi-language, multi-platform, rapid development environment particularly geared towards integration of network resources and applications.

5.2.5.1 Contribution to Solution

There was a need for software that would bridge the gap between information modellers and programmers working with EXPRESS-based models by providing tools to support understanding of the models and their deployment in programming environments. The

IFC Assembly Viewer was such a tool. It validates EXPRESS schema definitions, translates the definitions into the Visual Basic.NET programming language, compiles the generated source codes into a class library, and provides a two-way synchronisable implementation view to complement the HTML schema documentation. Some of the findings from this development are as follows:

1. Errors in the models could be detected early during validation. For example, the published IFC 2x standard was found to contain two non-critical errors;
2. The need for future language mapping efforts, with its associated cost and time investment is eliminated. This would facilitate faster and cheaper implementation;
3. Implementation options were extended to cross-platform and multi-language environments;
4. The need for developers to learn EXPRESS to any great depth was eliminated; and
5. The task of evaluating the suitability of the IFC model was simplified with single-point access to modelling and implementation views, and automated generation of assemblies and views.

Details of the implementation and benefits of the IFC Assembly Viewer can be found in Paper 3 (see the Appendix).

5.2.6 ComPROMISing on Standard

It was established that the product-modelling standard approach to product library development offers the best prospect for improvement and integration. The IFC was identified as the most applicable although it is impractical to implement it in its current state. A compromise is needed. This involved assuming a standardised product model and addressing the issues associated with integration of product information sources for construction projects. This is the PROMIS approach.

5.2.6.1 PROMIS Approach

The PROMIS approach is based on the concept of the Universal Building Model (UBM) developed within the sponsoring organisation. It is not dissimilar to the concept of the IAI shared project model (see Figure 15). It is synonymous with the *Shared Object Model* in that it aims to support interaction of the construction project team with a shared information model throughout the life of the building project. Figure 16 depicts this interaction.

PROMIS is an implementation of the UBM vision based on a non-standard information model. It provides an integration framework and supporting software tools for the management of whole-life product information. Paper 4 (see the Appendix) describes the PROMIS framework in detail including architecture, subsystems and supporting technologies. PROMIS is a secure, highly scalable, database-driven, component-based system consisting of three subsystems namely: ProductWeb, ModelManager.NET, and Project-Product Web Service (P2Service). ProductWeb is a Web-enabled database application for collection, storage and retrieval of construction product information. ModelManager.NET is a desktop application which encompasses building objects and facilitates the decomposition of building into objects for a specific construction project. P2Service is a Web Service which links information in ProductWeb with building objects within specific projects in ModelManager.NET. Communication between the subsystems is achieved through standard HTTP Web protocol. Communication with desktop applications is realised through the Component Object Model (COM) and file transfer for monolithic *legacy* applications.

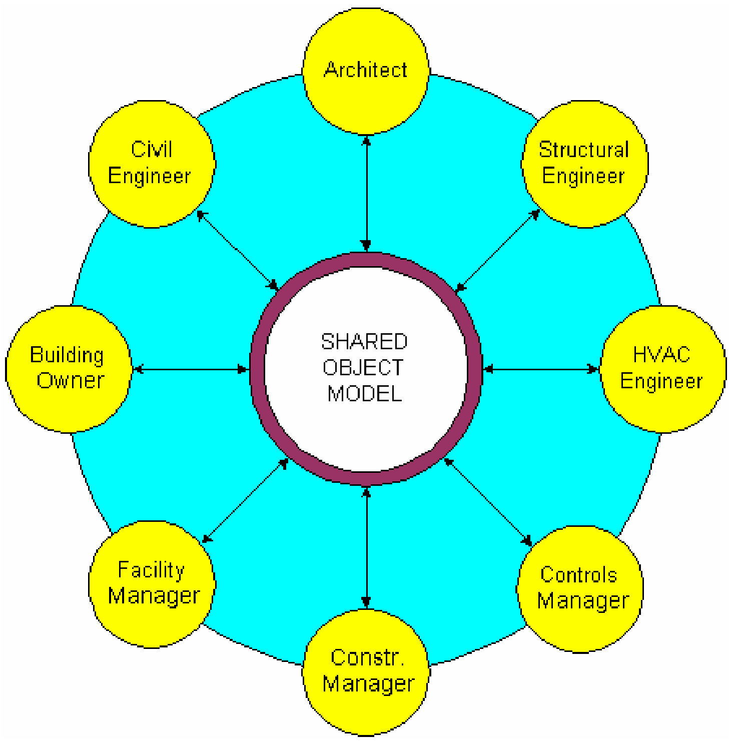


Figure 15: IAI Shared Project Model

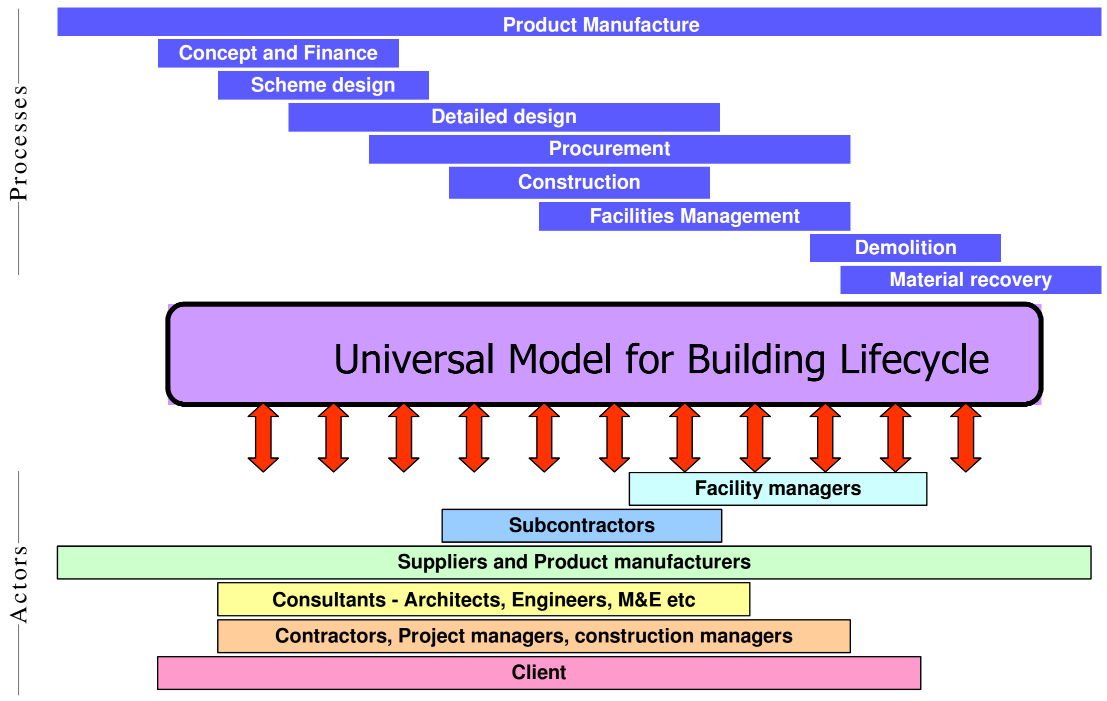


Figure 16: UBM Interaction Model

5.2.6.2 PROMIS - Current Status

PROMIS is evolving rapidly and considerable improvements have been made since the last version documented in Paper 4 (see the Appendix). Figure 17 highlights the present contribution of PROMIS to the UBM vision.

The core data repository has been implemented as well as the links for 3D CAD model, cost estimating, materials/quantities compilations, procurement, and manufacturing through Computer Numerical Control (CNC) Machine Tool programming. Project management and visualisation links have also been prototyped. It is worth noting that supply chain (or procurement) and manufacturing links are currently enabled via integration with SAP™ (*www.sap.com*) and hsbSoft's hsbCAD ™ (*www.hsb-cad.com*) respectively. SAP™ (Systems Applications and Products) is an Enterprise Resource Planning (ERP) system for managing business functions including marketing, sales, accounting, finance, human resources, and production and material management. hsbCAD™ is an object-oriented, 3D model-based, CAD/CAM software solution for the wood and steel construction industry.

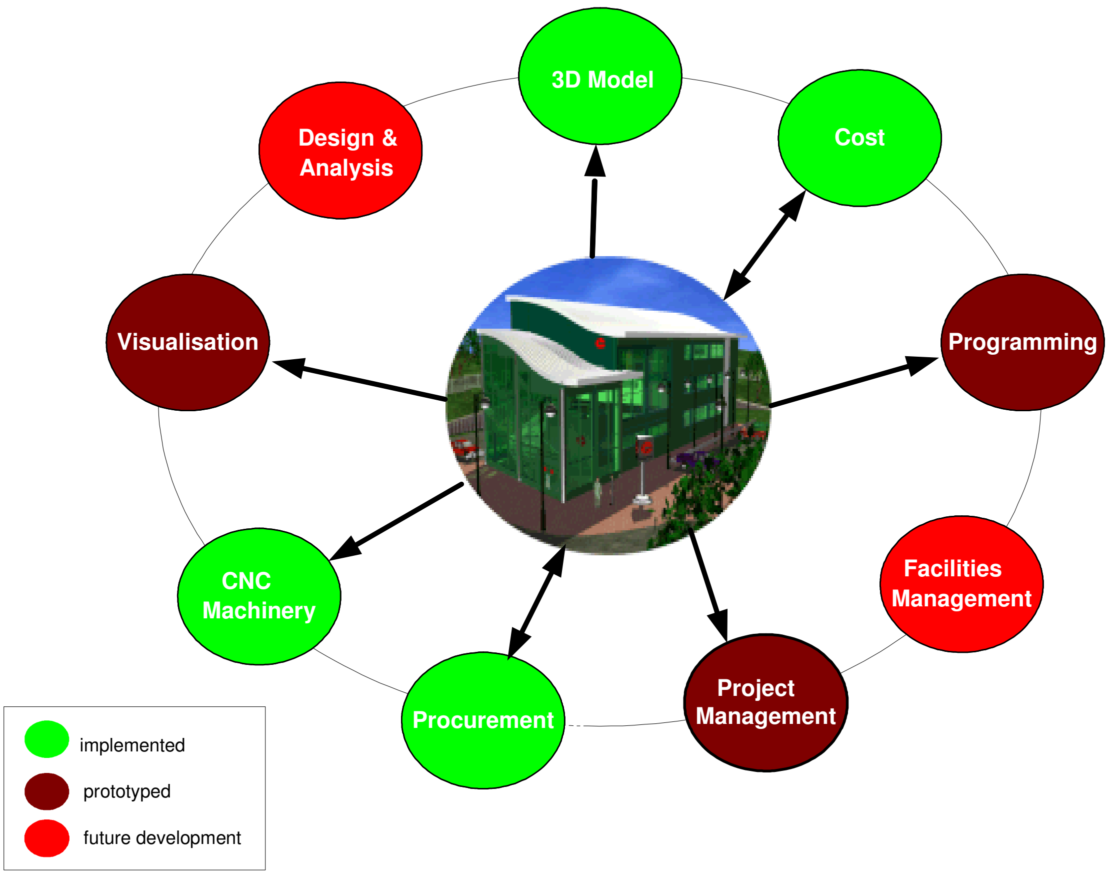


Figure 17: Current Status of PROMIS in the context of the UBM

5.3 IMPLICATIONS/IMPACT OF THE ENGD PROJECT

The development of PROMIS has the most far reaching impact and implications for the EngD project sponsor, particularly the target business unit. There are, however, other findings with their respective implications to the project sponsor and similar organisations.

The findings from the preliminary studies and evaluation of technologies add to the body of knowledge available today on the requirements for the implementation of product libraries. This knowledge is presently being used to improve the ProductWeb implementation.

The availability of the IFC Assembly Viewer puts the sponsor organisation in good stead to evaluate future releases of the IFC standard quickly for various applications within its business units. It has also been found to accelerate the development of prototypes based on the standard.

PROMIS has been deployed as the core business system for the target business unit. It provides information to other systems within the business unit and acts as the main source of construction project and product information. Some of the noted and expected impacts of this are discussed in subsequent subsections.

5.3.1 INTEGRATION OF MODULAR PRODUCT DEVELOPMENT PROJECT TEAMS

The modular product development project team within the target business unit works on the same information based on common objects albeit concealed through the implementation interface. This single-source system encourages transparency between the team through concurrent working and controlled visibility of information. Concurrent working is achieved through project information versioning. Published project information are assigned version number and revisions can be created by authorised users who can make changes by *unpublishing* the project information. Merging of related revisions to create new published versions is a manual task undertaken by a designated project manager. This environment facilitates traceability, activity control and work-flow management.

This way of working however has the potential to create anxiety for some individuals who may be well-exercised in the over-the-wall model. Since the success of the system is hinged on acceptability by all users, efforts are continuously being invested into ensuring the users feel in control of the system.

5.3.2 BETTER QUALITY PRODUCTS

Quality control is built into the system through precision product engineering incorporating engineering tolerances rather than construction tolerances. Products are collaboratively and visually designed before manufacture. Errors are considerably reduced and mistakes are more visible. A product approval process helps to control the use of products within assemblies and ensures that assemblies are *designed for manufacture*.

5.3.3 PRODUCT DEVELOPMENT

PROMIS provides an environment for collaborative and rapid prototyping and design. What-if analysis can be carried out to determine the viability of designs and design issues identified by the system are raised for resolution.

5.3.4 SUPPLY CHAIN MANAGEMENT

PROMIS implements a supplier/product approval process, which helps to ensure pre-selected products are incorporated into designs. Rapid just-in-time supply can be implemented to maximise efficiency and reduce storage requirements. The system can benefit from framework agreements as suppliers would be required to provide their product information through ProductWeb, hence conforming to a *standard*. These agreements are not yet in place and existing product information are being manually translated into forms acceptable to ProductWeb by the target business unit. It can be expected that the situation would change radically when the system is fully developed.

5.3.5 MODULAR PRODUCT DEVELOPMENT PROJECT IMPLEMENTATION

PROMIS reduces the design life cycle to a minimum and supports most of the construction life cycle through a building block approach. In addition, all information about products is related to virtual design and can be easily accessed in context (e.g. for health and safety).

Within the target business unit, PROMIS is expected to reduce capital cost, defects, accidents and construction time. It could also improve predictability, productivity, turnover and profit through rapid and virtual design, precise costing and engineering, and controlled waste production and reuse. It is expected that once the system is proved within the target business unit, it would be rolled out to similar businesses to harness the same benefits.

5.3.6 THE CONSTRUCTION INDUSTRY

It has been established that the wider construction industry would benefit greatly from integration of the construction project teams. All the implications described above are equally applicable to the industry but these hinges on the acceptance of a common information model. The information model developed for PROMIS is not sufficiently detailed to support the global construction industry. The IFC is the most suitable information model for industry-wide integration and efforts should be concentrated on supporting the development of the IFC, for example with tools such as the IFC Assembly Viewer. This would facilitate implementation and prototyping efforts that would eventually bridge the integration gaps.

The PROMIS framework offers a credible architecture for modular construction businesses with high incidence of repeat products. Perhaps the search for an industry-wide solution that fits all construction products and processes should be redefined and efforts should be concentrated on providing solutions like PROMIS for other types of construction business.

5.4 CRITICAL EVALUATION OF THE ENGD PROJECT

The tangible outputs from this EngD project are the IFC Assembly Viewer and the PROMIS system. The IFC Assembly Viewer was developed entirely by the researcher during the course of the project. The concept of the Universal Building Model existed before the researcher joined the sponsoring organisation. There was also a Web-based product library implementation, named COLIN, on which the ProductWeb implementation was built. Aspects of the UBM vision particularly interaction with desktop applications like AutoCAD and Ms Excel had also been prototyped. The

researcher brought his expertise in systems development to bear in developing an extensible and configurable software framework and application. This includes the development of production-strength ModelManager.NET application and P2Service application. Other members of the development team contributed their domain expertise in encapsulating product knowledge in ModelManager.NET building objects that were developed for the test modular construction business.
The following subsections highlight current gaps and limitations in the implementation of the various software products of this research project.

5.4.1 IFC Assembly Viewer

The IFC Assembly Viewer offers a novel environment that collates all information necessary to understand the IFC information model. It bridges the gap between the information modeller and the application developer by providing programming language interpretations of the EXPRESS information model with direct link to the model documentation. Relationships between different elements within the model are made more comprehensible and users can explore this to any desired level of details.
The IFC Assembly Viewer is however incomplete as an implementation tool. The implementation compiles EXPRESS models into .NET class library but the usefulness of the generated library is limited as a result of the following:

1. The IFC Assembly Viewer does not presently translate rules and algorithms which are the basis for model data validation;
2. It is based on a non-standard mapping of EXPRESS constructs to .NET programming language constructions. Mapping definitions are usually standardised definitions but currently no such standard exists for any of the .NET languages; and
3. The IFC Assembly Viewer is tightly-coupled with the HTML schema documentation provided with schema definitions rather than the original EXPRESS schema because it is easier to navigate. It parses links in the documentation to provide synchronisation documentation/implementation views. This implies that any change to the structure of the links in the documentation would adversely affect the functioning of the IFC Assembly Viewer.

5.4.2 PROMIS

PROMIS is an innovative software system that supports building modelling based on a shared product model, parametric product specification and distributed databases. It provides an extensible, collaborative and knowledge-driven design environment that incorporates suppliers' components and product information with virtual product designs. The developed system proves integration of industry applications can be achieved by separating data ownership from software applications that use the data.
PROMIS is currently limited in its application. It is not applicable to generalised buildings and structures but to modular construction manufactured off-site in a controlled factory environment. Complete and precise engineering details are required to model the product and given the time taken to finalise the product model, the effort is only worthwhile for repeatable products.
Another factor that limits the application of PROMIS to the wider construction industry is that it is not based on a standardised product model such as the IFC. The generic

framework or application architecture it defines was not tested with the current IFC for pluggability and compatibility issues are unknown. It can be expected that some additional development effort would have to be invested to adapt the framework for the IFC model.

Regarding current features and capabilities, PROMIS captures information about structures as well as interior fittings and finishes including details for services, it does not currently display the 3D model of the interior fittings and finishes. This limitation hampers definitive visual evaluation of designs particularly in relation to services.

The current deployment of PROMIS as an end-to-end solution for the construction process of modular buildings is lacking capabilities for facility management, high-end visualisation, and building design and analysis such as structural, lighting, thermal, and acoustic analysis.

There are other limitations that can be associated with specific subsystems of PROMIS. These are discussed in the following subsections:

5.4.2.1 ProductWeb

COLIN, the predecessor of ProductWeb, was developed as a presentation-oriented Web application using Active Server Page (ASP) technology. This technology poses some limitations to integration. The implication of this inherent limitation is that ProductWeb cannot easily be modified to integrate with other sources of product information or even its clone implementations. P2Service sought to address this problem through a back-door approach by gaining direct access to ProductWeb's back-end data repository but this has its own limitations. In addition, ProductWeb has not implemented any standardised classification system for construction products. Although it presents information with an intuitive interface, users that are familiar with specific classification systems could benefit from some mapping of the information to known/established classification systems.

Although based on parametric product specification, ProductWeb lacks a parametric search facility. Product parameters are presently only used in ModelManager.NET. There is therefore little motivation for stand-alone users of ProductWeb to specify product parameters.

There are plans to redesign ProductWeb as a multi-tier Web application to facilitate integration. It is also expected that the redesigned application would incorporate some classification systems as well as advanced search methods including parametric search and search-by-design-criteria. The latter would extend parametric product specification to enable searches by fit-for-purpose design parameters. For example, a search for universal beams within ranges of *radius of gyration* (for buckling calculations), and *second moment of area* (for deflection calculations)

5.4.2.2 ModelManager.NET

ModelManager.NET is unprecedented in its approach to handling complex construction products such as wall, floor, ceiling, window and door assemblies. ModelManager.NET is also capable of composing these elements with varying configurations in pre-engineered modules. These are sufficient for the current deployment. However, they are not enough for complete construction solutions. There are many cladding, roofing, and foundation solutions to be considered as well as the interfaces with these external assemblies.

ModelManager.NET is limited to the Windows Operating System platform. Future porting to other platforms would depend on the availability of .NET framework and

COM interfaces to the integrated desktop applications on those platforms. Also, because vendors of the integrated desktop applications can change their COM interfaces as desired, ModelManager.NET would have to be tested with every new release of those applications to ensure compatibility.
ModelManager.NET is not fully globalised hence it cannot be easily adapted to different regions and countries. Software globalisation involves isolating and encapsulating program elements likely to vary by locality, custom, and language including every text that will be presented to the user. While labels for product parameters and components are fully and dynamically configurable, messages, reports, and interface controls are described in the English Language.

5.4.2.3 P2Service

P2Service was envisioned as an information broker in the PROMIS architecture. It offers an immediate solution to integration problems caused by the chosen implementation technology for ProductWeb. A simple test of the application for throughput, latency, execution time, and transaction time showed that there is considerable performance degradation due to this additional communication tier between ModelManager.NET and ProductWeb. This performance problem can be attributed to the underlying messaging and transport protocols used - Simple Object Application Protocol (SOAP) and Hypertext Transfer Protocol (HTTP). While HTTP is long established, SOAP is maturing and harbours a number of performance and scalability problems in its multi-step communication process. P2Service was eventually removed from the configuration of the current deployment.
Improving the quality of service for P2Service is a priority for the next release of PROMIS. This would not only aim to improve its performance but also its reliability, accessibility and security.

5.5 RECOMMENDATIONS FOR FURTHER RESEARCH

PROMIS addresses the technological needs for integration of the construction project team for a specific type of construction business. There are sociocultural implications to this new way of working which need to be understood and addressed in future implementations. Further research into the effects of integrated construction project information system on construction processes and people is therefore recommended.
Further research into the requirements for incorporating the IFC into the PROMIS architecture and the implication for existing projects executed with PROMIS is also considered necessary. The result of this would facilitate the transition to the international product modelling standard when required.
The PROMIS architecture can be extended beyond modular construction into steel-framed building construction. There are software applications covering the structural steel design process from conceptual design to detailing, fabrication and construction. Many of these applications implement the CIMSteel Integration Standards (CIS/2) (Crowley & Watson, 2000) product model for project information exchange. Research to investigate and develop a CIS/2 translator for the PROMIS system is recommended to harness the capabilities of these applications.
The production of high quality, textured, interactive 3D-models of construction products from simple objects to entire building complexes would deliver immense benefits in construction project execution from briefing to operation and maintenance. PROMIS aims to incorporate visualisation to facilitate efficient knowledge transfer for

complex products and processes using 3D-realtime display and animation. Other benefits of visualisation in construction are identified in Aouad et al. (2000). More research is needed to determine the types of interaction required by the construction user and how best to deliver such efficiently within the PROMIS framework. Other recommended areas for further research, implementation and integration with the PROMIS framework include facility management and design analysis.

Parametric product specification opens up new possibilities in terms of search methods for construction product information. Knowledge-based search is one area of such research. In the process of executing this EngD project, it became apparent that this can be approached with step-wise refinement starting with parametric search and progressing to other search methods incorporating industry knowledge including search-by-design-criteria, and product and supplier performance factors such as reliability, proximity, whole-life cost, etc. Further research to investigate and implement advanced search methods for construction product information would be useful.

5.6 CONCLUDING NOTES

This research process has helped the researcher to understand the issues associated with the delivery of information technology for product information management. In this systems development project, available technologies were considered as well as past efforts towards the efficient delivery of product information. The shortcomings of these technologies and approaches were identified and a software framework and application was developed to address the shortcomings for a modular construction business. The developed system was integrated with other applications used in the industry to deliver the end-to-end solution. It was tested and verified with users through many revisions leading to its eventual deployment as the hub or core of an integrated environment running live construction projects for the target business. Although the benefits of the developed system to the sponsoring organisation were not measured due to time constraints, it is expected that the system would help reduce marketing costs, IT costs, planning times, lead times, operating costs, staff requirements and waste. It was observed that the system facilitated faster response to client enquiries and change request and it provided faster access to information and greater opportunity for data reuse across product development projects. The researcher believes that developments such as the PROMIS system and the IFC Assembly Viewer will significantly contribute to integration efforts within the construction industry.

6 REFERENCES

Amor, R. & Newnham, L. (1999), CAD interfaces to the ARROW manufactured product server, *in* G. Augenbroe & C. Eastman, eds, 'Proceedings of the Eighth International Conference on Computer Aided Architectural Design Futures [ISBN 0-7923-8536-5] Atlanta, USA', Kluwer Academic Publishers, pp. 1–11.

Anumba, C. J., Bouchlaghem, N. M., Whyte, J. & Duke, A. (2000), 'Perspectives on an integrated construction project model', *International Journal of Cooperative Information Systems* (3), 283–313.

Aouad, G., Ormerod, M., Sun, M., Sarshar, M., Barrett, P. & Alshawi, M. (2000), 'Visualisation of construction information: A process view', *International Journal of Computer-integrated Design and Construction* (4), 206–214.

Appleton, B., Berczuk, S., Cabrera, R. & Orenstein, R. (1998), Streamed lines: Branching patterns for parallel software development, *in* 'Proceedings of the 5th Annual Conference on Pattern Languages of Program Design (PLoP'98)', Washington University. Technical Report No. WUCS-98-25.

Augenbroe, G. (1994), An overview of the COMBINE project, *in* R. J. Scherer, ed., 'Proceedings of First European Conference on Product and Process Modelling in the Building Industry, Dresden, Germany'. copy available at http://erg.ucd.ie/combine/papers.html.

Augenbroe, G. (1998), Building product information technology, Executive white paper, Georgia Institute of Technology, Construction Research Center, Atlanta, GA 30332-0159.

Avison, D. & Fitzgerald, G. (2003), *Information Systems Development: Methodologies, Techniques, and Tools*, third edition edn, McGraw-Hill Education. ISBN 0-07-709626-6.

Beck, K. (1999), *Extreme Programming Explained*, 1st edn, Addison-Wesley Pub Co. ISBN: 0201616416.

Boehm, B. W. (1988), 'A spiral model for software development and enhancement', *IEEE Computer* (5), 61–72.

Clarke, R. (2000), Appropriate research methods for electronic commerce, Technical report, Department of Computer Science, Australian National University. Copy available http://www.anu.edu.au/people/Roger.Clarke/EC. Last visited: May 2004.

CPIC (1997), *UNICLASS: Unified Classification for the Construction Industry*, RIBA Publications. Construction Project Information Committee.

Crowley, A. J. & Watson, A. S. (2000), *CIMSteel Integration Standards Release (CIS/2)*, Vol. 2 - Implementation Guide, The University of Leeds and The Steel Construction Institute. ISBN 1859421008.

Dittrich, Y. (2000), Beg, borrow and steal - but what and what for?, *in* 'The 22nd International Conference on Software Engineering (ICSE 2000)', Limerick, Ireland.

DoE (1996), Feasibility of a library of building design objects, Technical report, Construction Sponsorship Directorate, Department of Environment, UK.

Ducker (2002), 'Study assesses internet usage in construction', *Glass Magazine* . Authored by Ducker Research Europe GmbH, Johannishof, Johannisstr. 20 D-10117 Berlin, Germany.

Egan, J. (1998), Rethinking construction, Technical report, Department of Environment, Transport and the Regions, HMSO, London. The report of the Construction Task Force to the Deputy Prime Minister, John Prescott, on the scope for improving the quality and efficiency of UK construction.

Evbuomwan, N. F. O. & Anumba, C. J. (1998), 'An integrated framework for concurrent life cycle design and construction', *Advances in Engineering Software* (7-9), 587–597. Published by Elsevier Science Limited and Civil-Comp Ltd.

Gable, G. G. (1994), 'Integrating case study and survey methods: an example in information systems', *European Journal of Information Systems* (2), 112–116.

Ghauri, P. & Gronhaug, K. (2002), *Research Methods In Business Studies, A Practical Guide*, second edn, Financial Times Prentice Hall. ISBN 0273651102.

Hannus, M. & Pirhonen, A. (2001), 'RINET - building product database', http://cic.vtt.fi/projects/rinet/index.html.

Harper, C. (2003), ICT at work for the LSE ProCurement chain: Putting information technology and communications technology to work in the construction industry, Technical Report ESPIRIT 29948 - ProCure WP3/T3500/R3501, Corus Group.

Heiskanen, A. (2004), The reflective information systems practitioner approach as a research and learning expedient, *in* D. Redmiles, A. Mørch, K. Nakakoji & G. Fischer, eds, 'CHI 2004 Workshop on Designing for Reflective Practitioners', Software Research ICS2 210, University of California, Irvine Irvine, CA 92697-3425.

IAI-ISG (2003), *International Overview of IFC-Implementation Activities*, IAI-Implementation Support Group, Organisation website: http://www.iai.fhm.edu/iai_isg/. Last visited 10 June, 2004.

ISO (1994), Industrial automation systems and integration – product data representation and exchange – part 1: Overview and fundamental principles, Technical Report ISO 10303-1, International Standards Organisation, Geneva, Switzerland.

Jepsen, L. O., Mathiassen, L. & Nielsen, P. A. (1989), 'Back to thinking mode - diaries as a medium for effective management of information systems development', *Behaviour and Information Technology* (3), 207–217.

Johnson, R. E. (1992), Documenting frameworks using patterns, *in* 'Proc. Of the OOPSLA-92: Conference on Object-Oriented Programming Systems', Languages, and Applications, Vancouver, Canada, pp. 63–76.

Kruchten, P. (1996), 'A rational development process', *Crosstalk* (7), 11–16.

Kruchten, P. (2000), *The Rational Unified Process: An Introduction*, 2nd edn, Addison-Wesley Pub Co.

Latham, M. (1994), Constructing the team, Technical Report ISNB 0-11-752994-X, Department of the Environment, UK.

Law, D. & Longworth, G. (1987), *Systems Development: Strategies and Techniques*, National Computing Centre (NCC) Publications, Oxford Road, Manchester M1 7ED, England, UK.

Lejk, M. & Deeks, D. (2002), *An Introduction to Systems Analysis Techniques*, Addison-Wesley Pearson Education Limited, Essex, England.

Liebich, T. & Wix, J. (2002), Standard analysis - current AEC situation - building models, Technical Report IST-2001-32035, prodAEC - European Network for IT in Architecture, Engineering, and Construction.

Lyytinen, K. (1987), *Critical Issues in Information Systems Research*, Information Systems Series, John Wiley & Sons, New York, USA. Edited By R.J. Boland and R.A. Hirschheim.

Mathiassen, L. (1998), 'Reflective systems development', *Scandinavian Journal of Information Systems* (1-2), 67–117. ISSN: 0905-0167.

Mathiassen, L., Seewaldt, T. & Stage, J. (1995), 'Prototyping and specifying: Principles and practices of a mixed approach', *Scandinavian Journal of Information Systems* (1).

Mathiassen, L. & Stage, J. (1992), 'The principle of limited reduction', *Information, Technology and People* (2).

Nunamaker, J., Chen, M. & Purdin, T. D. M. (1991), 'Systems development in information systems research', *Journal of Management Information Systems* (3), 89–106.

Owolabi, A., Anumba, C. J. & El-Hamalawi, A. (2003), 'Architecture for implementing IFC-based online construction product libraries', *ITcon* **Vol. 8**, 201–218. Special Issue IFC - Product models for the AEC arena, Copy available online at http://www.itcon.org/2003/15.

Paprzycki, M. & Abraham, A. (2003), Agent systems today: Methodological considerations, *in* 'International Conference on Management of e-Commerce and e-Government', Jangxi Science and Technology Press, China, pp. 416–421. ISBN 7-5390-2369-4.

Parnas, D. L. & Clements, P. C. (1986), 'A rational design process: How and why to fake it.', *IEEE Transactions on Software Engineering* (2), 251–257.

Perry, D. E., Siy, H. P. & Votta, L. G. (1998), 'Parallel changes in large-scale software development: An observational case study', *ACM Transactions: Software Engineering Methodology* (3), 308–337.

Radeke, E. (1999), Final report of global engineering networking intelligent access libraries (GENIAL), ESPRIT Project 22.284, Technical report, The GENIAL Consortium.

Raelin, J. A. (2001), 'Public reflection as the basis of learning', *Management Learning* (01), 11–30.

Schon, D. A. (1995), *The Reflective Practitioner: How Professionals Think in Action*, Ashgate Publishing Company.

Wesley-Tanaskovic, I., Tocatlian, J. & Roberts, K. H., eds (1992), *Expanding Access to Science and Technology: The Role of Information Technologies*, United Nations University Press, 53-70, Jingumae 5-chome, Shibuya-ku, Tokyo 150, Japan. ISBN 92-808-0844-3.

West, J. D. (2000), *Extreme Programming: A gentle introduction*, ExtremeProgramming.org, http://www.extremeprogramming.org/.

Wilson, T. (2002), Information science and research methods, *in* 'Library and Information Science', Department of Library and Information Science, Comenius University, Bratislava, Slovak Republic, pp. 63–71.

APPENDIX A PAPER 1

Owolabi, A. A., Anumba, C. J. & El-Hamalawi, A. (2003), “Towards implementing integrated building product libraries”, *Construction Innovation* **3**(3), 175–194.

TOWARDS IMPLEMENTING INTEGRATED BUILDING PRODUCT LIBRARIES

ABSTRACT: Electronic product catalogues and brochures are gaining popularity but there is little agreement on content, format and searching methods. This limits their usability and integration with existing construction software tools. This paper examines a product-modelling approach to delivering building product information and describes a proposed multi-tier client-server environment. ISO/STEP and IAI/IFC building product models are considered to facilitate representation, exchange and sharing of product information. The proposed architecture incorporates scalability with middleware components that would provide single or few points of entry to integrated product information. This paper is part of a research project, which builds on the results of related projects including ConstructIT Strategy, PROCAT-GEN, Active Catalog, COMBINE and ARROW, towards implementing the required software components.

KEY WORDS**:** Building product modelling, Industry Foundation Classes, product libraries, data sharing, multi-tier client/server application.

1 INTRODUCTION

Construction product information is needed throughout the lifecycle of a construction project, from the feasibility study stage to demolition planning. Construction industry practitioners require complete and up-to-date information to assist in comparative evaluation and selection of products as well as for specification, installation, maintenance and disposal.

The major part of product information originates from product manufacturers, suppliers, fabricators and distributors. Research institutions occasionally add performance information. Users may also provide feedback, which influences future development or selection of the product.

Users' expectation is that product information should be available, comprehensive, accessible and reusable in their software environments. Inadequate or unavailable information causes lost opportunity in sales to manufacturers whose products were not specified. On the other hand, repeated bad choices are costly to users. Poor accessibility results in wastage of time, and usability of the system hinges a great deal on the reusability of the information produced.

Product information is currently available from numerous information brokers who aggregate information about hundreds of products from several manufacturers and organise the information according to some classification system. In many cases, manufacturers also make their product information available directly. The information may be structured and presented in printed catalogues and brochures or in electronic formats on CD-ROMs, DVDs and online. There is little or no agreement on the contents, semantics, formats and search methods supported.

The characteristics of the construction industry and the limitations of the software technologies in use within the industry have limited the delivery and comprehensive use of building product information on construction projects. The prevalent contracting forms and fragmentation of the industry hampers sharing of product information at different phases of the building lifecycle. This is further complicated by the nature of

buildings in that a building can be designed to incorporate products from different parts of the world and built with infinite combinations. In addition, the software tools used by project teams are incapable of exchanging or sharing data without some manual or electronic transcribing or translation.
Figure 1 is a simplified representation of the existing practices, highlighting the need for manual or automated translation or transcribing of product information between industry practitioners and their tools. Anumba et al. (2000) detail the complex communication requirements, which are also applicable to product information, within the virtual construction project team.

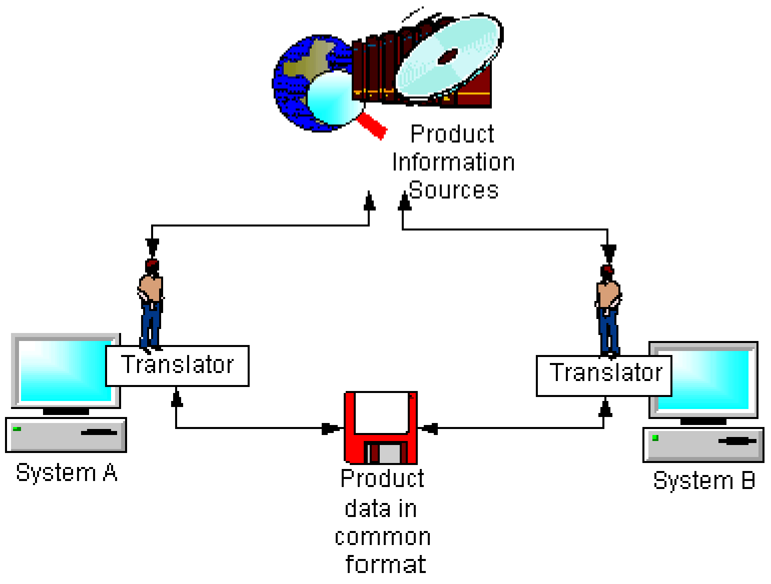


Figure 1: Existing method of obtaining and exchanging product information

In the UK, the construction industry is undergoing restructuring following the recommendations of Egan (1998). The dispersion of the industry is being addressed with partnering and team development. In addition, project teams are being encouraged to consider not only initial cost but also whole-life costs including opportunity costs and future costs; hence a “right first time” approach to eliminate waste and increase productivity. These changes require extensive sourcing and management of information, including product information.
Progressive international standardisation efforts defining industry-wide product models and specifications for exchange and sharing of product information have yielded implementable solutions addressing interoperability issues. The Industry Foundation Classes (IFC) being defined by the International Alliance for Interoperability (IAI), underpinned by International Standards Organisation’s Standard for the Exchange of Product Model Data (STEP), is most relevant. This is complemented by the IT standardisation efforts, including that of the World Wide Web Consortium (W3C) with eXtensible Markup Language (XML) specification, and Object Management Group with Common Object Request Broker Architecture (CORBA) specification.
This paper discusses a proposed architecture for the implementation and delivery of product libraries based on existing and emerging construction product models and open software technologies. It discusses current delivery methods, introduces the STEP and

IFC standards, describes the proposed architecture for the implementation of integrated product libraries and highlights relevant past projects.

2 DEFINITIONS

The starting point to understanding what constitutes product information is to define what the product is. Generally, a product is something produced by human or mechanical effort or by a natural process. Construction products are items acquired, manufactured or processed, for incorporation into construction work as defined by International Standards Organisation (ISO, 1989) and British Standards Institute (BSI, 1991). This is similar to the International Alliance for Interoperability (IAI, 2000) definition:

> *"A product is any object, manufactured, supplied or created for incorporation into an AEC/FM project."*

A project is understood here to mean some engineering activity leading to a product including objects that are created indirectly by other products. Hence an entire building is as much a construction product as a window frame.
Coyne et al. (1999) define *product information,* from the point of view of a designer, as "technical information about products, materials and finishes used in building and construction, including standardised components: doors, windows, taps, floor tiles, etc., which allow designers to make informed choices about alternative products, materials and finishes, and to know what attributes to specify, such as sizes, materials, finishes, colours, etc".
The concept of product model captures all perspectives because product models represent the dynamic evolution of product information in the course of the design, construction and facility management processes.

> *"A product model is a digital representation of a real world thing held to facilitate the unambiguous transfer of the information between computer systems and to support information sharing." (Eastman, 1999)*

Product information includes catalogue and technical information, specification, drawings, samples, costs and discounts, sources of supply and delivery, test reports, installation, operation and maintenance instructions, and disposal or recycling requirements. Performance information and user feedback could also be incorporated. Information required for a specific construction product would depend on the project phase, type of project and procurement route.
We define product library as:

> *A virtual or physical place, which contains systematically arranged digital information about physical, manufactured, supplied or created objects and provides access to users and owners of such information. It should support multiple classification systems, search methods and presentation formats.*

3 RELATED PROJECTS

In order to support the highly distributed or dispersed construction product information sources, a standardised, flexible, and extensible solution with built-in scalability, robustness and platform neutrality is required (Zarli and Poyet, 1999). Standard

interfaces, in the form of standardised data exchange, functional API to access shared data, common communication protocols and framework between applications, are also required to provide interoperability and an acceptable level of independence between the various subsystems. The reality is sources of product information are far from integrated and research efforts by different groups have attempted various aspects of the solution. Many of these projects achieved their set objective but an integrated solution still eludes us partly because all necessary standard interfaces and technologies were not available and partly because the developers assumed a utopia where all sources and users of product information would change or abandon their legacy systems, adopt a standard and invest in its implementation.
The architecture proposed here is designed as part of a research project towards developing a commercial strength integrated product library, building on findings of past feasibility studies and results of past research projects. This section describes some of these related projects including Construct IT Strategy, PROCAT-GEN, Active Catalog, RINET and ARROW.

3.1 CONSTRUCT IT STRATEGY

Towards the implementation of the Latham report recommendations (Latham, 1994), the Construction Sponsorship Directorate of the Department of Environment, with industry representatives, carried out a study, which established the UK construction industry information requirements. Product information was recognised as principal to all industry participants and the requirements for a library of design objects that correspond to actual building products were laid out. The report (DETR, 1996), detailing the findings and recommendations, also considered the implementation of different search methods, classification of library objects, integration of existing tools and the applicability of standards including ISO/STEP, IAI/IFC, and CORBA.

3.2 PROCAT-GEN

Product Catalogue for Global Engineering Network Intelligent Access Library, (PROCAT-GEN) was part of the GEN Project, an European initiative with academic and industrial cooperation. It established the market potentials for electronic delivery of product catalogues and developed prototype catalogues within the integrated, platform-independent GEN infrastructure. The GEN infrastructure embraces the Application Service Provider (ASP) delivery model and specifies the implementation of search by keyword, classification codes, and parameters (Radeke, 1999). Java, XML and CORBA are some software development technologies utilised to support interoperability.

3.3 ACTIVE CATALOG

The Active Catalog project, undertaken by Information Sciences Institute at University of Southern California, was aimed at improving the design environment by finding and using product information in building design applications within heterogeneous, internet-based, distributed computing environment (Cheng, 1997). The resulting prototype addressed the limitations of keyword search by implementing a knowledge-based search engine, which utilises domain knowledge in interpreting meanings of high-level user queries. Platform-independent data representation format, modular or component-based software design with well-defined interfaces and platform-independent implementation language are some of the attributes of the system used to facilitate reusability of information and interoperability across systems. CORBA and Java language were extensively used for their platform-independence and object-orientation.

3.4 ARROW

Advanced Reusable and Robust Object Warehouse (ARROW) project is a UK initiative carried out by Building Research Establishment and Visual Technology with support from Department of Environment (Newnham et al., 1997). It built on the COMBINE project (Augenbroe 1994) using an extended IFC model and purposed to implement a knowledge-based query handler with intelligent agents. The prototype application demonstrated integration with distributed manufacturer databases and CAD systems (Newnham and Amor, 1998; Amor and Newnham, 1999).

3.5 RINET

The RINET project is part of the VERA project being undertaken by VTT Building Technology, Finland (Hannus and Pirhonen, 2000). Like ARROW, it demonstrates the feasibility of an Internet-based, single point of entry to product information in distributed manufacturer databases. The prototype, implemented with C++, Java technologies and ODBMS, supported search by parameters, classification codes, and keywords. An authoring tool developed in C++ populates the product database from manufacturers' websites. A product data server along with an index database services user requests from Java applets running in Web browsers.

4 PRODUCT INFORMATION DELIVERY: CURRENT PRACTICES

This section explains current practices in electronic delivery of product information, particularly over the Internet. It highlights the possible search methods in Section 4.1 and provides a brief summary of the capabilities of existing systems in Section 4.2. The shortcomings of the systems were identified and solutions were proffered in Section 4.3. The rest of the paper addresses various parts of the solution.

4.1 SEARCH METHODS

Four search scenarios can be readily identified including search by keywords, classifications, parameters and navigation.

SEARCH BY KEYWORD

Keywords, or significant descriptive words can be associated with a product or group of products to initiate a database search, for example, a search for "window". This method is widely used by Internet search engines for its ease of implementation and use. This method is however inadequate for construction products because keywords are language dependent and domain specific. Considerable user time can be wasted weeding out unwanted information from a search result. Virtually all the existing online product information sources implement this method.

SEARCH BY CLASSIFICATION

This method requires the use of some classification code corresponding to products of a specific grouping. A product can be mapped to multiple classification codes. Some examples of classification systems are:

- Construction Index/Samarbetskommitten for Byggnadsfragor (CI/SfB), a Scandinavian system of classification widely used in the UK;
- Construction Specification Institute (CSI) used in the US;
- Electronic Product Information Co-operation (EPIC) employed in Europe;
- Coordinated Classification System (CCS) in Australia; and
- BSAB 96 in Sweden.

Many online sources of product information provide at least one classification system depending on their target users. Others categorised products according to some arbitrary or in-house system.

SEARCH BY NAVIGATION

This search method involves following hyperlinks from one product to related products, manufacturers or the industry knowledge base such as applicable standards, and performance information. It can provide fast access to all information relating to a product within user control due to avoiding information overload. Support for search by navigation is considerably limited in current implementations of online product libraries.

SEARCH BY PARAMETERS

This method requires that specific functional characteristics, or requirements, be specified as attributes of the product being sought. Values can be restricted or relaxed to narrow or expand the search criteria. This is the most intuitive way to search for products because rarely would a designer begin a search for “door” without having some acceptance criteria or attributes to determine the suitability of the “door”. This method however is rarely supported because current implementations of online product libraries seek to replicate functionalities supported by the paper catalogues, which cannot support search by parameters. In addition, there is no agreed vocabulary among suppliers to facilitate search by parameters.
Product models provide definitions of properties for classes of product thereby providing common vocabulary for describing them and enabling the implementation of search by parameters. There is ongoing research in Artificial Intelligence towards utilising domain knowledge in improving search methods and using intelligent agents armed with appropriate ontology and conceptual vocabulary for the domain of interest (Clark et al., 2000; Obonyo et al., 2001).

4.2 PRODUCT LIBRARIES ON THE INTERNET

Existing building product websites are built to service the local construction industries. Product information from suppliers within the country are hosted and categorised according to the classification system in use in the country. Even within the same

country the information brokers maintaining the website do not utilise the same classification scheme or information model. The information provided is presented in varied formats as static HTML pages with text and pictures in GIF or JPG formats. For easy download, a PDF version is sometimes provided. A handful of websites provide CAD details in DXF or DWF formats for download and use in most CAD systems. VTT Construct IT research in Finland maintains a comprehensive link and brief description of building product libraries and related projects worldwide[1] and a review of search methods supported and classification system used is maintained at the Department of Architecture, University of Edinburgh[2].
Some websites have started delivering product information as program objects that could execute as Java applets or Visual Basic objects on the user's system. These objects attempt to encapsulate the geometry, cost information and some behaviour to enable them to reconfigure and re-render themselves in response to parameter changes (Emmerik, 2001).

4.3 IMPROVING EXISTING WEB-BASED SYSTEMS

There is a need to improve on existing web-based systems to facilitate the integration of the construction teams and provide product information support throughout the project life cycle. The shortcomings of the existing Web-based systems for delivering product information are identified as follows:

- There is no agreed content, structure and format for presentation of product information. This hampers exchange and sharing of product information for re-use between applications and project teams.
- They are specific to particular localities because of classification systems and languages employed. This is increasingly becoming an issue with international construction teams.
- The search methods implemented are inadequate because they are not tailored towards the way users obtain the information;
- There are too many unconnected sources, which prolongs the search process;
- They lack business continuity hence persistent access to product information because they are largely affected by changes in Internet technologies; and
- They are incapable of feedback and automatic updates because they are largely implemented with Internet “pull” technology where the users make all the requests for information without automatic updates or “push” technology where data is available once the user has registered interest.

All these limitations can be adequately addressed with existing standards for representation of building product data implemented in a multi-tier client/server environment with component-based, object-oriented technologies. Product data representation standards like STEP and IFC provide common definitions for building products. This can incorporate multiple classification system and facilitate advanced search methods. It can also facilitate integration of different sources of product information including supplier databases, manufacturer product data management

[1] http://cic.vtt.fi/links/prodlib.html
[2] http://www.caad.ed.ac.uk/~salih/prodlink.htm

systems (PDM), Enterprise Resource Planning (ERP) systems, and other product libraries, through the provision of open data representation format and Application Programmer Interface (API). A universal data format like XML can ensure business continuity when combined with open component-based software architectures, which facilitate pluggability, that is, the ability to build or scale a system by adding or removing components (Stevens and Pooley, 2000). Messaging components can be incorporated to support one-to-many push model communication of feedback and updates to subscribed clients. Subsequent sections describe the elements of this solution starting with representation standards.

5 REPRESENTATION STANDARDS FOR PRODUCT MODELS

Typically, buildings are operational for hundreds of years outliving the project team and software tools that may have been used in storing all information relating to them. There is also the need to harmonise or integrate product information from heterogeneous sources given the global nature of building products. Standards for the representation of product information address these needs by promoting interoperability across software platforms, hardware systems and providing support for sharing and exchange of information.

Data exchange is the process of extracting, transforming and delivering data, sometimes across disparate systems. It should be independent of data format, syntax, data source or target. Data sharing connotes common data sources serving multiple systems. Data exchange and sharing seeks to eliminate data re-entry, adoption or imposition of one system over another and direct translation. To achieve these, a standard for data sharing and exchange should incorporate metadata management, data transformation (or views), administration and implementations, and support for heterogeneous platforms and data sources.

There are industry standards that are widely adopted though developed by a single company or group (e.g. Rich Text Format (RTF) for documents), and there are *de facto* standards that originate from research and development projects (e.g. DXF for CAD). Open standards have wider reach and acceptability because they are products of large international communities of players with interests in the technologies and industries. They are the essence of interoperability, portability and reusability. Interoperability enables seamless integration of both internal and external systems, portability facilitates platform scalability and avoids proprietary limitations, and reusability prevents expensive translation, transcribing and redevelopment (Cumbers, 2001).

STEP and IFC are open standards resulting from the work of the International Standards Organisation and International Alliance for Interoperability respectively. They incorporate other open standards in their specification including XML specifications from the World Wide Web Consortium (W3C) and CORBA from Object Management Group (OMG). Some other industry standards have been developed to support file exchange for construction and other engineering disciplines. These include IGES, SET, EDIF, VDA-FS, and POSC (Fowler, 1995). All these standards have either been superseded by or integrated into the STEP standard. The Data eXchange Format (DXF) from AutoDesk remains the industry standard for construction engineering but with increasing software vendor support for IFC, an industry-wide acceptance can be expected in the nearest future.

5.1 BUILDING PRODUCT MODELS

Product models provide formal and unambiguous (computer-sensible) representation of real world concepts like products and processes. They facilitate effective communication and seamless inter-working between disparate professionals by providing common terminologies, technologies, and ways of expressing and communicating information. They utilise an open data model, which provides common data representations to enable external programs to read and manipulate data. These models form the basis for automation, customisation, rich searching, and alternative interfaces (Myers, 1998). The need for a shared product model cannot be overemphasised given the fractured information systems prevalent in the construction industry, which has resulted in numerous problems for the industry.

ISO-STEP

The STandard for the Exchange of Product model data (STEP), otherwise known as ISO 10303 - Industrial Automation Systems - Product Data Representation and Exchange, is a family of specifications for the computer-interpretable representation of product data through the product's life cycle in a computing platform-independent manner (ISO, 1994). Product information including design, manufacture, use, maintenance and disposal can be described and stored in a neutral file exchange format to facilitate data sharing among dissimilar software applications.
The STEP architecture is layered with three logical levels namely; physical, logical, and application (Eastman, 1999). This separation distinguishes the logical structure of the information, or model, from the physical storage format and allows the definition of a subset of a complete model for a specific application. The standard utilises reusable constructs called classes, and it can be categorised into five structural groups as shown in Figure 2.

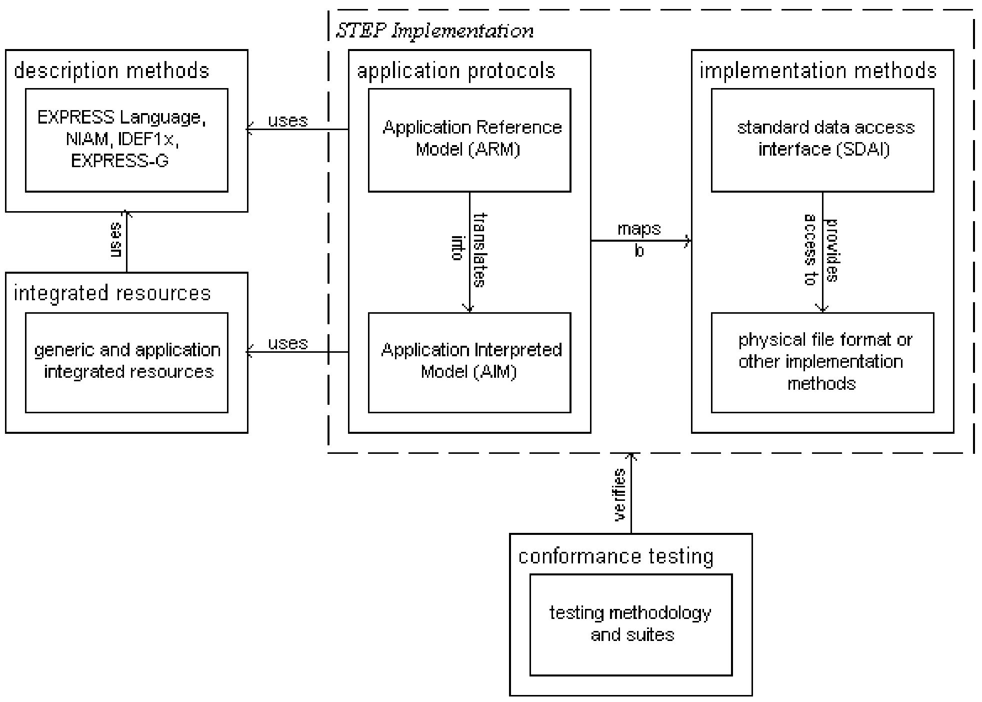

STEP Implementation
description methods
EXPRESS Language, NIAM, IDEF1x, EXPRESS-G
uses
application protocols
Application Reference Model (ARM)
implementation methods
standard data access interface (SDAI)
uses
translates into
maps to
provides access to
integrated resources
generic and application integrated resources
uses
Application Interpreted Model (AIM)
physical file format or other implementation methods
verifies
conformance testing
testing methodology and suites

Figure 2: Parts of STEP Architecture and their relationship

Description methods are information modelling languages used in specifying the other information models in the architecture. These include NIAM, IDEF1x, EXPRESS and EXPRESS-G. Integrated Resources are reusable conceptual model subsets that provide common requirements of product data for different application areas. Integrated Resources used in different domains are called Integrated Generic Resources, for example, geometry and material properties. Integrated Generic Resources can be extended into Integrated Application Resources to support the needs of a specific industry, for example, building, automotive, electronics, etc.

Application Protocols (APs) are implementable data specifications developed for particular application contexts using the Description Methods and Integrated Resources. It uses two types of models: Application Reference Model (ARM) and Application Interpreted Model (AIM). ARM specifies structures and constraints used to describe the information requirements of the application while the AIM is the interpreted version of the ARM. AIM contains a selection of integrated resources, constrained, specialised and completed to satisfy the information requirements of the ARM.

Application Protocols are combined with Implementation Methods, or standard implementation techniques, to form a STEP implementation. Implementation Methods include the STEP physical file exchange structure and Standard Data Access Interfaces, which provides an Application Programming Interface with language bindings for C, C++, Java and CORBA IDL. Conformance Testing provides conformance testing methodology, and framework and abstract test suites for testing implementations of various parts of STEP. This supports validation, interoperability, conformance, performance, robustness and acceptance tests.

STEP APs have been developed and applied successfully in many engineering disciplines including automotive, aerospace, shipbuilding, systems engineering and process plant particularly oil and gas. AP 225, Building elements using explicit shape representation, and AP 230, Building Structural Frame: Steelwork, were specifically targeted at the construction industry. The Building Construction Core Model (BCCM) was proposed as integrated resources for use by application protocols within building construction and for use in exchanging information between heterogeneous computer applications used by multiple disciplinary participants in the building construction domain but the project has been superseded by the IFC project.

IAI-IFC

The International Alliance for Interoperability (IAI) defines data structures, called Industry Foundation Classes (IFC) to support a shared project model for data sharing across applications specifically for the AEC+FM industry (IAI, 2001). IFCs represent building products (and abstract concepts like space, organisations, and processes) by describing their information requirements in a neutral computer language, the EXPRESS modelling language[3]. It therefore facilitates the exchange and sharing of information among the numerous homogenous and heterogeneous software applications employed in the multidisciplinary construction industry.

The IFC architecture provides a modular structure for the development of model components (IAI, 2000). Though it has a three-year release cycle like the STEP standard, it is more responsive to change due to its fewer architectural constraints and annual releases. The current release, IFC 2x, has four conceptual layers, including

[3] IAI actually employ all the STEP description methods.

resource layer, core layer, interoperability layer and domain layer, representing the organisation of the model schema. Classes in each layer are organised to ensure modularity for reusability and ease of maintenance.
The resource layer consists of classes representing objects that are independent of application or domain need, for example, geometry, cost, and measure resources. The core layer classes represent concepts, relationships and roles that can be specialised for interoperability and specific domain requirements. The interoperability layer defines concepts and objects common across multiple applications or industry domains represented by the domain models. Finally, domain layer provides models tailored for a specific industry or application, for example, architecture, electrical, HVAC, or facility management.

ISO-PLIB

Basic to the management of product information is the management of information about and references to parts or components that constitute the product. The ISO 13584 – Parts LIBrary (PLib) standard, developed by the same ISO subcommittee for STEP, provides a series of specifications to facilitate unambiguous, application-independent representation and exchange of all technical data about parts. PLib reuses STEP capabilities for the representations of the parts defined in the PLib library and employs EXPRESS as the data specification language in addition to STEP implementation methods including STEP Physical File Format and SDAI. Pierra et al (1998) provides a detailed introduction to the PLib standard.

5.2 IMPLEMENTING EXPRESS-BASED DATA MODELS

Common facilities for developing physical level implementations of EXPRESS models are provided in STEP SDAI and employed by the IFC specification. Both specifications have embraced XML as the physical format to replace the STEP physical format. Irrespective of physical formats or storage media, SDAI provides the same API for accessing the data repositories.
Four implementation levels, including passive file transfer, active file transfer, shared database access and integrated knowledge base, can be identified. Passive file transfer is the prevalent method and it involves the use of pre-processors for encoding and decoding the model data for the sending and receiving systems respectively. Active data transfer goes further by performing some analysis before sending. Shared database access extends the data exchange capabilities by storing the model data in an underlying database or repository for common access through a standard interface while the integrated knowledge base is envisioned to incorporate an intelligent design environment combining knowledge systems and artificial intelligence.
Implementing an EXPRESS data sharing system requires significant planning and software development effort. The system would need to provide functions for creating, destroying, accessing, updating and querying EXPRESS-data. It would model specific functions like translation, transformation, and rendering, in addition to low-level software functions like memory management and compilation. Tools for browsing model instances, validating constraints, visualising geometry and developing models would also be needed. Some tools already exist for implementing EXPRESS-data transfer and commercial strength EXPRESS-data repositories for shared database access are available from providers like EPM Technology, ProSTEP, STEP Tools Inc, and EuroSTEP Group.

6 IMPLEMENTATION OF INTEGRATED PRODUCT LIBRARIES

Product libraries must be implemented as an overlay to STEP or IFC standards because they support the product data modelling approach and are well suited to the needs of integration and management of the information resources over the whole life cycle (Wittenoom, 1998). Wittenoom (1998) predicted that technical and product information would proliferate and move from paper-based to on-line presentation. It was also predicted that there would be an increasing trend towards component software solutions in the "open" marketplace brought about by initiatives such as IAI/IFC.
It is worth reiterating at this point that an efficient implementation of product libraries must be built around the Internet as the communication infrastructure. It must also incorporate standardised product models and capabilities for organising product information effectively by different classification systems and representing the information in varied formats and languages. The system should be capable of interfacing with existing industry software systems, product data management systems and other sources of product information. In addition, it should provide an intuitive interface, be easy to set up and start up with minimal investment in hardware, software and training. The system should, as much as possible, be resilient to the ever-changing software technologies in order to secure users' buy-in and justify its development. This section discusses the necessary framework and business technologies for implementing such a product library according to the logical components and delivery model. It proposes a multi-tier client/server architecture delivered as an application service.

6.1 ENTERPRISE CLIENT/SERVER MODEL FOR PRODUCT LIBRARIES

In a logical client/server-computing environment, one or more machines act as server, providing shared resources, some application logic and presentation, and one or more machines act as clients providing some application logic and user interface. Since servers can connect to clients and other servers, and clients can similarly communicate with multiple servers and other clients, it is easier to describe the architecture by the basic elements of the application identified as presentation, logic and data; rather than by functions performed by machines. Shan and Earle (1998) identify five client/server topologies partitioned according to the three functional elements and the task division between the computers designated as server or client. The product libraries must incorporate all five topologies to cater for all usage scenarios as depicted in Figure 3.

Scenario 1: a client searches for product data from a basic Web browser with minimal rendering capability.

Scenario 2: a client using a dedicated GUI application, perhaps running in CAD software, browses the product library.

Scenario 3: a client accesses the product library from a Web browser capable of interacting with product objects or from the GUI application capable of rendering an entire building composed of product objects.

Scenario 4: a client (e.g. supplier) running an application capable of interacting directly with the product data repository to add or update restricted product information.

Scenario 5: an integrated project database interacting with the project data repository to obtain information about products applicable to a project or updates on products being incorporated into a project.

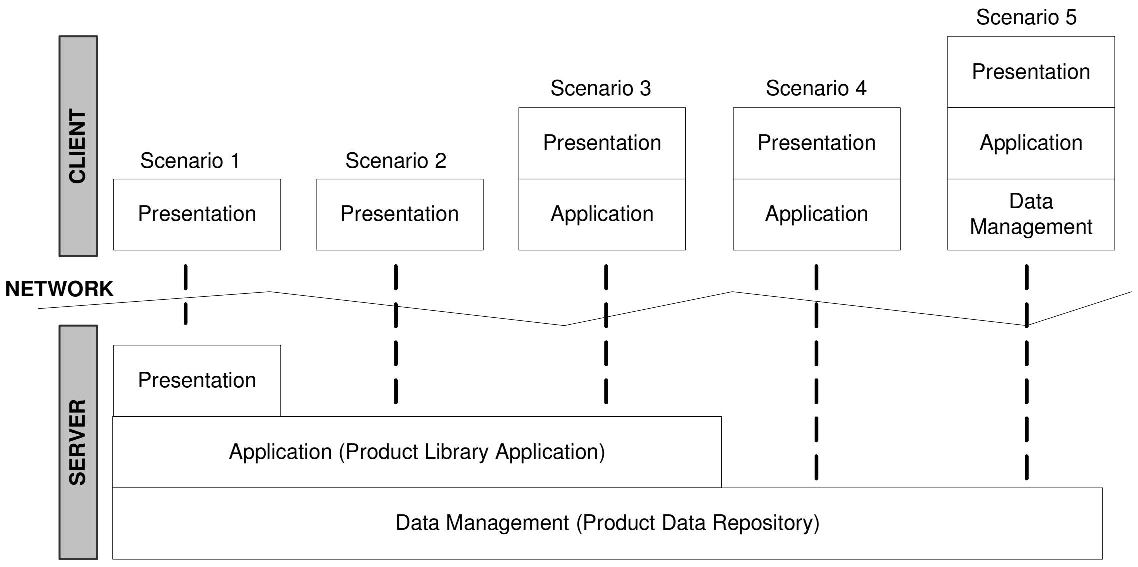


Figure 3: Client/Server Topologies for Product Libraries

To support the above scenarios, a multi-tier client/server architecture is required. This architecture clearly demarcates the presentation layer, logic or middle layer and data layer by providing a number of middle layer components, or middleware, for accessing one or more data layer systems and presenting the data obtained through presentation layer components. It is based on an open system, which is a vendor-independent computing environment consisting of commonly available products that have been designed and implemented in accordance to standardised APIs and protocols. In addition, it leverages component-based and object-oriented software development to offer:

- Portability of application across operating systems and hardware platforms;
- Interoperability across networks and application and computer vendors;
- Flexibility through ability to take advantage of new technologies and changing technologies;
- Scalability to suit different needs and number of users with little or no change to underlying applications; and
- Business continuity for both application provider and users with continuous and easy rollout of upgrades as new business processes, standards and technologies emerge.

This system can adequately support multiple presentation formats and integration with manufacturers' product data management systems and other product data repositories.
Figure 4 presents the proposed multi-tier architecture for delivering integrated product libraries. The data layer, Product Data Repository, captures all the data and translation components. The middle layer, Product Library Application, contains the core

components of the application providing extensible functionalities while the presentation layer consists of components for interfacing with Product Library Clients. The complexity and dynamism of the system discourages the use of simpler, one-tier or two-tier, architectures in which two or more of the logical functions are closely coupled. Shan and Earle (1998) provide more detailed discussion of client/server architectures.

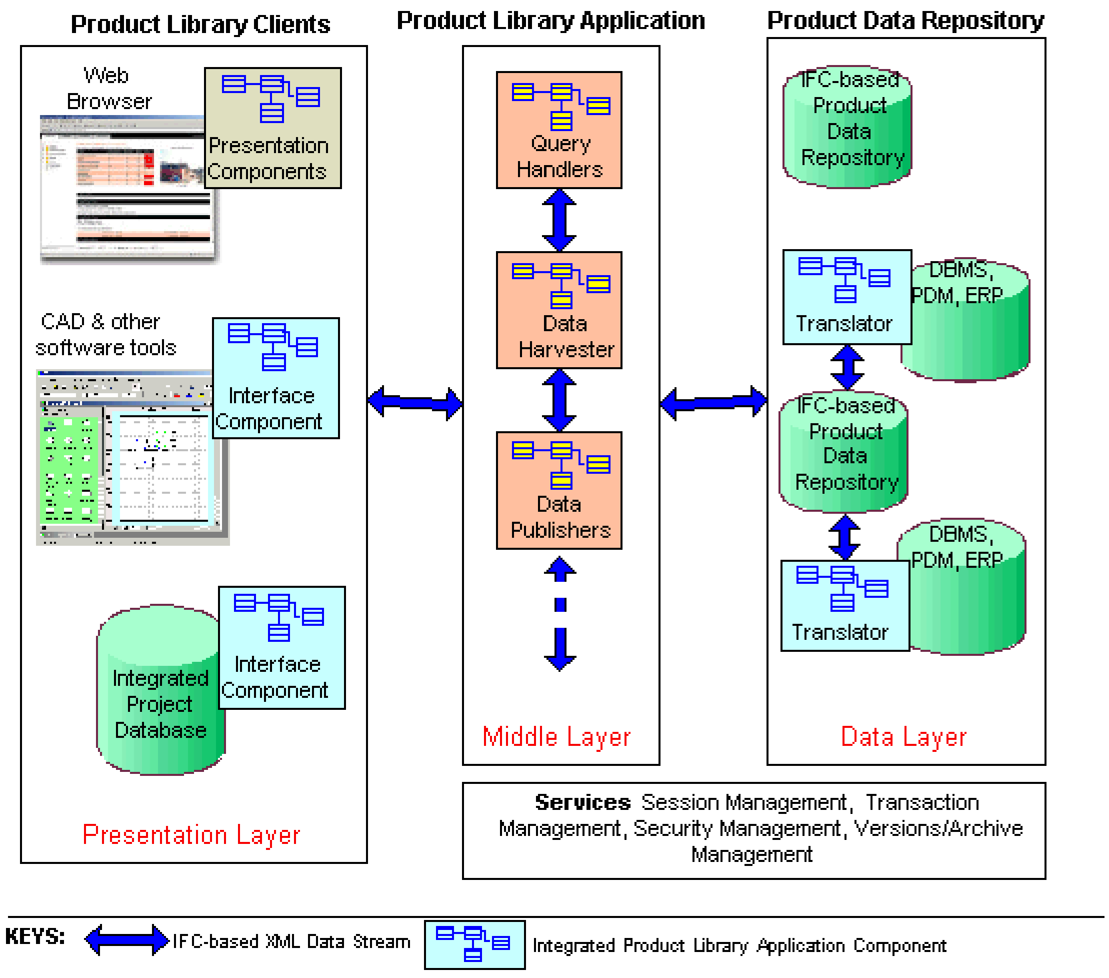

Product Library Clients
Product Library Application
Product Data Repository
Web Browser
Presentation Components
CAD & other software tools
Interface Component
Integrated Project Database
Interface Component
Presentation Layer
Query Handlers
Data Harvester
Data Publishers
Middle Layer
IFC-based Product Data Repository
Translator
DBMS, PDM, ERP
IFC-based Product Data Repository
DBMS, PDM, ERP
Translator
Data Layer
Services Session Management, Transaction Management, Security Management, Versions/Archive Management
KEYS:
IFC-based XML Data Stream
Integrated Product Library Application Component

Figure 4: Proposed Component-Based, Object-Oriented Multi-tier Client/Server Architecture for Implementation of Product Libraries

DATA LAYER

The data layer, also called the data tier, provides the information infrastructure. It would provide XML streams of IFC-based product information extracted from manufacturers' and suppliers' product data management (PDM) systems, database management systems (DBMS) and enterprise resource planning (ERP) systems. XML is preferred as the data-streaming format because it is largely technology independent and it supports on-the-fly content validation, transformation and presentation necessary to support all the identified application usage scenarios. Updates to the repository would be automated to run periodically, initiated by the middle layer Data Harvester or manually

from the source. Where the data is not modelled around IFC, translator components would be deployed to map and transfer the data into XML for storage in the repository.
XML is a simple subclass of the Standard Generalised Markup Language (SGML, ISO 8879), which is fast becoming the standard for information interchange on the Internet and among application. It provides facilities for syntactic validation of documents against formal rules and defines a robust character syntax for representing structured data objects (Kimber, 1999). XML is more widely acceptable than the STEP physical file format, which has been adopted by IFC standards. The IAI defines ifcXML language bindings that would translate EXPRESS Schema to XML Schema and integrate with earlier efforts including the EU-funded Building Construction XML (bcXML) and aecXML initiated by Bentley in North America (Liebich, 2001).
Traditional data models and systems have used relational, network and hierarchical databases but object database management systems (ODBMS) are more suited to engineering applications because they offer flexibility that is otherwise not available. Though ODBMS are easy to use with IFC specifications and modern programming languages, which use the object-paradigm, XML data servers are preferred because they eliminate the need to reformat data for communication between components in the middle and presentation layers. An XML data server is a system usually built on a DBMS that is capable of managing and delivering XML data. Examples of XML data servers include MS SQL Server 2000 from Microsoft and ObjectStore from Object Design. EXPRESS Data Manager from EPM Technology and ST-Repository from STEP Tools Inc. also provide XML support for querying and updating STEP databases.

MIDDLE LAYER

The middle layer consists of components and software tools, which facilitate building and running of multi-tier applications. An extensive middleware infrastructure is required to enable seamless integration and communication that encapsulates all user interactions with the product library as in the identified scenarios.
Some of the required middleware components can be identified as Query Handler, Data Indexer and Data Publisher. The Query Handlers would interpret user queries derived from different search methods and return the result to the appropriate presentation component.
The Data Indexer would subscribe to product information sources and capture published data through the Translator interface components. It would be implemented as a message-oriented component in a publish-subscribe model. In contrast to traditional remote procedure call (RPC), which may require periodic polling of sources of product information, this has the following advantages:

- It supports a failsafe architecture with loose coupling of remote subsystems like product information sources hence allowing for system crashes and scheduled downtimes;
- It provides a more flexible system architecture capable of supporting dynamic addition and removal of product information sources through subscriptions to the product information channel;
- It gives the suppliers of product information control over when to make their data, available for public viewing.
- It provides immediate access to product information while building on the

information broker method of publishing product information.

Similarly, the Data Publisher would be a message-oriented component allowing client applications like other product libraries, project databases, GUI applications, etc. to register interest in specific items of product information through their Interface Components. It would publish notifications, which may include the required information, to all subscribers. Additional components may be added to model business logic and services in a business-to-business request/response interaction such as Request for Information (RFI) and Request for Quotation (RFQ).

The choice of middleware architecture that can support the development of the above architecture, with synchronous RPC and object-oriented methods invocation and asynchronous messaging in point-to-point and publish-subscribe model, is currently limited to Microsoft COM+, Enterprise JavaBeans (EJB) and CORBA Component Model (CCM). The actual choice may be limited to EJB and CCM, because they are open standards with many implementations, rather than COM+, which has only one implementation supported by the owner, though it runs on the pervasive Windows OS. Stal (2000) provides a detailed comparison of CCM, EJB and COM+ platforms.

Microsoft COM+ is a Windows-based, language-independent middleware technology for building reusable components based on the Distributed Component Object model (DCOM), which specifies the structure, API and communication protocols for the components. COM+ provides middleware services such as transaction, security, and messaging. The Microsoft .NET framework improves on this with a common language runtime, which supports unified programming classes to provide a multi-language environment for building, deploying and running Web applications. Morris (2001) highlights the features of .NET.

Enterprise JavaBeans (EJB) architecture is the cornerstone of Java 2 Platform, Enterprise Edition, which is an open standard for deploying cross-platform distributed application. It comprises APIs for accessing distributed computing and network services including database systems, enterprise name and directory services, enterprise messaging systems, email systems and CORBA services. Monson-Haefel (2000) provides in-depth tutorial on developing EJB components.

Common Object Request Broker Architecture (CORBA), from Object Management Group, is an open middleware specification for distributed applications. It is independent of programming languages, operating system platforms, communication protocols, and hardware. The CORBA Component Model (CCM) extends the CORBA object model by defining features and services that enable application developers to implement, manage, configure and deploy components that integrate CORBA services such as security, transaction, persistence, and event notification in a standard environment. An overview of CCM can be found in Heineman and Councill (2001). CCM is modelled closely to the EJB specification and provides standard mappings to both EJB and COM+.

PRESENTATION LAYER

The need to support many interfaces is obvious when product information usage scenarios are considered. Presentation components would be required to deliver textual and graphical product information in web pages. They would also attempt to model product characteristics as much as can be supported by web browsers using Java applets or ActiveX objects for example. Interfacing with heterogeneous construction software tools would leverage the increasing IFC supports in those tools. In addition, project databases would be interfaced with the system to improve productivity by facilitating automated two-way communication between the databases and the product libraries.

6.2 APPLICATION SERVICE MODEL

Given the expected complexity of the application, it would be costly to implement by individual manufacturers. Not only would the hardware, software and IT staff requirement be prohibitive but it would also be a distraction from their core competence. Managing interaction between multiple sources would increase the complexity of the system and affect its productivity. Augenbroe (1998) recognised that a system that provides a single point of entry to all product information irrespective of location, yet supporting transparent awareness of the original sources and costs involved with the sources is the ideal. Radeke (1999) agrees and recommends the application service provider (ASP) model for delivering that solution.

An ASP is an organisation that deploys, hosts and manages software applications and data on its own server. Users access the applications over the Internet mostly using the Web browser. This enables manufacturers and suppliers to host their product information for a fee with the ASP and users to access the information from a single source. As mentioned earlier, existing manufacturers and suppliers do not have to change their systems as automatic translator components can be provided along with the service. The implementation of the system and the provision of the services can be adopted by an existing product information broker as an additional service or new product.

In defence of ASP computing for AEC industry, Unger (2001) identified lower total cost of ownership as the main reason why ASP will dominate the industry. This model does not only have the potential to lower cost and increase productivity but also to:

- Reduce time-to-benefits ratio for manufacturers and suppliers;
- Eliminate problems of upgrades to applications and hardware;
- Offer cost predictability; and
- Encourage global industry standardisation.

ASP models are gaining popularity with the evolution of e-business and the maturity of network infrastructures. The ASP Consortium is addressing security and other issues regarding this model.

7 SUMMARY AND DISCUSSION

The identified shortcomings of current methods of delivering product information and past projects can be classified according to their limitations in sourcing or acquiring product information, searching capabilities, level of integration with existing and emerging construction tools and mode of delivery. The proposed architecture, by employing product-modelling approach and multi-tier architecture including translation service for legacy data, component-based, message-oriented middleware, and multi-scenario (and possible multimedia) access, addresses these limitations and offers additional advantages as enumerated below:

1. It grants manufacturers, suppliers and other providers of product information the control over their information allowing them to make the information

available when they desire thereby supporting current processes rather than imposing a drastic changeover procedure.

2. It supports enterprise application integration through interface/translation components provided as a service thereby preserving the benefits of past investments in hardware and software.
3. It improves efficiency and increases productivity in publishing product information.
4. It can support versioning and archiving of published product information thus ensuring persistent access to the rather volatile product information.
5. It incorporates feedback thereby supporting learning. Users can learn from others' experience regarding a product and manufacturers can obtain necessary information for future improvements.
6. By separating business logic and application components from presentation components, it can support multiple usage scenarios and multimedia access as may be required in the future.
7. Longevity, reliability and maintainability of the product library are better assured through the adoption of open standards and a centralised approach to upgrades through the ASP model.

This paper has provided a high-level description of the proposed system from the product models to the architectural overview. It has highlighted the components that would comprise the system and how they would interoperate to achieve the improvements sought. There are however some barriers and technological constraints, which have to be overcome to make the architecture a reality. The industry comprises many small players to whom the cost of entry, usage and learning new systems, however small, is unsettling. Content providers may also be hindered by cost and the need to differentiate their products, which is undoubtedly limited by standardisation. Representing all required attributes of all products from all suppliers so they can be found through a single search that traverses local and distributed sources (mapping classification systems as it progresses), and aggregates the result for multilingual, multimedia environments would be a formidable technological challenge. There is a parallel project looking into the barriers and enablers of the commerce aspect[4] while this project continues to address the technological constraints.

8 CONCLUSION

It is evident that for product libraries to be acceptable to suppliers and users of product information they should be globally accessible, affordable and offer business continuity. In addition, they should be flexible, integrate with existing and emerging construction industry tools. This paper has described the necessary infrastructure for implementing product libraries that meet users' expectations, including existing software architectures and supporting technologies, standards for representation of whole-life information of construction products, and application delivery model.

[4] Project titled *Business Process Implications of eCommerce in Construction*, being undertaking in Centre for Innovative Construction Engineering, Dept. of Civil and Building Engineering, Loughborough University.

Some of the requirements for product libraries can be gathered from past projects and feasibility studies. However, additional information is required to cover all functional and non-functional requirements, which would ensure user satisfaction and take product libraries away from simply creation and acquisition of product information into sharing, exchange and modification of this information to reflect new knowledge and insight, hence intelligence. The next phase of this project would refine the requirements through involvement of typical end-users. Subsequently prototypes would be developed, tested, and refined towards developing commercial-strength integrated product libraries.

9 REFERENCES

Amor, R. and Newnham, L. (1999), CAD interfaces to the ARROW manufactured product server, in G. Augenbroe and C. Eastman, eds, Proceedings of the Eighth International Conference on Computer Aided Architectural Design Futures [ISBN 0-7923-8536-5] Atlanta, USA, Kluwer Academic Publishers, pp. 1-11.

Anumba, C. J., Bouchlaghem, N. M., Whyte, J. and Duke, A. (2000), 'Perspectives on an integrated construction project model', International Journal of Cooperative Information Systems 9(3), 283-313.

Augenbroe, G. (1994), An overview of the COMBINE project, in R. J. Scherer, ed., First European Conference on Product and Process Modelling in the Building Industry, Dresden, Germany. Copy available at http://erg.ucd.ie/combine/papers.html.

Augenbroe, G. (1998), Building product information technology, Executive white paper, Georgia Institute of Technology, Construction Research Center, Atlanta, GA 30332-0159.

BSI (1991), Glossary of building and civil engineering terms. Technical Report BS 6100-2.6:1991, British Standards Institute, London, UK.

Cheng, H. H., ed. (1997), Active Catalog: Searching and Using Catalog Information in Internet-Based Design, Sacramento, California. Copy available at http://www.isi.edu/active-catalog/asme-97.ps.

Clark, P., Thompson, J., Holmback, H. and Duncan, L. (2000), Exploiting a thesaurus-based semantic net for knowledge-based search, in 17th National Conference on Articial Intelligence and 12th Conference on Innovative Applications of Articial Intelligence (AAAI/IAAI'2000) [ISBN 0-262-51112-6], Austin Texas, USA, AAAI Press (Distributed by MIT Press), California, USA, pp. 988-995.

Coyne, R., Lee, J., Duncan, D. and O S. (1999), Managing product information on the world wide web, Technical report, Department of Architecture, University of Edinburgh, UK. A report from an EPSRC - Multimedia and Networking Applications Programme, sponsored project, available at http://www.caad.ed.ac.uk/Coyne/ProductCatalogReport/.

Cumbers, S. (2001), 'Open source for open government', Visual Systems Journal October, 14-15. Part published as Letter to the Editor in The BCS Computer Bulletin, September 2001, pp. 5-6.

DETR (1996) Construct-IT: bridging the Gap, Scoping study for the Construction Industry Knowledge Base, Construction Sponsorship Directorate, Department of the Environment, HMSO, UK.

Eastman, C. M. (1999), Building Product Models : Computer Environments Supporting Design and Construction, CRC Press LLC, 2000 N.W. Corporate Blvd., Boca Raton, Florida 33431, USA. ISBN 0-8493-0295-5.

Egan, J. (1998), Rethinking construction, Technical report, Department of Environment, Transport and the Regions, HMSO, London. The report of the Construction Task Force to the Deputy Prime Minister, John Prescott, on the scope for improving the quality and eÆciency of UK construction.

Emmerik, M. V. (2001), Product-Center: An integrated environment for selection, configuration and procurement of building materials via the internet, Technical report, Bricsnet, Portsmouth, USA. Copy available at http://corporate.bricsnet.com/about/documentation/ProductCenterWhitepaper.pdf.

Fowler, J. (1995), STEP for Data Management Exchange and Sharing, Technology Appraisals Ltd., Twickenham, UK. ISBN 1-871802-36-9.

Hannus, M. and Pirhonen, A. (2001), 'RINET - building product database', Project descriptions and demonstrations are available at the project website: http://cic.vtt./projects/rinet/index.html. Last visited: 12/12/01.

Heineman, G. T. and Councill, W. T. (2001), Component-Based Software Engineering, Addison-Wesley; ISBN 0201704854. Chapter 38 - An Overview of CORBA Component Model.

IAI (2000), IFC Technical Guide, International Alliance for Interoperability. Copy available at http://www.iai.org.uk/documentation/IFC 2x Technical Guide.pdf.

IAI (2001), 'Introduction to International Alliance for Interoperability', available at IAI UK Chapter website http://cig.bre.co.uk/iai uk/, last visited: 12/12/01.

ISO (1989), Building and civil engineering vocabulary - Part 1: General terms, Technical Report ISO 6707-1:1989, International Standards Organisation, Geneva, Switzerland.

ISO (1994), Industrial automation systems and integration - product data representation and exchange - part 1: Overview and fundamental principles, Technical Report ISO 10303-1, International Standards Organisation, Geneva, Switzerland.

Kimber, W. E. (1999), XML representation methods for EXPRESS-driven data, Technical Report GCR 99-781, Department of Commerce, Technology Administration, National Institute of Standards and Technology (NIST), USA. Available online at http://www.nist.gov/sc4/wg qc/wg11/n095/nistgcr99-781.pdf.

Latham, M. (1994), Constructing the team, Technical Report ISBN 0-11-752994-X, Department of the Environment, UK.

Liebich, T. (2001), XML schema language binding of EXPRESS for ifcXML, Technical Report MSG-01-001(Rev 4), International Alliance for Interoperability. Copy available at http://www.iai.org.uk/IFCXML.htm.

Lockley, S., Rombouts, W. and Plokker, W. (1994), The COMBINE data exchange system, in European Conference on Product and Process Modelling in the Building Industry, Dresden, Germany.

Monson-Haefel, R. (2000), Enterprise JavaBeans, 3rd edn, O'Reilly Publishers, UK. ISBN 0596002262.

Morris, P. J. (2001), 'What's good about .NET?', Visual Systems Journal, December pp. 14-17.

Myers, B. A. (1998), The case for an open data model, Technical Report CMU-CS-98-153, Human Computer Interaction Institute, School of Computer Science, Carnegie Mellon University, Pittsburgh, USA. Copy available at http://reports-archive.adm.cs.cmu.edu/anon/1998/CMU-CS-98-153.pdf.

Newnham, L. and Amor, R. (1998), Translation of manufacturer's product data for the ARROW product search system, in ECCPM 98 - The Second European Conference on Product and Process Modelling in the Building Industry, BRE, UK, Building Research Establishment (ISBN 1-86081-249-X), pp. 405-412.

Newnham, L., Parand, F., Amor, R. and Nisbet, N. (1997), The ARROW framework for a building object warehouse, in R. Drugemuller, ed., CIB W78 Workshop on Information Technology Support for Construction Process Re- Engineering, IT-CPR 97, International Council for Building Research Studies and Documentation, pp. 319-328.

Obonyo, E., Anumba, C. J., Thorpe, A. and Parkes, B. (2001), Agent-based support for electronic procurement in construction, in M. Mohammadian, ed., International Conference on Intelligent Agents, Web Technologies, and Internet Commerce (IAWTIC), Las Vegas, USA, University of Canberra, Australia [ISBN: 0858898470], pp. 268-279.

Pierra, G., Sardet, E., Potier, J. C., Battier, G., Derouet, J. C., Willmann N., and Mahir, A., Exchange of component data : the PLIB model, standard and tools, in CALS Europe '98 Conference, Paris, France, pp. 160-176.

Radeke, E. (1999), Final report of global engineering networking intelligent access libraries (GENIAL), ESPRIT Project 22.284, Technical report, The GENIAL Consortium.

RIBA (2001), 'The 2001 survey on information services in the UK construction industry', RIBA Companies Ltd., London. Copy available at http://www.ribac.co.uk/images/survey2001.pdf.

Schenck, D. and Wilson, P. (1997), Information Modeling : The Express Way, Oxford University Press, UK. ISBN: 0195087143.

Shan, Y.-P. and Earle, R. H. (1998), Enterprise Computing with Objects: From Client/Server Environments to the Internet, Object Technology Series, Edited by Grady Booch, Ivar Jacobson and James Rumbaugh, Addison-Wesley, USA.

Stal, M. (2000), Component technologies for the middle tier: CCM, EJB, COM+, in ACM Conference on Object-Oriented Programming, Systems, Languages and Applications, Minneapolis, Minnesota USA, OOPSLA 2000.

Stephens, J. (1999), Virtual enterprises using groupware tools and distributed architectures: Public nal report, Technical Report PFR-01, ESPRIT 20408 - VEGA.

Stevens, P. and Pooley, R. (2000), Using UML - Software Engineering with Objects and Components, Object Technology Series, updated edn, Addison Wesley Object Technology Series, ISBN 0201648607.

Unger, S. (2001), 'Why ASPs will dominate AEC', ConstrucTECH vol. 4(No. 10). copy available at http://www.constructech.com/printresources/v4n10/ v4n10034.htm.

Wittenoom, R. (1998), Automating realization of integrated project models, Technical report, AusSTEP Consultants, Dalkeith, Australia URL: www.ausstep.com.

Zarli, A. and Poyet, P. (1999), Distributed architectures and components for the virtual enterprises, Technical report, Centre Scientique et Technique du Batiment (CSTB), Sophia Antipolis, France (zarli/poyet@cstb.fr).

APPENDIX B PAPER 2

Owolabi, A., Anumba, C. J. & El-Hamalawi, A. (2003), “Architecture for implementing IFC-based online construction product libraries”, *ITcon* **Vol. 8**, 201–218. Special Issue IFC - Product models for the AEC arena, Edited by: Vaino Tarandi. Copy available online at http://www.itcon.org/2003/15.

ARCHITECTURE FOR IMPLEMENTING IFC-BASED ONLINE CONSTRUCTION PRODUCT LIBRARIES

ABSTRACT: Construction product information providers have responded to the demand for electronic delivery by providing online access, CD-ROMs and DVDs but these solutions have limited usability and are generally incapable of supporting prevalent and emerging industry practices. The product library implementations attempt to replicate the functionalities of the paper versions, which serve for independent specification and procurement but gives little thought to teams and tools integration through support for automated information exchange and sharing. The IFC standard provides common terminologies, technologies, syntax and semantics necessary to address present and future compatibility and integration issues, hence IFC-based implementation of product libraries have good prospect for meeting the industry requirements. This paper reviews current product information delivery methods and examines the applicability of the IFC and other standards. The requirements for IFC-based construction product libraries are identified and architecture for realising the requirements was presented.

KEY WORDS: IFC, product libraries, integration, data exchange and sharing, multi-tier architecture.

1 INTRODUCTION

Printed catalogues and brochures are widely used within the construction industry as the primary source of product information to facilitate the specification and procurement of products. Useful product information including design, properties, costs, availabilities and suppliers detail can be garnered from these periodic publications and employed in the construction process. The industry practices, influenced by information and communication technologies (ICTs), have created demands for electronic versions of the information. Product information brokers, manufacturers and suppliers have responded by providing online access, CDs and DVDs.

Electronic product catalogues attempt to replicate the functionalities of the printed ancestors and have achieved that goal and slightly exceeded it but paper catalogues are still more popular with industry practitioners (RIBA, 2001). This is an aberration particularly when compared with the rate of uptake or success of other electronic tools employed in the industry. The reason for this could be the traditional slowness of the construction industry to embrace new practices or the limitations of the implementations.

This paper focuses on usability problems of the existing systems by identifying the shortcomings and examining possible solutions. It goes further to propose application architecture and to examine implementation issues for a prospective solution.

Section 2 provides definition of terms. Section 3 summarises the limitations of existing system and section 4 highlights the part standardisation can play in addressing the identified problems. Section 5 justifies the product-modelling approach and the choice of IFC. It backs this with a cursory look at other local and international projects based on the product modelling approach. In order to highlight some implementation requirements, sections 6, 7 and 8 review the IFC architecture, classification systems and search methods respectively. The systems requirements are detailed in section 9 while section 10 describes two application architectures. The proposed implementation architecture is presented in section 11 and the conclusions in section 13.

2 DEFINITIONS

The following definitions have been adopted or derived for the purpose of this paper:

A *product* is any object, manufactured, supplied or created, for incorporation into an AEC/FM project where a project is some engineering activity leading to a product including objects that are created indirectly by other products.

Product information comprises of catalogue and technical information, specification, drawings, samples, costs and discounts, sources of supply and delivery, test reports, installation, operation and maintenance instructions, and disposal or recycling requirements. Performance information and user feedback could also be incorporated.

A *product model* is a digital representation of a real world object held to facilitate the unambiguous transfer of the information between computer systems and to support information sharing (Eastman, 1999).

A *product library* is any virtual or physical place or media, which contains systematically arranged digital information about physical, manufactured, supplied or created objects and provides access to users and owners of such information. This definition excludes the traditional methods of sourcing and maintaining product information - the printed catalogues.

Architecture as defined by ANSI/IEEE (2000) is "the fundamental organisation of a system embodied in its components, their relationships to each other and the environment, and the principles governing its design and evolution".

3 LIMITATIONS OF EXISTING SYSTEMS

The usability of existing product libraries is blighted by the lack of agreement in content and presentation format. This hampers the exchange and sharing of data for reuse in construction applications. Search methods implemented are limited and the multiplicity and lack of integration in the implementations have complicated the search process. Other problems including data (version) inconsistency and lack of persistent access can be associated with the existing process used in publishing updates. In all, the users' experience has been that of considerable time wastage, frustration and disappointment.
Despite these demerits, the reachability and accessibility offered by the Web implementations continue to attract users as signalled by the proliferation of commercial online product libraries. The authors believe that if the current delivery methods can be improved, better implementations that adequately support prevalent and emerging industry practices can yield a better patronage of online product libraries.

4 STANDARDISATION FOR INTEROPERABILITY

Standardisation addresses the problem of incompatibility of data formats by providing industry-wide syntax and semantic necessary to support automated data exchange and sharing. The International Alliance for Interoperability (IAI) defines the Industry

Foundation Classes (IFC) standard. The IFC is an effort to provide standard for digital representation of *things* to facilitate unambiguous transfer of information between computer systems and to support information sharing within the AEC/FM industry. Based on shared IFC schema definitions, aggregation of information from multiple sources can be implemented to provide an integrated access, and possibly single entry-point to product information.
The success of an IFC-based product library implementation hinges on the industry's acceptance of the IFC standard. In addition to addressing the shortcomings of the existing systems, the popularity of the solution would depend on its ability to integrate with existing systems across platforms and provide future compatibility with the ever-changing software technologies.

5 JUSTIFICATION FOR PRODUCT MODELLING APPROACH

Construction products are international and long lasting. Products may be acquired from several countries and they may be operational for hundreds of years thus outliving the project team and ICT tools employed in creating and storing the information. In addition, the industry is fast migrating from exchange of CAD documents to sharing of both geometric and non-geometric information. This provides the motivation and the justification for adopting an industry-wide product-modelling standard.
Product modelling standards seek to facilitate effective communication and seamless inter-working between disparate professionals by providing common terminologies, technologies, and ways of expressing and communicating information. They utilise an open data model, which provides common data representations to enable external programs to read and manipulate data. These models form the basis for automation, customisation, rich searching, and alternative interfaces (Myers, 1998).
Stouffs and Krishnamurti (2001) argue that standardisation is not the solution to data exchange, particularly in the design process, because it attempts to impose a common semantic model for all to adhere to with attendant restriction to possibly better solutions and impedance to creative new approaches to specific problems. The argument that if everyone adopts the same concepts, vocabulary and language, any data expressed within the language will be accessible to everyone was challenged on the basis of practicality and representational flexibility and extensibility. Stouffs and Krishnamurti (2001) therefore proposed a lexicon model based on syntactic similarity and data exchange by translations.
Howard (2001) likens the notion that standardisation limits design freedom to saying the alphabet limits literary expression. Adopting Stouffs and Krishnamurti (2001)'s propositions would do little to improve the existing complicated, highly-fragmented communication among the project team members with the accompanying omissions, repetitions, confusions, misunderstandings, errors, delays and litigations.

5.1 WHY IFC?

The IFC platform specification, ISO/PAS 16739, provides data structures for the AEC/FM industry shared project model and seeks to enable data sharing across heterogeneous applications by representing building products (and abstract concepts like space, organisations, and processes) and their information requirements in a neutral computer language - the EXPRESS modelling language (IAI, 2000). The use of IFCs,

for example, would enable a window manufacturer to provide its product data in a format that can simply be inserted into a CAD design program with embedded properties, such as dimensions, materials, strength, energy performance, fire rating, code compliance, applicability, cost, availability, and source. Appropriate property data about the window can then be exchanged with downstream applications such as cost estimating and energy analysis (IAI-NA, 2002).
IFC is consistent with and adopts a number of specifications from an earlier and ongoing standard, the STandard for Exchange of Product model data (STEP) (Daley et al., 1999), which has been successfully applied in other engineering disciplines including automotive, shipbuilding, aerospace, and process plant.

5.2 RELATED PROJECTS

Product modelling has been recognised as a basis for computer integrated engineering since the late 1980s. Several EU projects including ATLAS, CIMSteel, COMBI, and COMBINE have developed prototype product model environments to prove the validity and effectiveness of the approach and stimulate further research, standardisation and implementation activities (Turk et al., 1997). The culmination of AEC/FM product modelling efforts in IAI/IFC has created a new breed of implementation attempts. This is exemplified by the Advanced Reusable and Robust Object Warehouse (ARROW) project carried out in the UK by Building Research Establishment and Visual Technology Ltd (Newnham et al., 1997). Online demonstration is available online at http://cig.bre.co.uk/arrow/OSDemo/devVer/mockup/index-frames.html.
Another key project which adopts the product modelling approach is Product Catalogue for Global Engineering Network (PROCAT-GEN) (Faux et al., 1998a), part of the EU-funded GEN project. PROCAT-GEN is exploring the requirements for and building multimedia online catalogue application to support acquisition, extraction and publishing of product information. It aims to support integration of suppliers' existing environments and users' software tools (Faux et al., 1998b). More details and project updates are available at http://www.procat-gen.org/.
Alternative approaches to integrated product libraries, based on proprietary technologies or *de-facto* standards and translators have been developed and are widely deployed in the industry. The most popular online product libraries are more interaction-oriented though some offer product drawings, usually in DXF or DWG format, which can be imported into CAD software and included in a design.

6 IFC ARCHITECTURE - AN OVERVIEW

IFC employs a layered architecture with four conceptual layers, namely resource layer, core layer, interoperability layer and domain layer. Each layer is comprised of modules containing reusable constructs called classes. The resource layer consists of classes representing objects that are independent of application or domain need, for example, geometry, cost, and measure resources. The core layer classes represent concepts, relationships and roles that can be specialised for interoperability and specific domain requirements. The interoperability layer defines concepts and objects common across multiple applications or industry domains represented by the domain models. Finally, domain layer provides models tailored for a specific industry or application, for example, architecture, electrical, HVAC, or facility management. Figure 1 depicts the architecture. A detailed explanation of the IFC architecture is available in IAI (2000).

IFC uses basic data units of the EXPRESS language, entities, attributes and relations, to define fundamental objects like *IfcObject*, *IfcRelationship* and *IfcPropertyDefinition*. These objects are further specialised into classes like *IfcProduct*, *IfcProcess*, and *IfcControl* to form the core classes. Core Layer Extensions specialises core classes into basic objects. For example, *IfcProductExtension* defines objects like *IfcElement*, which is further specialised into *IfcBuildingElement* and *IfcOpeningElement*. Interoperability classes, like *IfcSharedBldgElements*, inherits the core layer classes to provide constructs like *IfcWall*, *IfcSlab* and *IfcWindow*, which are common to different actors and disciplines. Specialised extensions such as *IfcLanding*, specific to disciplines, are derived from lower classes and further distinguished through property sets. Figure 2 shows an annotated inheritance tree for *IfcLanding*.

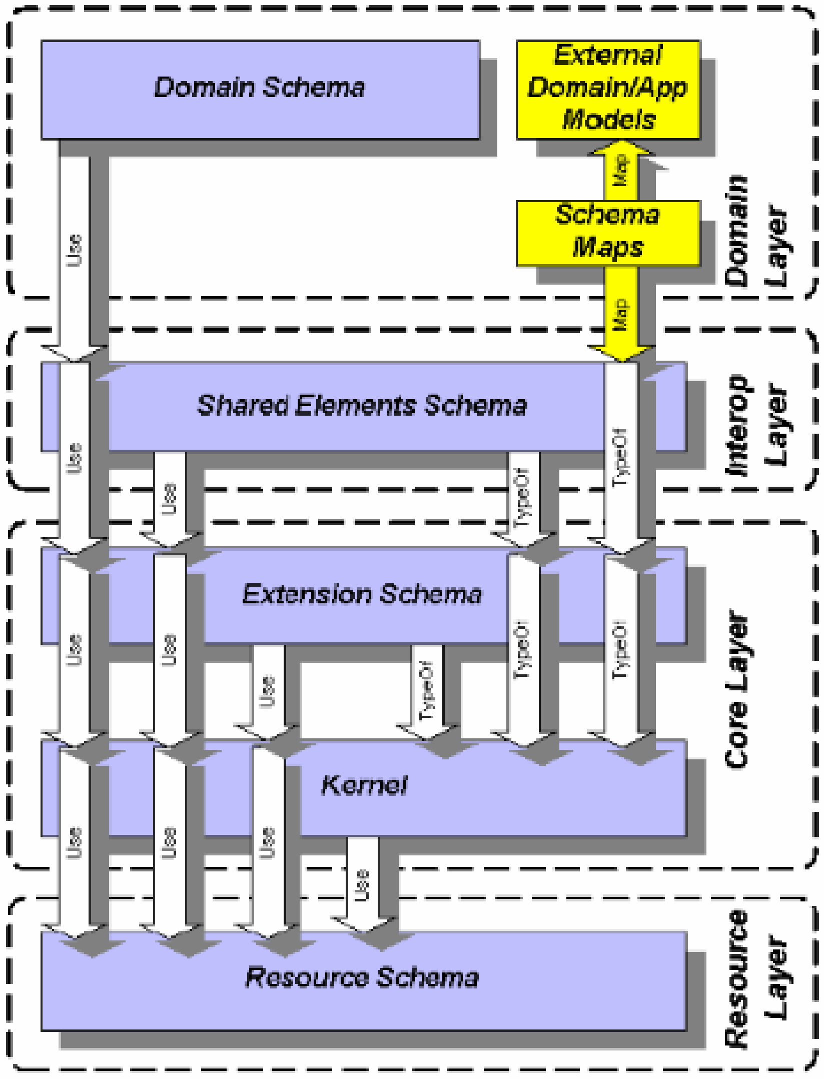


Figure 1: IFC Model Architecture

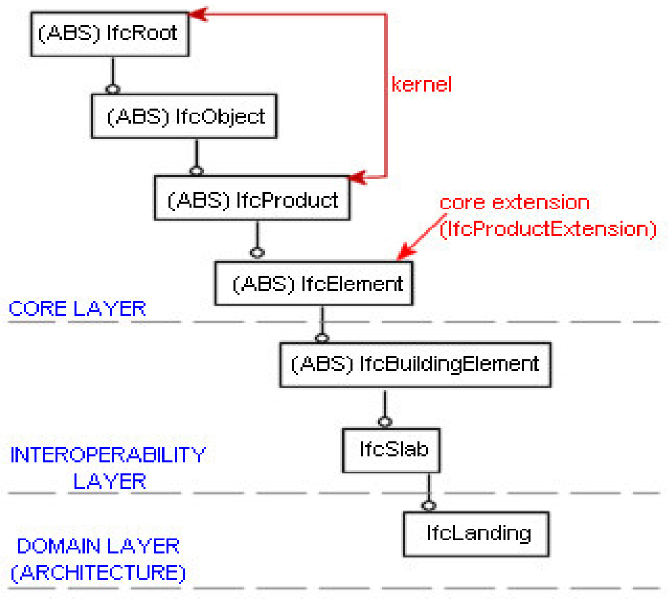


Figure 2: Inheritance Tree for *IfcLanding*

6.1 PROPERTY SETS

Property Sets consists of property definitions, which may be defined generically to form part of the published IFC release or specifically to meet different application requirements. This is particularly important to the implementation of product libraries because they provide the means for differentiation of products without compromising the ability of the applications to exchange or share standard information. IAI provides guidelines for user-defined property sets and is putting considerable effort into maintaining consistency and coordinating these extensions and the explicitly defined property sets (Adachi, 2001; Ekholm, 2002).

6.2 PARTS LIBRARY

The IFC standard is insufficient to implement product libraries. It is necessary to manage information about and references to parts or components that constitute products. The ISO 13584 - Parts Library (PLib) standard (Pierra et al., 1998) caters for this through a series of specification to facilitate unambiguous, application-independent representation and exchange of all technical data about parts. PLib, like IFC, reuses STEP capabilities including the EXPRESS modelling language, STEP Physical File format (Part 21), and Standard Data Access Interface (SDAI).

7 CLASSIFICATION SYSTEMS

The organisation of information is critical to its understanding and effective usage and there are a number of systems used to classify construction information today. National and international classification systems have been developed and employed to facilitate electronic exchange of construction information between project partners and suppliers. While search by keywords can be implemented to locate desired information, the existence of some agreed vocabularies, glossaries or thesauri is essential to data exchange and sharing.

The ISO/CD 12006-2 (ISO, 1997) classification system is a framework that identifies a set of classification tables based on views of construction works. The ISO PAS 12006-3 (ISO, 2000), introduces object-oriented information organisation by defining classes and their relationships and properties. Other national and international classification systems include MasterFormat™ (CSI, 1995), UniFormat™ (CSI, 1998), Uniclass (CPIC, 1997), EPIC (EPIC, 1999), OCCS (OCCS, 2001), SfB (Ray-Jones and Clegg, 1991), LexiCon (Woestenenk, 2000), and BSAB (Ekholm et al., 2000). Apart from, but not dissimilar to these classification systems are the product modelling classification systems like IFC and STEP. The harmonisation of the ISO12006-3 with IFC is an on-going task within the IAI-IFC R3 Project. Project information is available at http://cic.vtt.fi/niai/technical/ifc3/R3XM7.html. Reviews and comparisons of subsets of the available classification systems can be found in Ekholm (1999), Wright (1998) and Howard (2001).
It is impractical to map all the classification methods when implementing an integrated product library because of the diversity in their approach but each product can be associated with multiple classification codes. With the progressive convergence of national and international classification systems a single acceptable classification system may emerge in the future, but even then, there would be a need to integrate legacy systems.

8 SEARCH METHODS

Currently there are three approaches to finding products that meet users' requirements. These are text-based search by keyword, search by classification code and parametric search. Researchers are working on two additional search methods termed knowledge-based search and performance-based search.

8.1 SEARCH BY KEYWORD

This borrows from the traditional descriptive search, which involves the use of words that can be associated with actual products. It is usually language and domain specific and can benefit from a background implementation of domain thesauri, dictionaries, glossaries and classification systems when interpreting queries. For example, a search for *window* in English or *Fenster* in German should be interpreted similarly within Architecture domain. Kosovac et al. (2000) describes the development of thesauri for the roofing domain and software for eliminating irrelevant terms and extracting key phrases from textual data.

8.2 SEARCH BY CLASSIFICATION

Classification systems are traditionally hierarchical lists of standard elements sometimes with associated classification code. This can foster search by navigating the user through the tree hierarchy with the advantage that user errors, inherent in search by keyword, are eliminated. For example, 08 is the classification code for *Doors and Windows* in the Construction Specifications Institute (CSI) 16 Division format while 08310 is a more specific code for *Access Doors and Panels*. Furthermore, classification systems provide means of accessing data through different sets of views, for example, users can navigate to relevant information by the process model.

8.3 PARAMETRIC SEARCH

In parametric search, or attribute-based search, specific functional requirements can be specified as attributes of the product sought. These requirements are evaluated and compared with product attributes to determine appropriate matches. Values can be relaxed or restricted to narrow or expand the search as desired. For example, a search for a door with specified finishing, glazing type, fire rating, light transmission, thickness, water permeability, etc.

The lack of common *language* amongst producers of product information has limited the implementation of parametric search until recently. The ARROW (Newnham et al., 1998), PROCAT-GEN (Faux et al., 1998b) and RINET (Hannus and Pirhonen, 2001) projects have implemented or plan to implement parametric search.

This search method is widely recognised as the most intuitive way to source product information because consumers are usually armed with more information for product discovery, comparison and evaluation than other search implementations allow them to input. This does not eliminate the need for consumers to evaluate the product, as it may be impossible to specify some design criteria in terms of product attributes. For this reason, Jain and Augenbroe (2002) argues that attribute-based search are more suitable for the construction phase of the building project when product specification are available than during the early stages of design when the product specifications are being developed.

8.4 PERFORMANCE-BASED SEARCH

The performance of a building object (or product) is defined by Woestenenk (1999) as a measure of the object's reaction to the environment in which it is placed. Performance-based search would extend parametric search by considering the role a product would play in a functional system and by attempting to match the product information with the functional requirements (Jain and Augenbroe, 2002). A method for quantifying, storing and evaluating performance requirements electronically would be required to successfully implement a performance-based search system. There is no known implementation of this search method for construction products but the Active Catalog project (Ling et al., 1997) provides a proof of the concept for electro-mechanical engineering parts library. The prototype, based on a *try-before-you-buy* paradigm, helps the designer to repetitively evaluate candidate components in the context of the system design until a satisfactory product is found.

8.5 KNOWLEDGE-BASED SEARCH

Knowledge-based search exploits a conceptual vocabulary, domain knowledge and Artificial Intelligence (AI) search techniques in interpreting queries and returning best matches. It requires an unambiguous representation of user's requirements to apply domain knowledge to *reason* about the user's request (Clark et al., 2000). This search method can also benefit from categorisation or classification, context-based interpretation and management of user profiles. One emerging knowledge-based search technique is the use of agent technology (Denzinger et al., 2002). The application of this technique in the procurement of construction products is the subject of an ongoing research (Obonyo et al., 2001).

9 SYSTEM REQUIREMENTS

Extensive system analysis approached from the needs of users and drilled down to deployment issues is required before implementing any software. This section presents the functional and operational requirements for product libraries. The high-level functional requirements are discussed with the aid of a use case model.
The goal of the product library is to provide product information that is complete, accessible, and reusable. Completeness is the ability of the information to support all construction phases from feasibility to demolition as shown in Figure 3. Accessibility implies the information is easily and speedily obtainable. Reusability is the degree to which the information can be readily utilised in consumer application environments. The system should be reliable, flexible and available. System reliability relates to the number of bugs in the software and flexibility is degree of responsiveness to changing user needs and technologies. Availability relates to the system's portability and maintainability. It is the ability of the software to run on readily available hardware, operating system, etc.

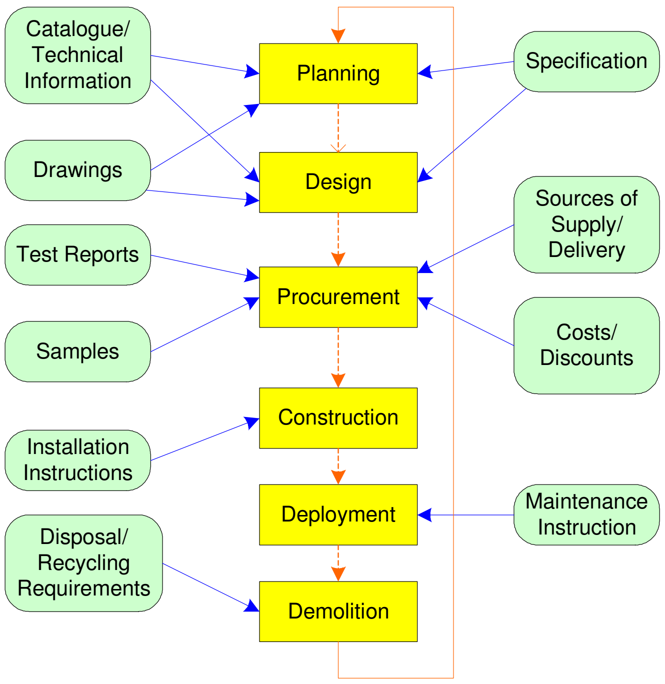


Figure 3: Construction Process and Product Information Requirements

9.1 FUNCTIONAL REQUIREMENT

Use cases document the behaviour of a system from the users' point of view. Figure 4 presents a use case model of the system. It details the high level needs and desired features of the system. *Actors*, represented by stick persons, are the roles that would be played by stakeholders and target users while *use cases* are the tasks they need to perform. This role-based approach would facilitate personalisation and customisation. For example, product information can be presented based on the construction project phase and the role being played by the actor. Specifiers, Designers, Engineers and Managers have been generalised as Registered Consumers. These are in turn

generalisation of specific disciplines not explicitly represented. A single user or a group of users within an organisation or project team may play each actor's role.
All users should be able to interface with the system via standard Web browsers and their specific software applications. *Guest* (unregistered) users should be allowed to register for proper identification otherwise they should be limited to product browsing. *Registered Producer* should be able to publish their product information through Web interfaces or automatically by manual or scheduled running of an application on their local systems. *Registered Consumer* should be able to browse the product information by a variety of methods including from their specific applications. Facilities should also be provided for controlled exchanges between the producers and consumers in the form of updates and messages like feedbacks, request for information, etc. The system should be extensible to accommodation commercial exchanges and future requirements.

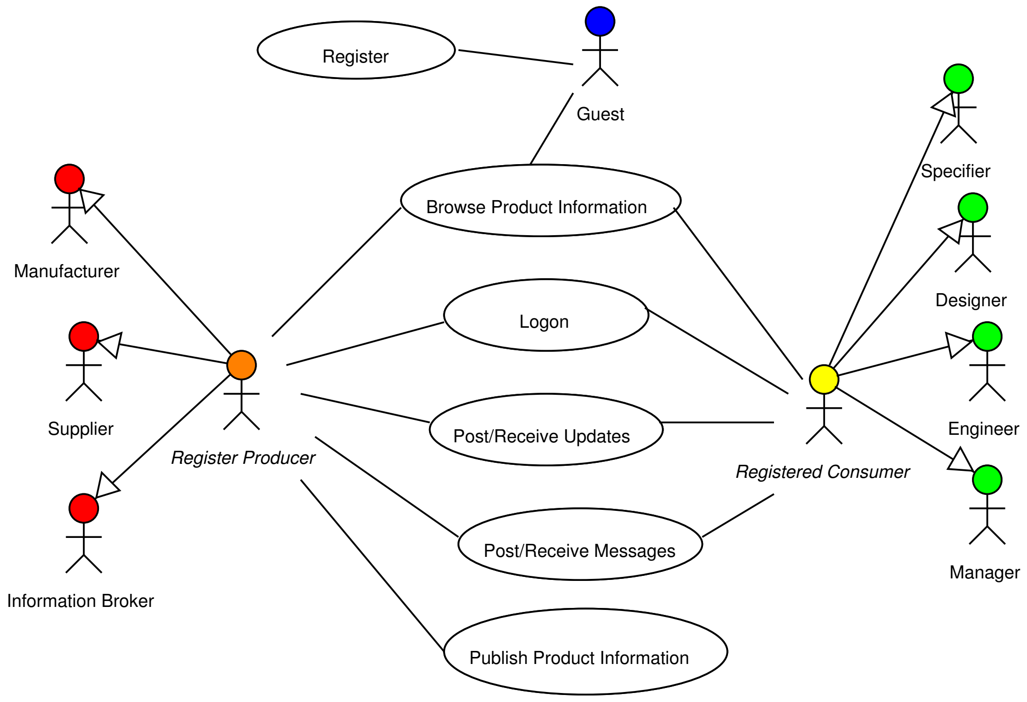


Figure 4: Use Case Diagram for the Product Library

It is not feasible for a single implementation to cater for all producers and consumers. Given that this is a data-intensive application, a central implementation would constitute a bottleneck and create maintenance nightmare. The system should therefore provide the facility for transparent cooperation with other implementations and sources of relevant product information.

9.2 OPERATIONAL REQUIREMENTS

Traditionally, manufacturers, suppliers and information brokers have the responsibility of packaging and delivering product information. Manufacturers and suppliers should retain the ability to choose between implementing their own system and subscribing to information broker. A subscription-based implementation topology that enables implementations to cooperate to satisfy users' requests and eliminate duplication of data is required. This would also enable the formation of dynamic supply chains so that a *closed* group of producers and consumers can be formed around the project teams or businesses.

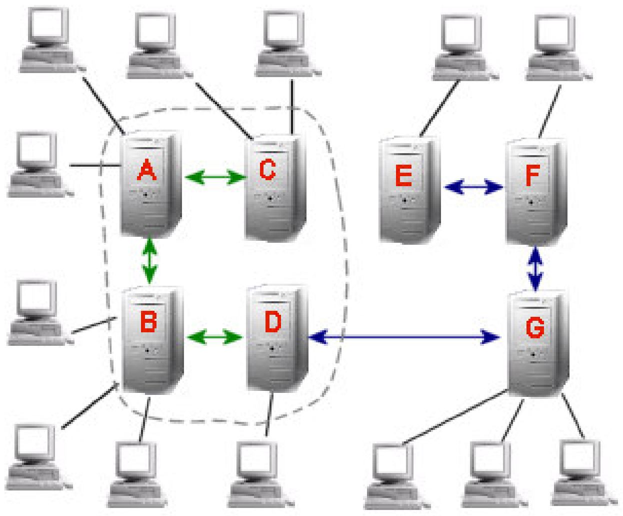


Figure 5: Subscription-based implementation topology

Figure 5 presents the required implementation topology with seven implementations labelled A through G. G can access E because it can access F but it cannot access A, B and C even though it can access D because A, B, and C are in a closed implementation group with D.
Each implementation should maintain a list of implementations it can reach including information on types of products and product information hosted. The list should be updated when subscribing to a new implementation to avoid duplicates. The system should be able to select relevant implementations to satisfy users' requests.
This topology would offer a truly dynamic, distributed network of product information sources that eliminates the need for multiple copies of the same information and is capable of supporting an integrated project process. This is a practical alternative that is more likely to succeed in the industry than the centralized approaches because it provides for gradual changeover and eliminates the access bottlenecks usually associated with single, centralized sources.

9.3 DEPLOYMENT CONTEXT

This section examines the relationship between each implementation to other systems that it would interface with as shown in Figure 6.
The implementation requires interface components for accessing other product library implementations. Interface components are also required for industry software tools, existing producer databases, integrated project databases and other sources of product-related information. The implementation will provide Web clients for consumers and producers in addition to Web-enabled applications for producers.

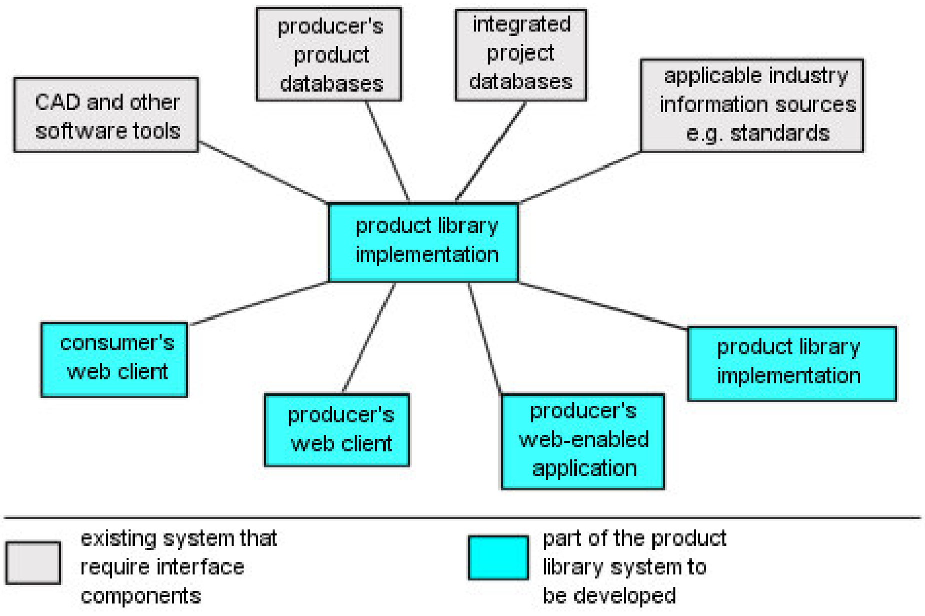

producer's product databases
integrated project databases
CAD and other software tools
applicable industry information sources e.g. standards
product library implementation
consumer's web client
producer's web client
producer's web-enabled application
product library implementation
existing system that require interface components
part of the product library system to be developed

Figure 6: Context for the product library system

10 APPLICATION ARCHITECTURE

A detailed analysis of all the requirements suggests that a combination of application architectures is needed. This section introduces the multitier client-server architecture and the Web Services architecture.

10.1 MULTITIER CLIENT-SERVER ARCHITECTURE

The system requirements and deployment context described above suggest a client-server application scenario whereby a client (or user's machine) running a Web browser or dedicated application sends requests to a server (or product library implementation) over a communication network. The request triggers the execution of a set of programs on the server and a response is returned to the client.
The server architecture that interprets requests is usually component-based. These components are grouped into tiers (as shown in Figure 7) for user interface, application and business logic and backend services such as file, database and communication services. The degree of coupling between these components determines the number of tiers in the server architecture. In one-tier, for example, all three components are closely coupled while they are separate in three-tier architecture. The multi-tier, or n-tier, extends the three-tier architecture to include scenarios where many middle layers cooperate with one another to respond to a request. This is the most applicable scenario to the product library implementation.
Multitier architecture helps to decouple business logic from presentation and data access. It offers increased scalability, performance, network efficiency, and service reusability. It enables a great number of concurrent users and facilitates easier deployment and management of applications. Raduchel (1998) covers the benefits of multitiered architecture in more detail.

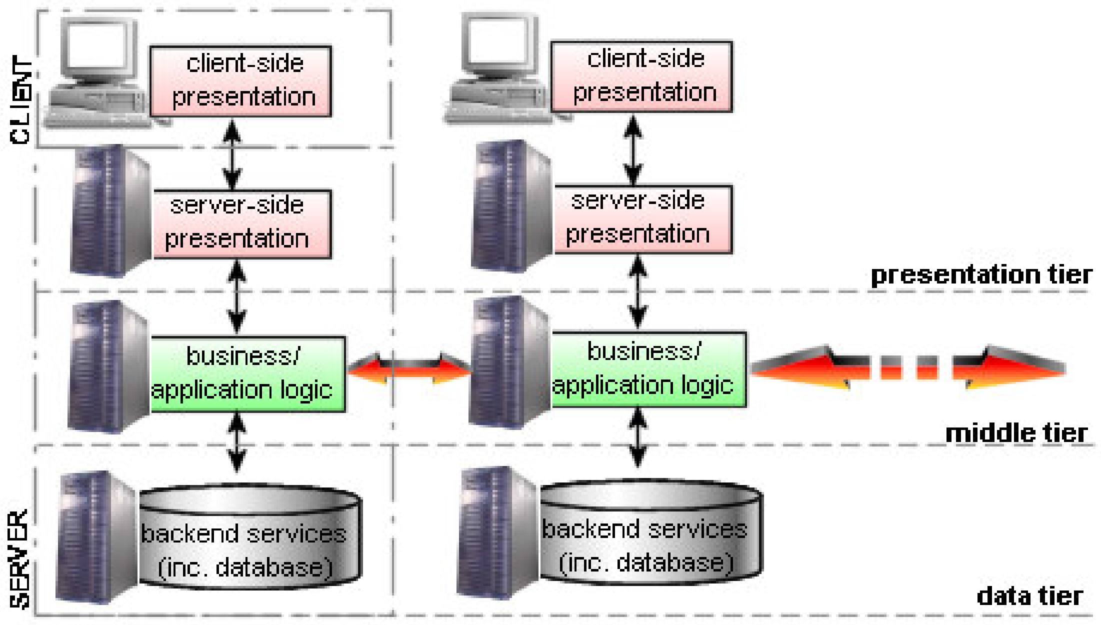


Figure 7: Multitier Client/Server Architecture

10.2 WEB SERVICES ARCHITECTURE

The implementation is required to provide multiple interfaces to the same data from different platforms and devices. It should be easily extensible to cater for new devices that may emerge in the future. This requirement necessitates the use of open, cross-platform standards like TCP/IP for universal networking and XML (eXtensible Mark-up Language) for universal structured data. Web services offer this openness.

Web services are self-contained, modular applications capable of supporting dynamic collaboration or just-in-time integration using loosely coupled components. They are network applications that use Simple Object Access Protocol (SOAP) and Web Service Description Language (WSDL) to exchange information in the form of XML documents. SOAP is an XML grammar and an open application protocol for Remote Procedure Call (RPC) and asynchronous messaging. WSDL is also an XML-based Interface Definition Language (IDL), which can be used to describe Web services including the kind of message format expected, the Internet protocol used, and the Internet address of the Web service. A detailed introduction to Web services and the benefits of the architecture can be found in Tidwell (2000), Gottschalk (2000) and Monson-Haefel (2002).

The Web services architecture contains three components and three operations as shown in Figure 8. The Service Provider encapsulates implementation details and publishes an Application Program Interface (API) for use by other services on the network. Service Requestors can discover available services through Service Brokers and then invoke (or request) those services from the Service Provider.

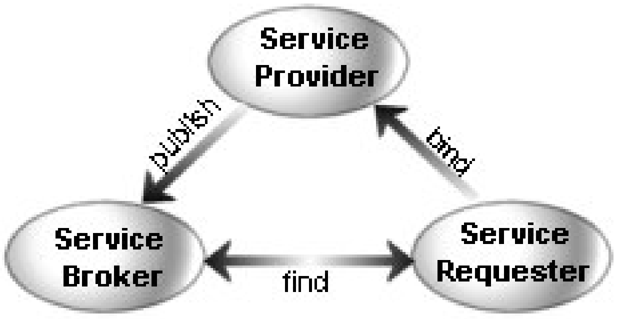


Figure 8: Web Services Architecture

11 IMPLEMENTATION ARCHITECTURE

The implementation is envisioned as a multitiered, service-based, message-oriented, client-server application. This is a hybrid solution, which combines the elements of Web services and multitier applications by implementing the middle tier of the latter as the Service Provider and Service Broker of the former. The backend services are still encapsulated in the middle tier and presentation components access the business and application logic as services. Figure 9 shows the proposed architecture.

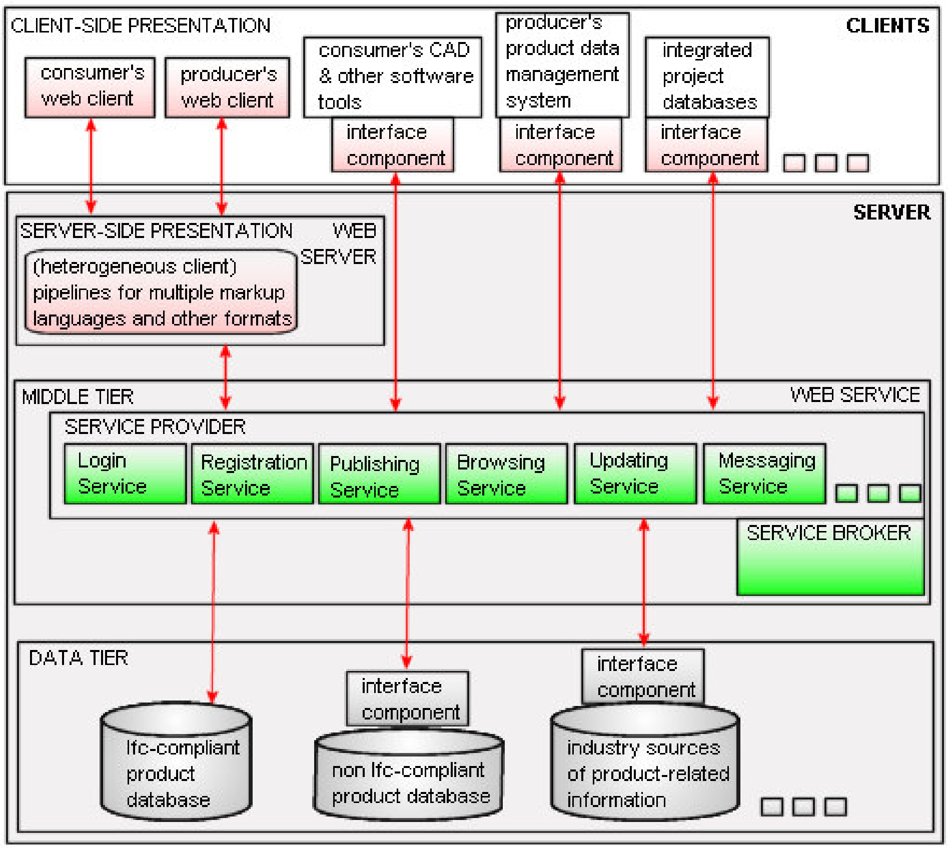


Figure 9: Multitiered, service-based, message-oriented client-server application architecture

In the past, many researchers have proposed and employed various aspects of the above architecture for IFC related implementations. Adachi (2002) adopted a Web Service approach to the development of IFC Model Server. Halfawy et al. (2002) recommended IFC-based messaging for systems interoperability and efficient data exchange. Faraj et al. (2000) developed an IFC-based distributed computer integrated environment using the three-tier architecture. With recent developments in software development platforms it is possible to harness the benefits of all these approaches in a single, yet cohesive, solution.

11.1 DATA TIER

The data tier consists of an implementation of IFC-complaint database and legacy or existing product data sources. Where possible, interface components that transform data from the non-compliant sources like manufacturers' existing product databases and other sources of product-related information, would be developed or implemented.
To ensure persistent access, data in this tier would grow incrementally. This implies, data, once published cannot be deleted but new versions can be released.

11.2 MIDDLEWARE

The middleware is a SOAP-based Web Service, which implements the application features including login, registration, publishing, and browsing components. Updating service and Messaging service would be asynchronous message-based services. The Updating Service would provide update information to consumers and allow consumers to register interest in specific parts of a particular product's information. It would implement a publish-subscribe communication model whereby notification of updates would be broadcasted to subscribed consumers. The Messaging Service would support one-to-one communication between producers and consumers. Additional services could be plugged in as required.

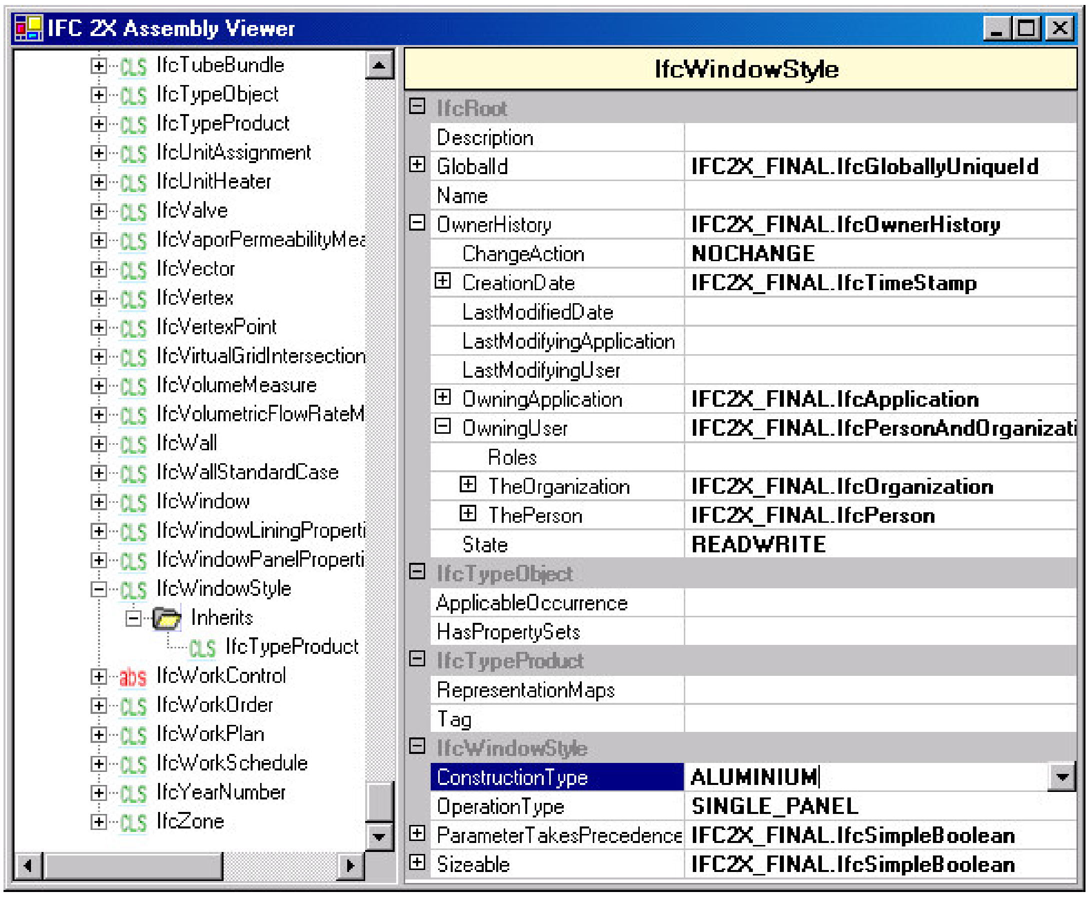
11.3 PRESENTATION

The Web Server would provide a transformation engine, which receives XML messages from the Web Service and translates it into required data formats and device or browser specific mark-up language such as HyperText Mark-up Language (HTML) and Wireless Mark-up Language (WML). This application would provide Web access to consumers and producers. Interface components would be developed to interface with other applications that would access the system over the Internet. These components would receive XML data directly from the Web service and transform or process it based on the specific deployment environment.

12 IMPLEMENTATION STATUS

The proposed solution is being implemented as part of an ongoing research project being undertaken by Loughborough University and Corus (UK) Ltd. Detailed specifications for the prototype applications are being produced as outstanding implementation issues are being clarified. Microsoft .NET platform was chosen for the delivery after a comparative study of applicable implementation platforms on cost, ease and efficiency of implementation. Available software tools ranging from integrated systems for storing and managing EXPRESS model data to basic tools like translators and libraries are being investigated.

An EXPRESS to C# code generator is being developed based on the Java EXPRESS parser developed by National Institute of Standards and Technology (NIST), US (Lardet, 2001). The code generator is being tested with IFC2x. It reads the IFC specification file, parses it, builds an abstract syntax tree and traverses the tree creating the C# source code file. The output code is then compiled into a .NET class assembly using Visual Studio .NET. The parser is being further developed to include methods for generating STEP P21 and IfcXML representation in each class.
Figure 10 is a screenshot of an IFC Assembly viewer designed for the generated .NET class Assembly. It uses the .NET object reflection to provide a navigable tree representation of all the class attributes, including inherited attributes. The representation for *IfcWindowStyle* is shown in the figure.
This tool has proved an invaluable tool in fostering better understanding of the IFC model amongst the development team. The development project is expected to complete in Spring, 2004 with the demonstration of the prototype application.



```
⊞ CLS IfcTubeBundle
⊞ CLS IfcTypeObject
⊞ CLS IfcTypeProduct
⊞ CLS IfcUnitAssignment
⊞ CLS IfcUnitHeater
⊞ CLS IfcValve
⊞ CLS IfcVaporPermeabilityMea
⊞ CLS IfcVector
⊞ CLS IfcVertex
⊞ CLS IfcVertexPoint
⊞ CLS IfcVirtualGridIntersection
⊞ CLS IfcVolumeMeasure
⊞ CLS IfcVolumetricFlowRateM
⊞ CLS IfcWall
⊞ CLS IfcWallStandardCase
⊞ CLS IfcWindow
⊞ CLS IfcWindowLiningProperti
⊞ CLS IfcWindowPanelProperti
⊟ CLS IfcWindowStyle
    ⊟ Inherits
        CLS IfcTypeProduct
⊞ abs IfcWorkControl
⊞ CLS IfcWorkOrder
⊞ CLS IfcWorkPlan
⊞ CLS IfcWorkSchedule
⊞ CLS IfcYearNumber
⊞ CLS IfcZone
```

IfcWindowStyle	
⊟ IfcRoot	
Description	
⊞ GlobalId	IFC2X_FINAL.IfcGloballyUniqueId
Name	
⊟ OwnerHistory	IFC2X_FINAL.IfcOwnerHistory
ChangeAction	NOCHANGE
⊞ CreationDate	IFC2X_FINAL.IfcTimeStamp
LastModifiedDate	
LastModifyingApplication	
LastModifyingUser	
⊞ OwningApplication	IFC2X_FINAL.IfcApplication
⊟ OwningUser	IFC2X_FINAL.IfcPersonAndOrganizati
Roles	
⊞ TheOrganization	IFC2X_FINAL.IfcOrganization
⊞ ThePerson	IFC2X_FINAL.IfcPerson
State	READWRITE
⊟ IfcTypeObject	
ApplicableOccurrence	
HasPropertySets	
⊟ IfcTypeProduct	
RepresentationMaps	
Tag	
⊟ IfcWindowStyle	
ConstructionType	ALUMINIUM
OperationType	SINGLE_PANEL
⊞ ParameterTakesPrecedence	IFC2X_FINAL.IfcSimpleBoolean
⊞ Sizeable	IFC2X_FINAL.IfcSimpleBoolean

Figure 10: Ifc Assembly Viewer

13 DISCUSSION AND CONCLUSION

The IFC model is the foundation for interoperability in the AEC/FM industry. A well-engineered application architecture and implementation based on the model can meet the industry's present and future requirements for reliable, flexible and available product information, which is complete and reusable in software applications employed in the industry. This paper identifies key requirements for online product libraries and presents an architecture, which has been derived for the implementation.

The derived architecture is aimed at supporting present and emerging industry practices in the production and consumption of product information. The multi-tiered, service-based, client/server architecture presented can support role-based, multimedia presentation and subscription-based cooperation among implementations. This would enable the systems to support the project team and the supply chain. In addition, it makes provision for gradual buy-in and staged system changeovers as each system is capable of existing independently or within a group.

Loose coupling of components and asynchronous messaging are employed to further decouple the system and increase flexibility of integration. The development of interface components is proposed as a mean for integrating existing systems and tools. Versioning and persistent access problems are addressed through a write-once publishing approach.

The major hindrances to utilising the IFC model for the implementation of a prototype system are the maturity of the model and the lack, or limited availability, of software tools that can facilitate rapid application prototyping. The IFC model is fast evolving and has good prospects for industry acceptance hence the decision to adopt the standard. The EXPRESS modelling language, used to define the model, poses some challenges to developers. It is designed to be human-readable but as the number of, and inter-relationships between, its elements grow its human-readability diminishes. The graphical representation, in EXPRESS-G, does not improve the situation much. The Ifc Assembly Viewer (presented in Section 12) addresses this problem.

There are still a number of questions and issues that must be addressed in implementing the architecture proposed here. Some of these include:

- Is the industry ready for standardisation considering the cost of translation?
- How should the solution be packaged and delivered to the industry?
- Who should provide the translators?
- What happens if the IFC changes in its core?
- How to manage versions, persist links and eliminate data relating to dead products.
- How should multiple classification systems be mapped to objects?

These and many other issues have been identified and are being researched as the implementation progresses.

14 REFERENCES

Adachi, Y. (2001). IFC 2x property set development guide. Technical Report MSG-01- Draft 2, International Alliance for Interoperability.

Adachi, Y. (2002). Overview of IFC model server framework. In European Conference of Product and Process Modelling (ECPPM) eWork and eBusiness in AEC (Turk, Z. and Scherer, R., editors).

ANSI/IEEE (2000). 1471-2000 IEEE Recommended Practice for Architectural Description for Software-Intensive Systems. ISBN 0-7381-2518-0.

Clark, P., Thompson, J., Holmback, H., and Duncan, L. (2000). Exploiting a thesaurus-based semantic net for knowledge-based search. In Proceedings of 17th National Conference on Artificial Intelligence and 12th Conference on Innovative Applications of Artificial Intelligence (AAAI/IAAI'2000) [ISBN 0-262-51112-6], Austin Texas, USA, pages 988-995, California, USA. AAAI Press (Distributed by MIT Press).

CPIC (1997).UNICLASS: Unified Classification for the Construction Industry. RIBA Publications. Construction Project Information Committee.

CSI (1995). MasterFormat 1995.Construction Specification Institute, 99 Canal Center Plaza, Suite 300, Alexandria, VA 22314.

CSI (1998).UniFormat 1998 Edition. Construction Specification Institute, 99 Canal Center Plaza, Suite 300, Alexandria, VA 22314.

Daley, W. M., Bachula, G. R., and Kammer, R. G. (1999). STEP: The Grand Experience. US Department of Commerce, Technology Administration, National Institute of Standards and Technology. Edited by Sharon J. Kemmerer.

Denzinger, J., Sinz, C., Avenhaus, J., and Uchlin, W. K. (2002). Teamwork-PaReDuX: Knowledge-based search with multiple parallel agents. In Proceedings of International Conference on Massively Parallel Computing Systems, Ischia, Italy.

Eastman, C. M. (1999). Building Product Models : Computer Environments Supporting Design and Construction.CRC Press LLC, 2000 N.W. Corporate Blvd., Boca Raton, Florida 33431, USA.ISBN 0-8493-0295-5.

Ekholm, A. (1999). Co-ordination of classifications for product modelling and established building classifications. In CIB W78 Workshop on Durability of Building Materials and Components 8, Vol. 4 Information Technology in Construction (Lacasse, M. A. and Vanier, D. J., editors), Ottawa, Canada. NRC Research Press.

Ekholm, A. (2002). Principles for classification of properties of construction objects. In Distributing Knowledge in Building (Agger, K., Christiansson, P., and Howard, R., editors), Aarhus School of Architecture, Denmark. CIB w78 Conference. ISBN 87-90078-36-5.

Ekholm, A., Haggstrom, L., and Karlsson, H. (2000). BSAB 96: The Swedish Construction Industry Classification System. The Swedish Building Centre Systematics, S-113 87 Stockholm, Sweden.

EPIC (1999). EPIC Master Version 1999.EPIC General Secretariat, WTCB/CSTC, Lozenberg I Nr 7, B-1932 Zaventem, Belgium, version 2.0 edition.

Faraj, I., Alshawi, M., Aouad, G., Child, T., and Underwood, J. (2000). An industry foundation classes web-based collaborative construction computer environment: WISPER. Journal of Automation in Construction, 10:79-99.

Faux, Kesteloot, Stewing, and Pusch (1998a). Product catalogues in global engineering networking. In Proceedings of ACM. Also presented at GEN '97, Antwerp, Belgium, 1997.

Faux, I., Radeke, E., Stewing, F.-J., Broek, G. V. D., Kesteloot, P., and Sabin, A. (1998b). Intelligent access, publishing and collaboration in global engineering networking. Computer Networks, 30(13):1249-1262.

Gottschalk, K. (2000). Web services architecture overview: The next stage of evolution for e-business. IBM DeveloperWorks. Available online at ftp://www6.software.ibm.com/software/developer/library/w-ovr.pdf.

Halfawy, M. R., Pouria, A., and Froese, T. (2002). Developing message-based interoperability protocols for distributed AEC/FM systems. In Distributing Knowledge in Building (Agger, K., Christiansson, P., and Howard, R., editors), Aarhus School of Architecture, Denmark. CIB w78 Conference. ISBN 87-90078-36-5.

Hannus, M. and Pirhonen, A. (2001). RINET - building product database. http://cic.vtt.fi/projects/rinet/index.html.

Howard, R. (2001). Classification of building information - European and IT systems. In Construction Information Technology, International Conference : IT in Construction in Africa, pages 9-1 to 9-14. CSIR, Building and Construction Technology.

IAI (2000). IFC Technical Guide. International Alliance for Interoperability. Copy available at http://www.iai.org.uk/documentation/IFC_2x_Technical_Guide.pdf.

IAI-NA (2002). Interoperability in building industry moves forward with ISO ballot. News release published online at http://www.iai-na.org/news/121802.php. International Alliance For Interoperability, North America. Last visited: 28 March, 2003.

ISO (1997). Organisation of information about construction works - part 2: Framework for classification of information. Technical Report ISO/TC 59/SC 13 N 86, International Organisation for Standardisation, Geneva, Switzerland.

ISO (2000). Building construction - organization of information about construction works - part 3: Framework for object-oriented information exchange. Technical Report ISO/PAS 12006-3, International Standards Organisation, Geneva, Switzerland.

Jain, S. and Augenbroe, G. (2002). Performance-based electronic product catalogues for the building industry. In Proceedings of e-2002, eBusiness and eWork Conference in Prague.

Kosovac, B., Vanier, D. J., and Froese, T. M. (2000). Use of keyphrase extraction software for creation of an AEC/FM thesaurus. Electronic Journal of Information Technology in Construction, Edited by Bo-Christer Björk, 5:25-36.Available online at http://itcon.org/2000/2.

Lardet, S. (2001). Java Express Parser User Documentation. National Institute of Standards and Technology, US.

Ling, S. R., Will, P., Kim, J., and Luo, P. (1997). Active Catalog: Searching and using catalog information in internet-based design. In Proceedings of DETC '97 - 1997 ASME Design Engineering Technical Conferences (Cheng, H. H., editor), Sacramento, California. Copy available at http://www.isi.edu/active-catalog/asme-97.ps.

Monson-Haefel, R. (2002). EJB 2.1 web services. theserverside.com. Available at www.theserverside.com.

Myers, B. A. (1998). The case for an open data model. Technical Report CMU-CS-98-153, Human Computer Interaction Institute, School of Computer Science, Carnegie Mellon University, Pittsburgh, USA. Copy available at http://reports-archive.adm.cs.cmu.edu/anon/1998/CMU-CS-98-153.pdf.

Newnham, L., Amor, R., and Parand, F. (1998). Gaining quality manufactured product information through ARROW.PDT Days, BRE, UK, pages 39-46.

Newnham, L., Parand, F., Amor, R., and Nisbet, N. (1997). The ARROW framework for a building object warehouse. In CIB W78 Workshop on Information Technology Support for Construction Process Re-Engineering, IT-CPR 97 (Drugemuller, R., editor), pages 319-328. International Council for Building Research Studies and Documentation.

Obonyo, E., Anumba, C. J., Thorpe, A., and Parkes, B. (2001). Agent-based support for electronic procurement in construction. In International Conference on Intelligent Agents, Web Technologies, and Internet Commerce (IAWTIC) (Mohammadian, M., editor), Las Vegas, USA, pages 268-279, Australia [ISBN: 0858898470]. University of Canberra.

OCCS (2001). Overall Construction Classification System. The OCCS Development Committee. www.occsnet.org.

Pierra, G., Sardet, E., Potier, J. C., Battier, G., Derouet, J. C., Willmann, N., and Mahir, A. (1998). Exchange of component data : The PLIB model, standard and tools. In CALS Europe '98, pages 160-176, Paris, France.

Raduchel, W. (1998). Implementing a multitier, services-based architecture on the Java platform at Sun - a case study. Technical report, Sun Microsystems. Copy available at http://www.sun.com/980602/wp/index.html.

Ray-Jones, A. and Clegg, D. (1991). CI/SfB Construction Indexing Manual. RIBA Publications, London, 2001]

RIBA (2001). The 2001 survey on information services in the UK construction industry. RIBA Companies Ltd., London. Copy available at http://www.ribac.co.uk/images/ survey2001.pdf.

Stouffs, R. and Krishnamurti, R. (2001). Standardization: A critical review. In International Conference on IT in Construction in Africa, pages 42-1 to 42-10. CSIR, Building and Construction Technology. Conference website and papers at http://buildnet.csir.co.za/constructitafrica/.

Tidwell, D. (2000). Web services - the web's next revolution. IBM DeveloperWorks. Available online at http://www-106.ibm.com/developerworks/.

Turk, Z., Wasserfuhr, R., Katranuschkov, P., Amor, R., Hannus, M., and Scherer, R. J. (1997). Conceptual modelling of a concurrent engineering environment. In Proceedings of First International Conference on Concurrent Engineering in Construction, July 3-4, 1997, London, UK

Woestenenk, K. (1999). LexiCon - a common vocabulary for the construction industry. In Proceedings of 2nd International Conference on Concurrent Engineering and Construction (CEC99), Espoo, Finland.

Woestenenk, K. (2000). The LexiCon: An update. In Proceedings of Product and Process Modelling in Building and Construction (Gonçalves, R., Steiger-Garção, A., and Scherer, R., editors). ECPPM 2000, Balkema, Rotterdam.

Wright, T. (1998). Classifying building information: An historical perspective. Technical report, Construction Information Systems Australia Pty Ltd.

APPENDIX C PAPER 3

Owolabi, A., Anumba, C., El-Hamalawi, A. & Harper, C. (2004), "Development of an Industry Foundation Classes (IFC) Assembly Viewer", ASCE Journal of Computing in Civil Engineering. (***Accepted for publication, in press***)

Development of an Industry Foundation Classes (IFC) Assembly Viewer

ABSTRACT: The construction industry has invested considerable effort into integration of project information in the last decade. One such effort is the definition of Industry Foundation Classes (IFCs) to facilitate data sharing across applications through a shared project model. In order to achieve the integration objectives, the industry software vendors need to commit to the implementation of IFC in their products.
IFC is defined in EXPRESS, which is a platform-independent, object-flavoured, data modelling language. The EXPRESS-based models must be translated into some programming language model for specific implementation. To achieve this, developers need to evaluate and select a suitable model and programming language for their implementation. Developers therefore need to understand both EXPRESS and a host of programming languages. This initial knowledge requirement may hinder the take-off or adoption of IFC-based implementation.
This paper describes a software solution that reduces this initial knowledge requirement considerably by providing a .NET class library translation and an implementation view of the IFC model, based on the EXPRESS definitions. Complemented by the online documentation provided with the IFC definitions, the software provides a hierarchical view of the IFC-based programming objects with drill-down facility for developers to capture and appreciate the information requirement for specific objects.

KEY WORDS**:** *IFC, EXPRESS, language translation, .NET class library*

1 INTRODUCTION

In the construction industry, Information and Communication Technologies (ICT) have long been identified as key enablers to the achievement of greater productivity through integrated information management and collaborative working. An integrated approach requires a standardised format for the representation and possibly presentation of information. In other words, the syntax and semantics for coding information needs to be predefined to facilitate sharing or exchange of the information amongst all the stakeholders in the construction process. The International Alliance for Interoperability (IAI) has specified the Industry Foundation Classes (IFCs) to provide standard for digital representation of *things* and facilitate unambiguous transfer or sharing of information between computer systems within the Architectural, Engineering, Construction and Facility Management (AEC/FM) industry.
Large software vendors like Autodesk, Bentley, Graphisoft, and Nemetschek have shown commitment to the support of IFCs on their products. Dozens of IFC-compliant software prototypes have been built as a result of various research projects. Unlike the large software vendors, researchers and small/medium software vendors seeking to implement IFCs, or any EXPRESS-based information model, would be faced with the problem of resourcing the knowledge required first to make an educated decision on the suitability of the information model and then to actually carry out the implementation. Developers need to understand both the EXPRESS language and their target information model which usually consists of hundreds of interrelated entities. This initial knowledge requirement may hinder the development of IFC-based software tools.

The IAI provides a number of documents to assist developers. These include general documentation of how EXPRESS is used in IFC definitions and model-specific documents such as implementation guides, technical guides, and model documentations and schemas. These guides describe the concepts, architectures, contents and structures of defined information models thus serving as the first port of call for prospective implementers.
In addition to documentation, tool support is critical to the implementation of any information model and a host of software tools have been developed to work with EXPRESS definitions. The availability of tools would influence the software architecture or high-level decisions on the overall structure of the system such as choice of database and programming language. Presently, there are tools for translating EXPRESS definitions into various other forms including:

1. Data Definition Language (DDL) – definitions of data structures for relational databases such as MS SQL, ProgresSql and MySql;
2. Document Type Definition (DTD) – definitions of the legal building blocks of an XML document detailing the document structure with a list of allowable elements;
3. Xml Schema Definition (XSD) – alternative definitions to the DTD;
4. Interface Definition Language (IDL) - language-independent definitions of the data structures passed between implementations; and
5. Programming Languages – including C, C++, and Java.

Programming language mappings and translations are essential to developers and availability of a translation in their chosen programming language would speed up the development process. This paper presents the development of a software tool which provides a programming language mapping and translation that has the potential to eliminate the need for any other programming language mapping. The software tool, named IFC Assembly Viewer, harnesses the benefits of the language-independent programming model provided by .NET. It produces a class library that can be used with any .NET compliant programming language including C++, C#, Visual Basic, Visual J#, COBOL, Fortran, Standard ML, Pascal, Perl, Python, etc. In addition, it presents a hierarchical view of the library or IFC-based programming objects with drill-down facility to allow developers query the object definitions to any desired level of detail. This eliminates or reduces the need for developers to learn the information model via the EXPRESS modelling language and tools.
Section 2 of this paper contains a brief introduction to EXPRESS modelling language followed by a case for the IFC Assembly Viewer in Section 3 and a discussion of EXPRESS mapping in Section 4. Section 5 discusses the architecture of the IFC Assembly Viewer including the processes implemented. The paper concludes in Section 6 by highlighting the benefits and limitations of the implementation and identifying areas for future developments.

2 A GENTLE INTRODUCTION TO EXPRESS

EXPRESS is a lexical, object-flavoured modelling language developed to enable a formal specification of the ISO Standard 10303 Product Data Representation and Exchange, otherwise known as STEP. In addition to some other STEP parts, the IFC uses EXPRESS with some restricted constructs. A description of the EXPRESS language is specified in Part 11 of STEP (ISO, 1994b) while a detailed introduction can be found in Schenck & Wilson (1997) and a brief introduction to its use within IFC development can be found in IAI (2001a).

The principal elements of the EXPRESS language are Type, Entity, Schema, Algorithm and Rule. EXPRESS is based on an entity-attribute paradigm. It specifies domain information in terms of entities, corresponding to classes in object-oriented programming. Attributes are associated with entities and can be constrained with a combination of declarative and procedural specifications called Rules. Algorithms can be specified as functions and procedures for general use in entity specifications. Supertype/subtype relationships can be defined between entities and entities can be grouped into schemas, which in turn can be combined to form models.

There are five classes of EXPRESS data types used in IFC development. These include:

Simple Data Types – These represent the atomic unit of data and include: REAL, INTEGER, NUMBER, LOGICAL, BOOLEAN, BINARY, and STRING. The BINARY data type is not used in IFC definitions.

Aggregate Types – These represent a collection of things and include: ARRAY, BAG, LIST, and SET.

Named Types – These represent user-defined data items and form the bulk of most schema definitions. There are two types: declared types (TYPE) and entity types (ENTITY). Declared types are based on some underlying type and possibly refined with some constraint.

Constructed Types – These are defined like the declared types but with ENUMERATION or SELECT constructs. ENUMERATION allows a selection from a list while SELECT is essentially a named grouping of other data types.

Generalized Types – These are non-instantiable types which serve as a generalisation of some other types. These include AGGREGATE for all aggregate data types (ARRAY, LIST, BAG and SET) and GENERIC for all possible data types.

Attributes represent properties of the entities in terms of their data requirements using the data types defined above. An attribute can be described as explicit, derived or inverse. Explicit attributes have static, independent values and they may be classified as OPTIONAL. Values in derived attributed may change in response to changes to other attribute values and inverse attributes point to the objects that use the current object as value of one of their explicit attributes.

Figure 1 shows an annotated diagram of some EXPRESS constructs as they relate to *IfcOccupant* entity in the IFC version 2x schema.

A graphical subset of EXPRESS, called EXPRESS-G, is available for visual representation of EXPRESS specifications. EXPRESS-G diagrams contain the data structures, inheritance (Subtype/Supertype) relationships, attributes and relationships between data structures in the EXPRESS information model. They, however, do not depict the constraints and constraint mechanisms (or rules) provided by EXPRESS. Coincidentally, the implementation discussed in this paper does not yet incorporate rules.

2.1 PROBLEMS WITH EXPRESS

EXPRESS modelling language is an expressive and unambiguous modelling language. It is however not the easiest of modelling languages and tool availability is comparatively very limited. In addition, most developers are not familiar with the modelling language and as the number of entities and their interrelationship grows, the difficulty in understanding an EXPRESS schema is amplified. An attempt to trace every relationship for just *IfcOccupant* in Figure 1 to gain complete understanding of its information requirement would highlight the difficulty of understanding a schema with more than 400 entities.

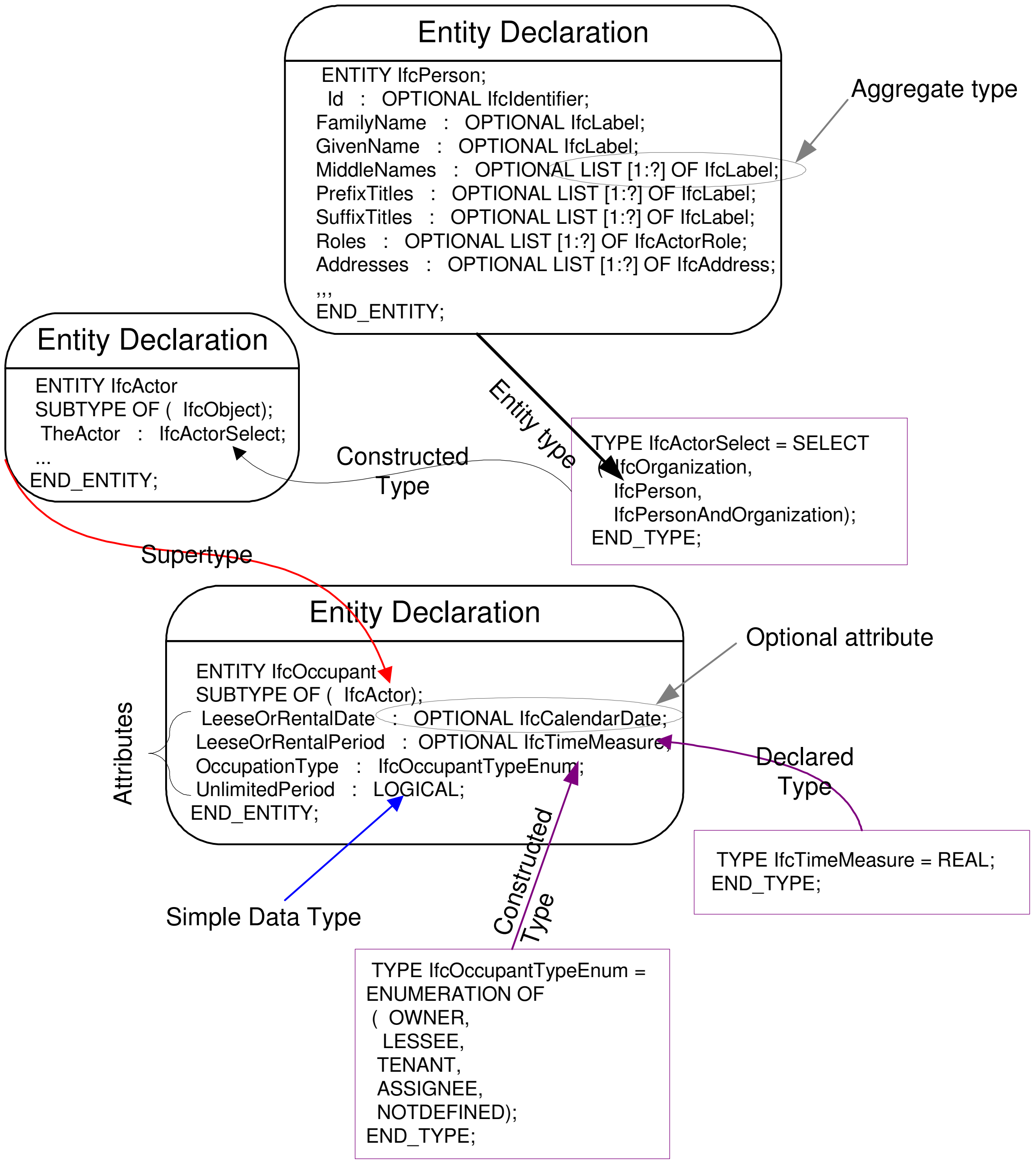

Entity Declaration
ENTITY IfcPerson;
Id : OPTIONAL IfcIdentifier;
FamilyName : OPTIONAL IfcLabel;
GivenName : OPTIONAL IfcLabel;
MiddleNames : OPTIONAL LIST [1:?] OF IfcLabel;
PrefixTitles : OPTIONAL LIST [1:?] OF IfcLabel;
SuffixTitles : OPTIONAL LIST [1:?] OF IfcLabel;
Roles : OPTIONAL LIST [1:?] OF IfcActorRole;
Addresses : OPTIONAL LIST [1:?] OF IfcAddress;
,,,
END_ENTITY;
Aggregate type
Entity Declaration
ENTITY IfcActor
SUBTYPE OF (IfcObject);
TheActor : IfcActorSelect;
...
END_ENTITY;
Entity type
Constructed Type
TYPE IfcActorSelect = SELECT
(IfcOrganization,
IfcPerson,
IfcPersonAndOrganization);
END_TYPE;
Supertype
Entity Declaration
ENTITY IfcOccupant
SUBTYPE OF (IfcActor);
LeeseOrRentalDate : OPTIONAL IfcCalendarDate;
LeeseOrRentalPeriod : OPTIONAL IfcTimeMeasure;
OccupationType : IfcOccupantTypeEnum;
UnlimitedPeriod : LOGICAL;
END_ENTITY;
Optional attribute
Attributes
Declared Type
TYPE IfcTimeMeasure = REAL;
END_TYPE;
Simple Data Type
Constructed Type
TYPE IfcOccupantTypeEnum =
ENUMERATION OF
(OWNER,
LESSEE,
TENANT,
ASSIGNEE,
NOTDEFINED);
END_TYPE;

Figure 1: EXPRESS Modelling Language Elements

While EXPRESS-G improves understanding of a model, its tool support is generally feature-limited compared to a more popular modelling language like the Unified

Modelling Language (UML). There are on-going efforts aimed at harmonising the two graphical modelling languages. Arnold & Podehl (1998) examine the feasibilities and issues associated with mapping EXPRESS models to UML.

3 PROPOSING AN IFC ASSEMBLY VIEWER

As highlighted in the Introduction (Section 1), several tools have been developed to allow EXPRESS specifications such as the IFCs, to be viewed, edited, validated, and transformed into functional programming language codes, database schema definitions and online documentation. Tools for mapping EXPRESS definitions to Java packages and C++ class libraries have been employed in a number of commercial and research implementations. Other solutions that work directly with EXPRESS rather than some translation have also been widely deployed particularly in the manufacturing and aerospace industries. The design, capabilities and complexities of these tools can, however, present barriers to developers. In this section, a brief review of available translation tools is presented and a case is made for yet another translation tool - from EXPRESS to .NET class library.

3.1 EXISTING EXPRESS TOOLS

The most popular EXPRESS translators provide C++ and Java classes. Examples include:

1. JSDAI EXPRESS Compiler, from LKSoft (*www.lksoft.com*) – This parses EXPRESS schemas and generates corresponding Java packages with interface and class files for the schemas;
2. Open Source IFC Generated C++ Classes and XP-COMPILER – This is an EXPRESS language parser written in Yacc and C, from OSS QualiSTEP, CSTB,
3. France (http://cic.cstb.fr/csc);
4. STEP Tools EXPRESS compiler (*www.steptools.com*) – This takes information models defined in the EXPRESS language as input, and compiles them into C++ code for STEP Tools' ROSE, an API for CAD and data exchange applications; and
5. NIST EXPRESS Toolkit (Libes, 1993) – This provides C and C++ translators in addition to a host of other tools for working with EXPRESS.

JSDAI is comparatively well developed and commercially available but it limits the development platform to Java. The C++ classes provided by QualiSTEP are only for IFC 2.0 and as such do not take advantage of recent extensions to the model. The C++ code produced by Step Tools EXPRESS Compiler, on the other hand, is targeted at ROSE developers. These inadequacies have prompted research into further translations. Examples of such work include:

1. Express2C – a translator of EXPRESS model to C language model (Chunyan et al., 2003);
2. pyEXPRESS – a Python Open Source EXPRESS Compiler/Code Generator (Koning, 2003); and

3. Express2Perl – a translator from EXPRESS to Perl which can currently read EXPRESS files and list schemas, entities, and some attributes (Wichmann, 2002).

These translators are still in their infancy and not yet suitable for production-strength deployment. There are, however, other tools working directly with the EXPRESS models encapsulating the translation process. Examples include:

1. EuroSTEP EXPRESS Parser (*www.eurostep.com/prodserv/exff/exff.html*)– This parses EXPRESS models and generates some form of XML;
2. EXPRESS for Free (exff, *exff.sourceforge.net*) toolset – This parses EXPRESS schema into a UML model to lower barriers to implementation of EXPRESS-based standards;
3. EXPRESS to XML Schema Converter (EXC, *cic.vtt.fi/projects/ifcsvr*) – This converts EXPRESS schema into an intermediate XML meta model format for further transformation with XSLT and XML parsers.
4. EPM Technology EXPRESS Data Manager (*www.epmtech.jotne.com/products*) – This offers a complete solution for managing EXPRESS data and models;
5. ST-Developer (*http://www.steptools.com/products/stdev*) - Like EPM Data Manager, this is a suite of applications for working with EXPRESS data and models.

Apart from EPM Data Manager, which presents a complete solution including APIs for developers to build applications on the data manager, the above tools provide discrete solutions that strongly suggest specific implementation routes.

3.2 .NET CLASS LIBRARY

The motivation for developing a translator for a .NET language can be found in the revolutionary approach to programming presented in the Microsoft .NET Framework - Language and O/S Platform independence. This programming model allows development to be undertaken on a cross-platform and multi-language environment thus giving programmers the freedom to contribute to a development using their preferred platform and language rather than having to use the system designer's platform and specified language. In other words, there is potentially no limit as to where a .NET class library can be used.
Considerable time is usually spent working out and standardising EXPRESS programming language bindings or mappings. Sometimes a language can even fall out of use before the completion of the process thus leading to a lot of wasted effort. By adopting the .NET approach, efforts can be concentrated only on specifying bindings for just one .NET language as variants between the languages are eliminated through the .NET compilation process.
It can be expected that adopting the .NET approach would lower implementation thresholds, ensure portability across platforms and languages, facilitate future availability of data exchange solutions, and reduce dependency on unstable third-party tools and third-party licenses.

4 MAPPING EXPRESS

How an EXPRESS information model is used in a software environment can be categorised as either Early Binding or Late Binding. Early-bound implementation works with programming language data structures representing each construct in the model. For example, a C++ early-bound implementation would contain C++ classes representing EXPRESS entities. This allows for compile-time error detection by exploiting the type-checking facility provided by the language compiler. To produce an early-bound implementation, a translator needs to parse an EXPRESS model and produce source code for the defined data structures.

Late-bound implementations use a data dictionary to provide access to data values stored with a generic data structure. The EXPRESS model is compiled to produce the data dictionary though individual constructs may also be translated in a mixed binding environment. Late binding offers simplicity by providing a single interface independent of the EXPRESS model. Loffredo (1999) provides more details on binding and implementation issues.

The IFC Assembly Viewer is an early-binding implementation although it demonstrates the feasibility of extending the implementation to late-binding.

5 IFC ASSEMBLY VIEWER

The IFC Assembly Viewer validates EXPRESS schema definitions, translates the definitions into the Visual Basic.NET programming language, compiles the generated source codes into a class library and provides a two-way synchronisable implementation view to complement the schema documentation. This software bridges the gap between information modellers and programmers working with EXPRESS-based models by providing tools to support understanding of the models and their deployment in programming environments.

The IFC Assembly Viewer implementation involves three processes as follows:

1. Translation process – EXPRESS definitions are parsed and converted into an in-memory representation called an Abstract Syntax Tree (AST);
2. Mapping process – Transforming EXPRESS constructs in the AST into a programming language of choice; and
3. Viewing process – Generating a view of the compiled programming language constructs.

Figure 2 shows the flow diagram of the implementation process.

5.1 TRANSLATION PROCESS

The translation process employs a Java EXPRESS Parser generated by ANTLR Java parser Generator using a modified version of the EXPRESS grammar definition provided by the National Institute of Standards and Technology (NIST) (Lardet & Lubell, 2001) as input. The output Java application takes an EXPRESS schema file (.exp) as input, parses it to verify that the constructs are syntactically valid, and generates an in-memory tree data structure called Abstract Syntax Tree (AST) to represent the EXPRESS schema.

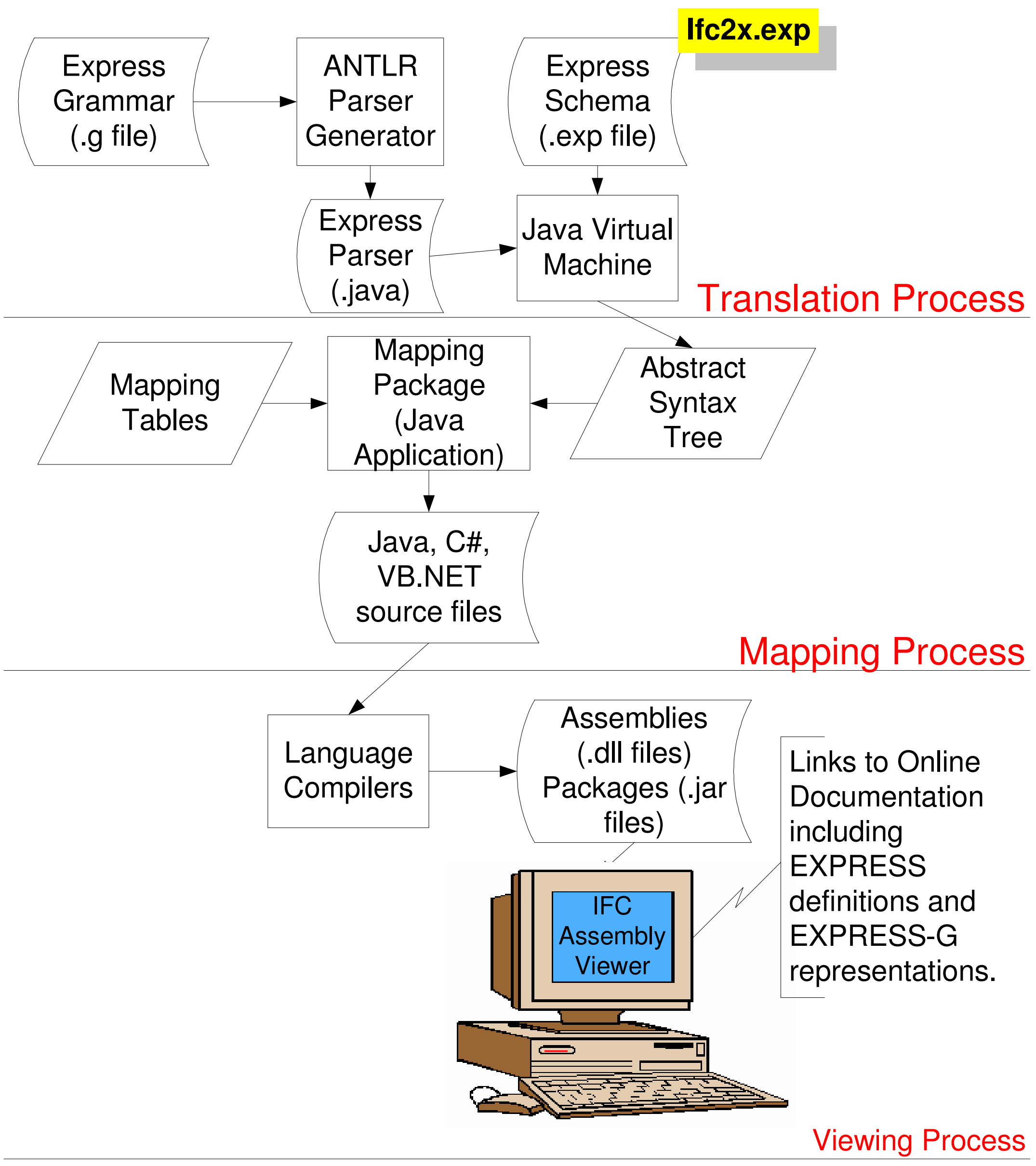

Ifc2x.exp
Express Grammar (.g file)
ANTLR Parser Generator
Express Schema (.exp file)
Express Parser (.java)
Java Virtual Machine
Translation Process
Mapping Tables
Mapping Package (Java Application)
Abstract Syntax Tree
Java, C#, VB.NET source files
Mapping Process
Language Compilers
Assemblies (.dll files) Packages (.jar files)
Links to Online Documentation including EXPRESS definitions and EXPRESS-G representations.
IFC Assembly Viewer
Viewing Process

Figure 2: IFC Assembly Viewer Implementation Flow Diagram

```
subtype_declaration returns [String s]
   { String er=null;}
     :   #(SUBTYPE_DECLARATION er=entity_ref
            { currentClass.setBase(er);}
            ( er=entity_ref
            { currentClass.setBase(er);}
          )*
          );
```

Figure 3: Sample EXPRESS Language Rules in EBNF and added Actions

ANTLR (ANother Tool for Language Recognition) is a tool that accepts grammatical language descriptions and generates programs that recognise sentences (or constructs) in those languages. The language grammar, including rules specified with Extended Backus-Naur Form (EBNF) (ISO/IEC, 1996) notation, may be augmented with *operators* and *actions* to customise the building of the AST and/or generate outputs. Parr (2003) provides a detailed introduction to ANTLR. Custom actions were added to the grammar definitions provided by NIST to capture information for the mapping process. Figure 3 highlights the actions (in red) added to the rule for processing subtype declarations.

5.2 MAPPING PROCESS

The AST generated in the translation process is read by the mapping procedure included in the translation (Java) package to generate Visual Basic.NET source files based on some mapping rules. Though Visual Basic.NET is selected here, the application could easily be modified to output another language by providing the appropriate mapping rules. Table 1 describes the translation rules used in this implementation.
To support the viewing process at run-time, some metadata attributes are added to the classes, methods and properties in the generated code. These are used as follows:

- To associate attributes with exactly where they are defined in the inheritance hierarchy for categorisation purposes;
- To specify how the data types are converted during processing, for example, *IfcDaysInMonth* is treated essentially as an Integer; and
- To specify what editors (or forms) should be used in displaying the data type, for example, an interface is displayed in a form that allows the selection and editing of the classes that implement it.

Figure 4 shows the source listing for *IfcPersonAndOrganisation*. Derived and Inverse attributes are not mapped in the current implementation. Derived attributes can be mapped to read-only properties but this would require mapping and translations of both built-in and declared functions. Inverse attributes can also be mapped to properties but with the addition of metadata and manual coding of service methods to manage the references to objects as the objects are created and destroyed. This implementation of this has performance implication for runtime memory management and data persistence.

Table 1: Mapping Table for EXPRESS to VB.NET

EXPRESS	VB.NET	Description
Simple Types are mapped to classes as follows:		
REAL	IfcSimpleReal	encapsulates the *DOUBLE* data type
INTEGER	IfcSimpleInteger	encapsulates the *Integer* (or *Int32*) data type
NUMBER	IfcSimpleNumber	encapsulates the *Decimal* data type
LOGICAL	IfcSimpleLogical	designed to model the EXPRESS LOGICAL type
BOOLEAN	IfcSimpleBoolean	encapsulates the *Boolean* data type
BINARY	IfcSimpleBinary	designed to model the EXPRESS BINARY data type
STRING	IfcSimpleString	encapsulates the *String* data type
Aggregate Types are mapped to classes which inherit *CollectionBase* with behaviours to match specific types such as ARRAY, BAG, LIST, SET. For example, *BagOfIfcLabel*. Where applicable, constructors are used to specify ranges.		
Declared Types are translated as follows:		
ENUMERATION	Enum type	e.g. *IfcOccupantTypeEnum*
SELECT	Interface type	Empty interface definitions which are then implemented by every type in the SELECT list. E.g. *IfcActorSelect*
Declared types based on other types	Class	This class inherits the underlying type specified in the EXPRESS e.g. *IfcDayInMonthNumber* inherits *IfcSimpleInteger*
Entity	Class	An empty constructor is provided and where there is at least one non-optional attribute, a constructor with all the attributes is provided with optional attributes represented by *optional* parameters.

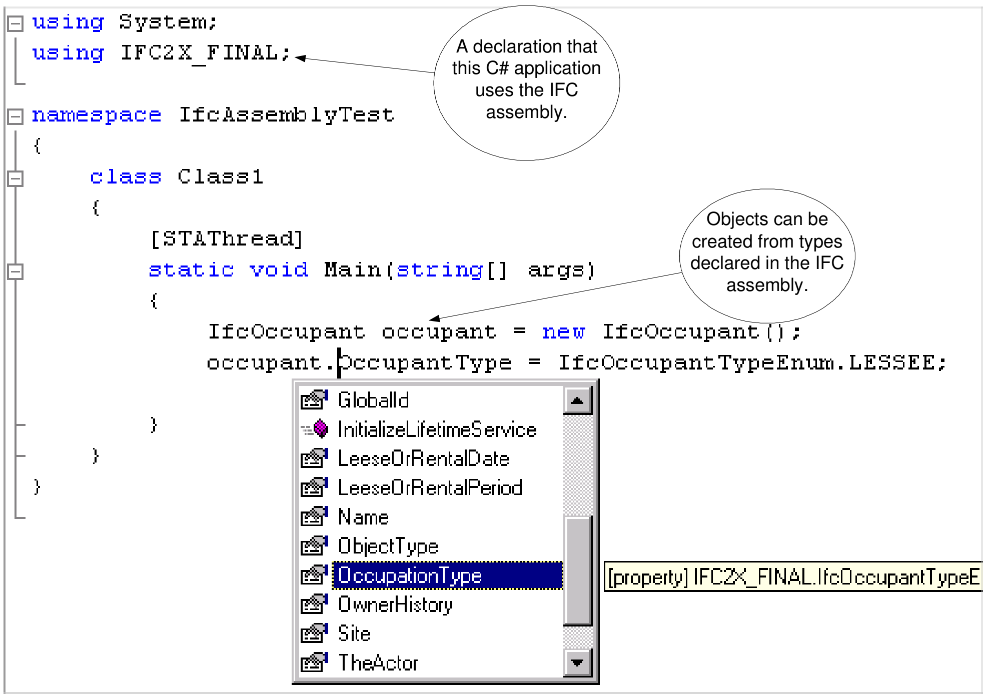
```
Public Class IfcPersonAndOrganization
    Implements IfcObjectReferenceSelect, IfcActorSelect
    Private _thePerson As IfcPerson
    Private _theOrganization As IfcOrganization
    Private _roles As IfcAggregate.ListOfIfcActorRole
    Public Sub New()
        MyBase.New()
        _thePerson = New IfcPerson()
        _theOrganization = New IfcOrganization()
        _roles = Nothing   'optional attribute
    End Sub
    Public Sub New(ByVal pThePerson As IfcPerson, _
        ByVal pTheOrganization As IfcOrganization, _
        Optional ByVal pRoles As IfcAggregate.ListOfIfcActorRole = Nothing)
        MyBase.New()
        ThePerson = pThePerson
        TheOrganization = pTheOrganization
        Roles = pRoles
    End Sub
    <System.ComponentModel.Category("IfcPersonAndOrganization"), _
        System.ComponentModel.Description("IfcPerson")> _
    Public Property ThePerson() As IfcPerson
        Get
            Return _thePerson
        End Get
        Set(ByVal Value As IfcPerson)
            _thePerson = Value
        End Set
    End Property
    <System.ComponentModel.Category("IfcPersonAndOrganization"), _
        System.ComponentModel.Description("IfcOrganization")> _
    Public Property TheOrganization() As IfcOrganization
        Get
            Return _theOrganization
        End Get
        Set(ByVal Value As IfcOrganization)
            _theOrganization = Value
        End Set
    End Property
    <System.ComponentModel.Category("IfcPersonAndOrganization"), _
        System.ComponentModel.Description("IfcAggregate.ListOfIfcActorRole"), _
        Editor(GetType(PropertyGridHelper.CollectionUIEditor), _
        GetType(System.Drawing.Design.UITypeEditor))> _
    Public Property Roles() As IfcAggregate.ListOfIfcActorRole
        Get
            Return _roles
        End Get
        Set(ByVal Value As IfcAggregate.ListOfIfcActorRole)
            _roles = Value
        End Set
    End Property
    ...
End Class
```

Figure 4: Sample Source Listing Generated From Mapping Process

5.3 VIEWING PROCESS

The source files generated from the mapping process are compiled into a single assembly (.dll) using the Visual Basic .NET command-line compiler (vbc) bundled with Microsoft .NET Framework[5]. The resulting library can be imported into any .NET programming environment. Figure 5 shows a sample C# application using the created assembly.

[5]available for free download at msdn.microsoft.com



```
using System;
using IFC2X_FINAL;

namespace IfcAssemblyTest
{
    class Class1
    {
        [STAThread]
        static void Main(string[] args)
        {
            IfcOccupant occupant = new IfcOccupant();
            occupant.OccupantType = IfcOccupantTypeEnum.LESSEE;
        }
    }
}
```

A declaration that this C# application uses the IFC assembly.
Objects can be created from types declared in the IFC assembly.
GlobalId
InitializeLifetimeService
LeeseOrRentalDate
LeeseOrRentalPeriod
Name
ObjectType
OccupationType
OwnerHistory
Site
TheActor
[property] IFC2X_FINAL.IfcOccupantTypeE

Figure 5: Application Using VB.NET class libraries created from the IFC 2X Schema

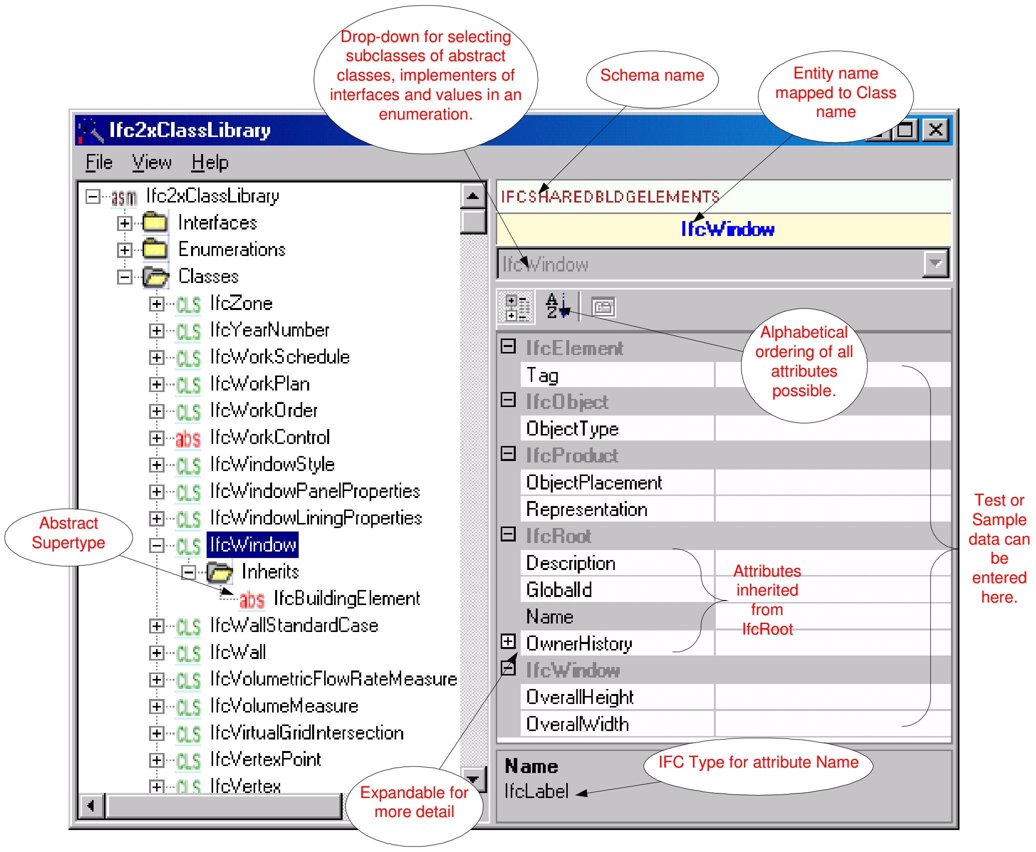
A Windows application was developed for viewing the generated class library. The application uses *Reflection*[6] to query the generated library and builds a tree view of classes, interfaces, enumerations and aggregates. Reflection can also be used as a basis for late-bound implementation. The application provides three possible views:

Implementation View – This groups the EXPRESS constructs into programming language constructs including Interfaces, Enumerations and Classes, irrespective of where they are defined in the IFC Architecture (see Liebich & See (1999) for more detail on the IFC Architecture). Figure 6 shows a screenshot of the implementation view.

Schema View – As for the Implementation View, with programming language constructs grouped according to their schema in the IFC Architecture. This would be useful for users interested in particular schema within the IFC model architecture. Figure 7 shows a screenshot of the schema view.

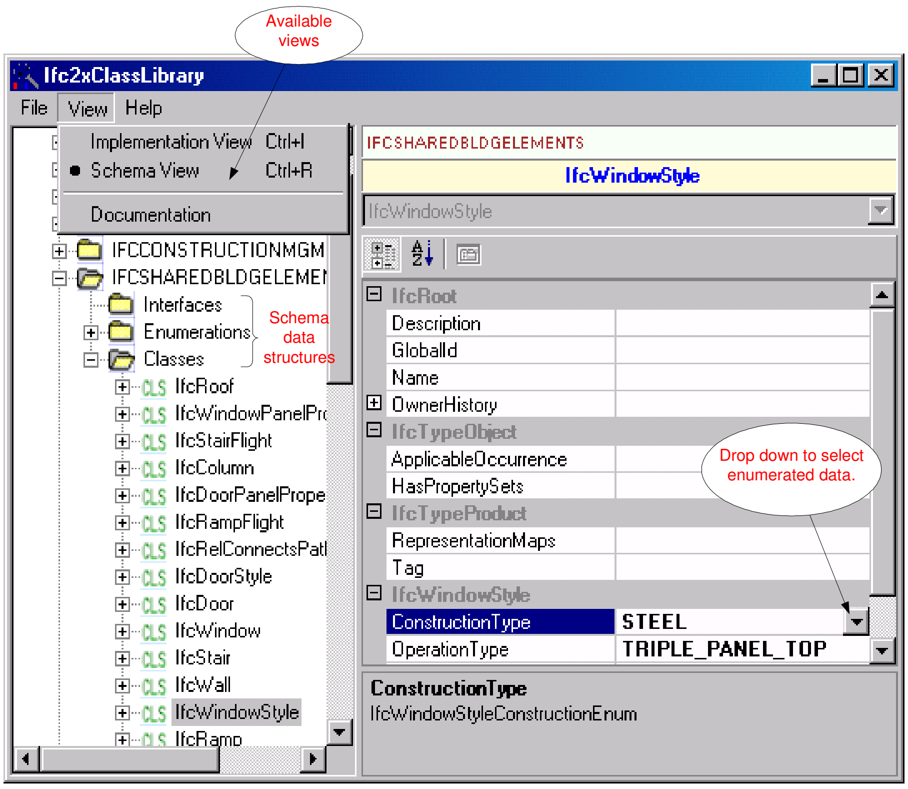
Documentation View – This can be viewed alongside the other two views. It synchronises the links in the IFC documentation (Adachi et al., 2003) with the language constructs they have been mapped to, either in the Implementation or Schema view. The IFC documentation provides EXPRESS definitions, EXPRESS-G representations, hyperlinked attributes, inheritance graphs, references. Figure 8 shows a screenshot of the documentation view.

[6]the ability of a program to observe and/or manipulate the inner working of its environment programmatically and dynamically



Name
IfcLabel

Figure 6: IFC Assembly Viewer - Implementation View

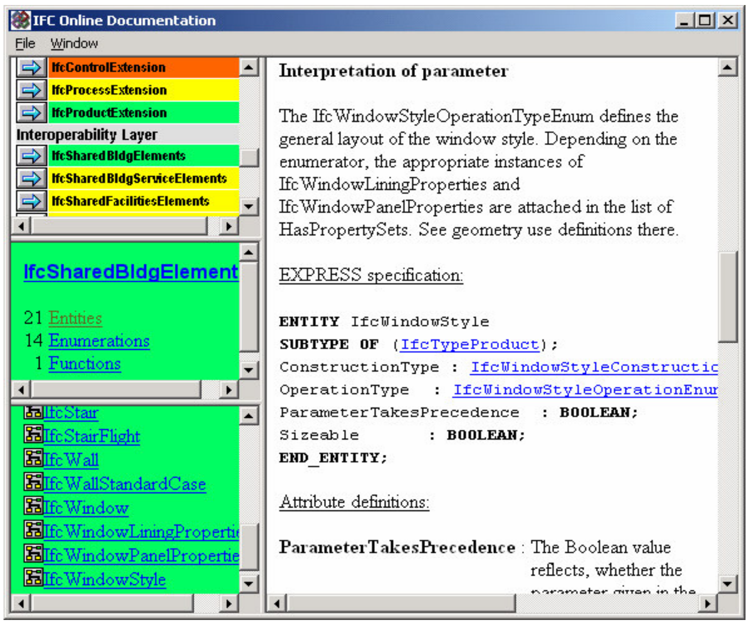


ConstructionType	STEEL
OperationType	TRIPLE_PANEL_TOP

ConstructionType
IfcWindowStyleConstructionEnum

Figure 7: IFC Assembly Viewer - Schema View

Both implementation and schema views uses a tree structure to represent the EXPRESS structure and provides forms for entering sample data as shown in the 3-part screenshot, Figure 9. Part (A) of the figure shows that each Entity (*Class*) on the tree can be expanded to reveal its relationships including super type (*inherits*), sub types (*subclasses*) and implemented interfaces (SELECT types that include the entity). Part (B) shows the form for entering data into SELECT type attributes of an entity while Part (C) shows the form for entering collection of (matching) types for aggregate type attributes. The Documentation view is synchronised automatically to the documentation page for the item that has input focus in the implementation and schema views.



Interpretation of parameter

The IfcWindowStyleOperationTypeEnum defines the general layout of the window style. Depending on the enumerator, the appropriate instances of IfcWindowLiningProperties and IfcWindowPanelProperties are attached in the list of HasPropertySets. See geometry use definitions there.

EXPRESS specification:

```
ENTITY IfcWindowStyle
SUBTYPE OF (IfcTypeProduct);
ConstructionType : IfcWindowStyleConstructic
OperationType   : IfcWindowStyleOperationEnur
ParameterTakesPrecedence  : BOOLEAN;
Sizeable        : BOOLEAN;
END_ENTITY;
```

Attribute definitions:

ParameterTakesPrecedence : The Boolean value reflects, whether the

Figure 8: IFC Assembly Viewer - Documentation View (HTML documentation source: Adachi et al. (2003))

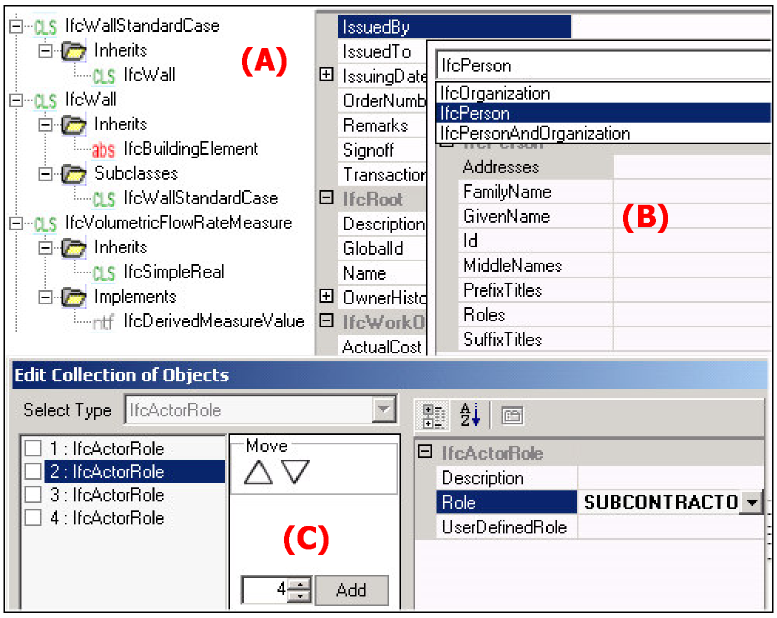
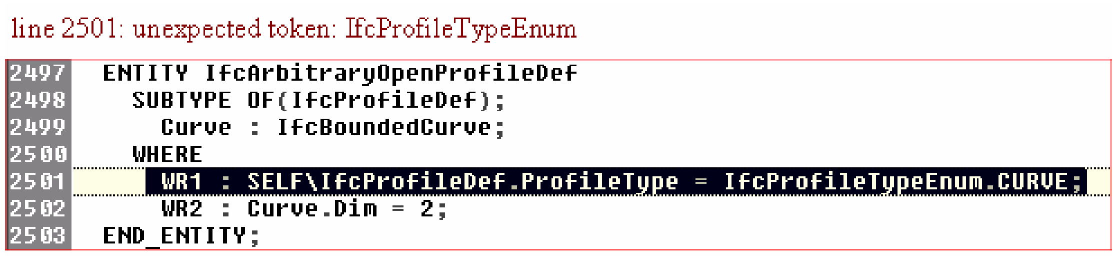


Figure 9: Working with the IFC Assembly Viewer

6 TESTING AND VALIDATION

The IFC Assembly Viewer was tested with the IFC product model release 2x. This involved providing the EXPRESS schema definition file for the product model as input to the IFC Assembly Viewer to translation, mapping and viewing. The translation process discovered an EXPRESS language error in the product model as shown in Figure 10. As shown in the figure, the use of *IfcProfileTypeEnum.CURVE* causes a syntax error because a simple factor such as a reference to an enumeration type is required and *IfcProfileTypeEnum* is an enumeration type not a reference to an enumeration type.



```
line 2501: unexpected token: IfcProfileTypeEnum
2497  ENTITY IfcArbitraryOpenProfileDef
2498    SUBTYPE OF(IfcProfileDef);
2499      Curve : IfcBoundedCurve;
2500    WHERE
2501      WR1 : SELF\IfcProfileDef.ProfileType = IfcProfileTypeEnum.CURVE;
2502      WR2 : Curve.Dim = 2;
2503  END_ENTITY;
```

Figure 10: EXPRESS Language error in IFC 2x

The error did not affect the mapping process as WHERE RULEs are not mapped in the current implementation. The mapping process produced a total of 675 source classes including:

- 459 classes representing entities;
- 116 enumeration types;
- 23 interfaces; and
- 77 aggregate types (1 Bag, 33 Lists, and 43 Sets).

The class library produced was tested and verified in Visual Basic.NET, C++ and C# environments using a simple test application. In all three environments, there was no noticeable difference in terms of execution speed or reliability of the library. The developers found the IFC Assembly Viewer complementary to the IFC documentation which provided detailed descriptive definition of attributes. One of the noted benefits was that the developers could easily discover the corresponding simple data type for simple types in the IFC model. For example:

- *IfcLabel* is treated as a simple String;
- *IfcHeatFluxDensityMeasure*, *IfcThermodynamicTemperatureMeasure* and *IfcEnergyMeasure* are treated as real numbers; and
- IfcMinuteInHour is treated as an integer.

7 SUMMARY AND CONCLUSIONS

This paper has presented a software solution that can make considerable contribution to the uptake of the IFC standard approach to integration in the AEC/FM domain by supporting implementers and highlighting implementation issues to the modellers. This tool automates the entire process of validating, translating and viewing EXPRESS models, particularly the IFCs, such that:

- Errors in the models can be picked up early during validation;
- Need for future language mapping efforts with its associated cost and time investment can be eliminated;
- It opens up the implementation options to cross-platform and multi-language environments;
- It eliminates the need for developers to learn EXPRESS to any great depth;
- It simplifies the task of evaluating the suitability of EXPRESS information models by providing single-point for accessing modelling and implementation views; and
- It creates working assemblies and views in seconds thereby reducing cost, time and effort for implementing future releases of the IFC model.

The implementation has been tested with IFC models though it theoretically should work with other EXPRESS-based models that use EXPRESS as IFC does. For example, IFC excludes multiple inheritance though EXPRESS allows it. See IAI (2001a) for more details on the use of EXPRESS within IFC.
The current IFC Assembly Viewer implementation can be enhanced by the addition of the following features:

- Model instance validation through implementation of rules and algorithms;
- Model persistence using STEP P21 (ISO, 1994c) neutral file exchange format and IfcXML (Liebich, 2001);
- DDL translation and relational database tables creation with choices of database system and model subsets; and
- Inclusion of late-binding for a mixed-binding implementation that presents developers with flexibility to choose the implementation method that fits their application needs.

This project was initiated to solve a problem facing a development team on another software research project - the need to understand the IFC model and evaluate its suitability for the application. It solved that problem. Further research revealed the problem is not peculiar to the team but industry-wide. The authors believe that this tool will go a long way in removing the barriers to IFC implementation, consequently clearing the road for better integration of systems within the AEC/FM domain.

8 REFERENCES

Adachi, Y., Forester, J., Hyvarinen, J., Karstila, K., Liebich, T. & Wix, J. (2003), ‘Industry foundation classes IFC2x edition 2 – online documentation’, Available online at http://www.iai-international.org/iai_international/Technical Documents /iai_documents.html.

Arnold, F. & Podehl, G. (1998), Best of both worlds - a mapping from EXPRESS-g to UML, *in* J. Bezivin & P.-A. Muller, eds, ‘The Unified Modeling Language

(UML) '98: Beyond the Notation', First International Workshop, Mulhouse, France, Springer Verlag, pp. 49–63. ISBN: 3-540-66252-9.

Chunyan, Y., Minghui, W., Nairuo, L., Yueting, Z. & Yunhe, P. (2003), 'Translating express language model into c language model', *SIGPLAN Not.* (6), 30–39. Available online at http://doi.acm.org/10.1145/885638.885646.

IAI (2001), *The EXPRESS Definition Language for IFC Development*, International Alliance of Interoperability.

ISO (1994*a*), Industrial automation systems and integration – product data representation and exchange – part 11: Description methods: The EXPRESS language reference manual, Technical Report ISO 10303-11, International Standards Organisation, Geneva, Switzerland.

ISO (1994*b*), Industrial automation systems and integration – product data representation and exchange – part 21: Implementation methods: Clear text encoding of the exchange structure, Technical Report ISO 10303-21, International Standards Organisation, Geneva, Switzerland.

ISO/IEC (1996), *ISO/IEC 14977:1996(E) – Extended Backus-Naur Form (EBNF)*, Internation Organisation for Standardisation, Case postale 56 1, rue de Varembé CH-1211 Genève 20 Switzerland.

Koning, H. P. (2003), pyExpress: A python open source EXPRESS Compiler/Code generator. 2003 NASA-ESA Workshop on Aerospace Product Data Exchange, NIST, Gaithersburg, Md, USA.

Lardet, S. & Lubell, J. (2001), *Java Express Parser User Documentation*, National Institute of Standards and Technology, US.

Libes, D. (1993), 'The NIST EXPRESS toolkit – introduction and overview'. NISTIR 5242, National Institute of Standards and Technology, Gaithersburg, MD, USA.

Liebich, T. (2001), XML schema language binding of EXPRESS for ifcXML, Technical Report MSG-01-001(Rev 4), International Alliance for Interoperability. Copy available at http://www.iai.org.uk/IFCXML.htm.

Liebich, T. & See, R. (1999), *IFC Object Model Architecture Guide*, International Alliance of Interoperability (IAI).

Loffredo, D. (1999), *Fundamentals of STEP Implementation*, STEP Tools, Inc. http://www.steptools.com/library/.

Parr, T. (2003), *AntLR (ANother Tool for Language Recognition) Reference Manual*, www.antlr.org. Available online at http://www.antlr.org/doc/ index.html.

Schenck, D. & Wilson, P. (1997), *Information Modeling : The Express Way*, Oxford University Press, UK. ISBN: 0195087143.

Wichmann, I. (2002), *Express2Perl Manual*, Bergische Universitat Wuppertal (uni-wuppertal.de).

APPENDIX D PAPER 4

Owolabi, A., Anumba, C., El-Hamalawi, A., Harper, C. & Parkin, P.(2004), "Implementation of PROMIS – An Integrated Product Information Management System", ***Unpublished reviewed paper***

Implementation of PROMIS - An Integrated Product Information Management System

ABSTRACT: Product modelling has been identified as one way to facilitate the integration of applications used in the construction industry. Standardisation bodies, researchers and software vendors have been developing product models and implementing the software infrastructures to enable exchange and sharing of construction product information between applications. Today, pockets of solutions exist in different stages of development but a coherent deployable system is still unavailable. This paper presents a software system designed to support the management of product information and its integration with construction project information. It proposes a modularised, parametric approach to building design as a foundation for the creation and management of product information throughout the building life-cycle. The developed system, **PRO**duct **I**nformation **M**anagement **S**ystem (PROMIS), comprises a Web application, a desktop application and a Web Service. The Web application provides controlled access to product information for publishing and browsing, the desktop client manages product data within projects and provides views to common desktop applications used in the industry. The Web Service acts as a broker between the Web application and desktop application by providing a dynamic link between product and project information. The paper examines the requirements and implications of adopting this system across the industry and concludes with its expected benefits.

KEY WORDS: *building modelling, product libraries, application integration, data sharing*

1 INTRODUCTION

Coulter (1998) suggests that the technological problem facing the construction industry is more complex than can be solved by an information standard but a holistic approach involving the management of knowledge and its sharing without loss of meaning and understanding. Without undermining the importance of an agreed information model, most industry practitioners would agree with this view. Most would also agree that 3D CAD modelling is a pre-requisite for construction integration. 3D CAD modelling tools are essential for design, construction and maintenance of building products. They can be used to observe and evaluate designs for buildability, usability, cost, safety and maintainability. Construction sequences, programmes and progress can be viewed and monitored during construction. Post-completion evaluations for facility management and eventual demolition can also be carried out.

The CAD software vendors have rightly implemented 3D representation but are reluctant to fully embrace standardised product modelling for the obvious fear of losing market share as individual members of the project teams would be liberated to use their preferred design tools. The consequence of the prevailing lack of interoperability is that the industry is still largely reliant on 2D drafting. Best et al. (2002) reckons that this is the *single largest problem* or obstacle to integration in the industry.

This paper suggests a not-so-radical shift in the approach to integration. Rather than starting a project with CAD design, building projects can begin with considerations of overall objectives and individual elements, components, sub-systems and systems, and the composition of these objects to achieve the set objectives. This decomposition of

buildings into objects can greatly benefit from building (product) modelling and easily lends itself to object-orientation whereby building (software) objects can encapsulate the *what* (information) and *how* (procedures) knowledge for their use in various applications. For example, a beam object can be developed to know its profile and how to draw itself in 3D CAD packages or cost itself in costing packages.
The paper describes the implementation of an integrated system which supports the modularisation of building designs and the development of building objects in a knowledge-based engineering environment. The system, **PRO**duct **I**nformation **M**anagement **S**ystem (PROMIS), consists of three applications namely:

- ProductWeb;
- ModelManager.NET; and
- Project-Product Web Service (P2Service).

ProductWeb is a Web-enabled database application for collection, storage and retrieval of construction product information. ModelManager.NET is a desktop application which encompasses building objects and facilitates the decomposition of building into objects for a specific project. P2Service is a Web Service which provides a link between product information in ProductWeb and building objects in ModelManager.NET projects.
Section 2 provides the background including a definition of the PROMIS approach to integration, product modelling and building product information. Section 3 presents the overall system architecture of PROMIS. The features and functionalities of ProductWeb, ModelManager.NET and P2Service are discussed in Sections 4, 5 and 6 respectively. The paper examines the implications of adopting this system across the construction industry in Section 7 and concludes in Section 8 with a discussion of the expected benefits.

2 BACKGROUND

2.1 THE PROMIS APPROACH TO INTEGRATION

The increasing complexity of building processes and communication (or lack of it) between the multidisciplinary project team members has highlighted the need for the integration of construction project information. A high level description of processes and actors, and their interaction with project information, as shown in Figure 1, demonstrates the need for the management of product information through the lifetime of the building. Anumba et al. (2000) examine the complexity of communications within a typical construction project team.
A major and critical part of project information is information about every product created, referenced or modified within the project. The dynamic nature of product information suggests that a central repository is required to avoid omissions, repetitions, confusion, misunderstandings, errors, delays and possibly litigations that are commonplace today. Such a repository would have to be developed on a universal building (product) model in order to support interaction with various other applications through the lifetime of the product. Figure 2 shows the types of applications that would need to interact with the repository.

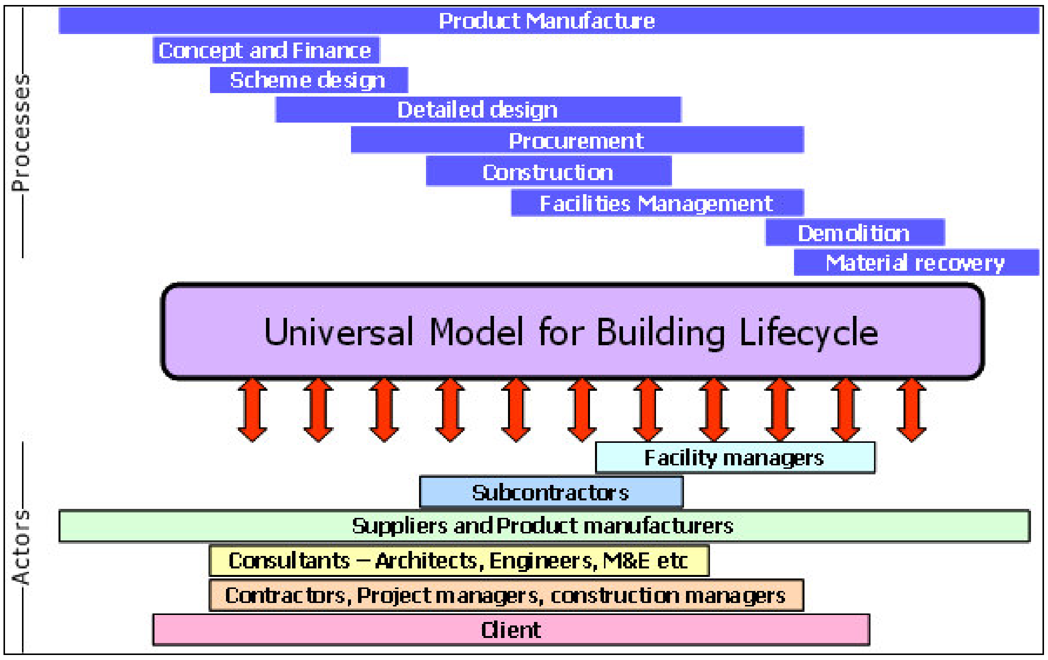


Figure 1: Information for the Complete Life Cycle of a Building

The current industry approach to integration is through shared project databases, communication between applications or a hybrid combination of both. Project databases provide electronic means of storing and distributing project information held in documents and database tables. This is an extension of electronic document management, which is usually limited to document storage, retrieval, versioning and approval, to a model-based approach in which the information is described and represented through an object model, and is contained in integrated databases (Rezgui & Cooper, 1998). Today, project databases can be Web-enabled to provide ubiquitous access and controlled interaction with the data. Harper (2003) discusses the general characteristics, uses and industry examples of project websites.

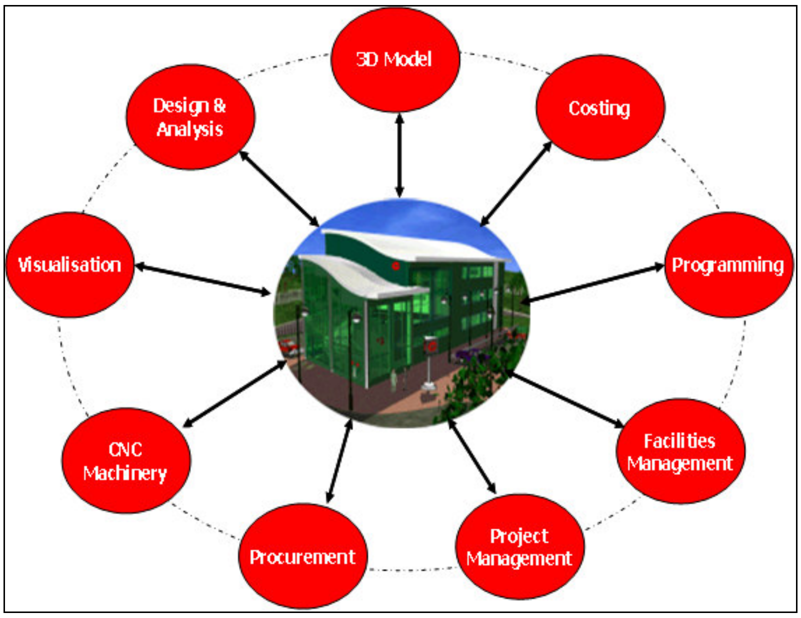


Figure 2: The Universal Building Model

Brown et al. (1996) identified three approaches to communication between applications. These include:

1. Specific software integrating a set of chosen applications;
2. Geometry in commercial CAD packages; and
3. Knowledge-based interfaces linking multiple applications.

Today, new technologies have been developed to ease the implementation but the approaches are largely the same. The PROMIS system is a hybrid solution focused on product information. It stores and manages product information available to the projects executed within an enterprise and provides a desktop application that interacts with common industry applications.

Ekholm & Fridqvist (1997) identified three pre-requisites for a computer-integrated construction process: a standardised product model, support for the product model by CAD software, and a standardised classification system for structuring information within the model. The International Alliance for Interoperability (IAI) is working relentlessly on the Industry Foundation Classes (IFC) product model development, CAD vendors claim support for the product model (IAI-ISG, 2003), and fairly standardised and usable classification systems are available today (Howard, 2001; Ekholm 2002). In spite of these developments no system has been deployed commercially with noticeable impact on the industry's prevalent fractured information systems. This is perhaps because the industry accepts product model support by CAD software as a pre-requisite for integration and thus is inadvertently limited by CAD software vendors.

There is little dispute that an electronic integration task can benefit from a common representation standard or product model. In other words, a mutually agreed information syntax and semantic simplifies the integration exercise. There is also general agreement that CAD software play a great role in building design visualisation. PROMIS exploits a well known software technology, the Microsoft Component Object Model (COM), to enable customisation, extension and interaction with industry software tools including CAD software.

2.2 PRODUCT MODELLING

A construction product is any object, manufactured, supplied or created, for incorporation into construction work or an AEC/FM project. A product model is a digital representation of real world objects held to facilitate the unambiguous transfer of the information between computer systems and to support information sharing (Eastman, 1999). It captures all perspectives of a product including technical information (such as sizes, materials, finishes, colours, etc.) and the dynamic evolution of the information through the life of the product including design, construction, and facility management through to demolition.

Clearly a product model is not a 3D visual model, rather it contains information for a 3D representation. It follows therefore that the development of a product model should start with a comprehensive definition of an information structure and not an ad-hoc extension of 3D models to incorporate additional product information. Leading to the IFC model, Liebich & Wix (2002) document the historical development of building product models highlighting the contributions of the best known international efforts, including COMBINE, CIMsteel, ATLAS, COMBI, ISO 10303 (STEP) Part 225, and BCCM.

The IFC model, defined by the International Alliance for Interoperability (IAI), is currently the most applicable product model to the construction industry. The IFC model is a specification of data structures to support a shared project model for data sharing across homogenous and heterogeneous applications. It represents building products (and abstract concepts such as space, organisations, and processes) and their information requirements in a neutral computer language - the EXPRESS modelling language. It is perhaps the single, most important contribution to the integration effort because it offers a global standard for interoperability across software platforms and hardware systems.

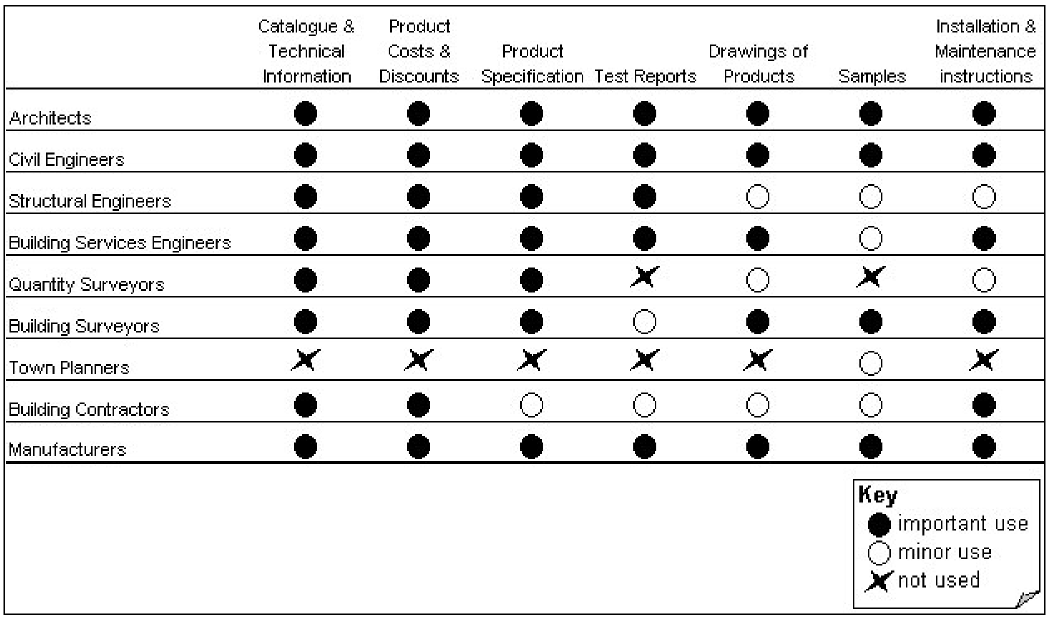
Although the IFC is rightly being developed through information rather than visualisation considerations, its adoption is being limited by architectural CAD tools which claim to produce product models from 2D/3D designs. Essentially, these tools produce documents, particularly drawings. While most integration solution starting with today's CAD tools can achieve integration through lossless exchange of documents between applications, optimal utilisation of a product model is achieved through shared access to common data. Only this can ensure availability of realtime information and eliminate the errors and information losses inherent in document translation. Major CAD vendors such as Bentley and Graphisoft are embracing a model-based approach to integration through the concept of building information modelling (BIM) (Cyon Research, 2003) but the effectiveness of these vendor-specific solutions remain to be proved (Ibrahim et al., 2004).

Ekholm et al. (2002) reported that IFC is focused on architecture and currently lacks many classes that are not represented in CAD such as those needed for cost estimating systems. This is not a criticism of the standard as it is continually in development and more classes are being added to every release.

The solution presented here was developed to meet some specific business integration objectives. An investigation of the an IFC specification in its current state, supported by IFC Assembly Viewer software tool (Owolabi et al., 2004), revealed that it cannot support the PROMIS system hence a non-standard product model which is equally incomplete has been developed. It is worth noting that the authors are not proposing the developed product model as an alternative to the IFCs; the PROMIS application framework is designed to be generic enough to adapt to the IFC model when it is sufficiently developed.

2.3 BUILDING PRODUCT INFORMATION

Building product information includes catalogue and technical information, specifications, drawings, samples, costs and discounts, sources of supply and delivery, test reports, installation instructions, maintenance instructions, and disposal/recycling requirements. What information is created or accessed would depend on a number of factors including procurement route, project phase, and roles played by different members of the project team. Figure 3 shows possible product information requirements for key project team members.



	Catalogue & Technical Information	Product Costs & Discounts	Product Specification	Test Reports	Drawings of Products	Samples	Installation & Maintenance instructions
Architects	●	●	●	●	●	●	●
Civil Engineers	●	●	●	●	●	●	●
Structural Engineers	●	●	●	●	○	○	○
Building Services Engineers	●	●	●	●	●	○	●
Quantity Surveyors	●	●	●	✗	○	✗	○
Building Surveyors	●	●	●	○	●	●	●
Town Planners	✗	✗	✗	✗	✗	○	✗
Building Contractors	●	●	○	○	○	○	●
Manufacturers	●	●	●	●	●	●	●

Key
● important use
○ minor use
✗ not used

Figure 3: Product Information Requirement By Building Project Team Members (adapted from DoE (1996))

Traditionally, product information is sourced from manufacturers, suppliers and information brokers. This information, usually contains images and text, and is delivered in printed or electronic catalogues. Electronic versions sometimes include 2D CAD drawings for inclusion in building design. Sourcing this information is costly and time consuming, yet it is difficult to keep up-to-date and is rarely connected to or used beyond the design phase. The availability of a product model which provides common definition for building products is key to integrating these essential sources of information. Owolabi et al. (2003) provide a detailed coverage of building product information, delivery methods, and shortcomings, and offer suggestions for improvement. ProductWeb, described later in Section 4, is an implementation of a typical online product library. It is based on the product model defined as part of this project.

3 PROMIS - SYSTEM OVERVIEW

This section presents an overview of PROMIS including its purpose, architecture, nature and operation, current status and planned developments.

3.1 PURPOSE OF PROMIS

The development of PROMIS is part of a research project investigating the creation, use and management of product information within a construction project. The aim of PROMIS is to provide an integration framework and supporting software tools for the management of whole-life product information.
Building construction is an information intensive exercise with many, possibly geographically-dispersed, participants. Usually participants address their information requirements discretely thus producing a fractured information system for the construction project. The need for a reduction, and possibly elimination, of the apparent inefficiencies of this disintegrated system and its attendant effects on project cost, quality and delivery time prompted this research and development activity.

3.2 PROMIS ARCHITECTURE

In its simplest form, the architecture consists of an instance of the ProductWeb Web application, ModelManager.NET desktop application, and P2Service Web Service. ProductWeb is a database application designed for product information management. ModelManager.NET facilitates building product modelling and the incorporation of product information in specific construction projects. P2Service provides a link between product information in ProductWeb and objects in ModelManager.NET projects. Figure 4 shows a simplified, high-level structure and interaction between these systems and external desktop applications.

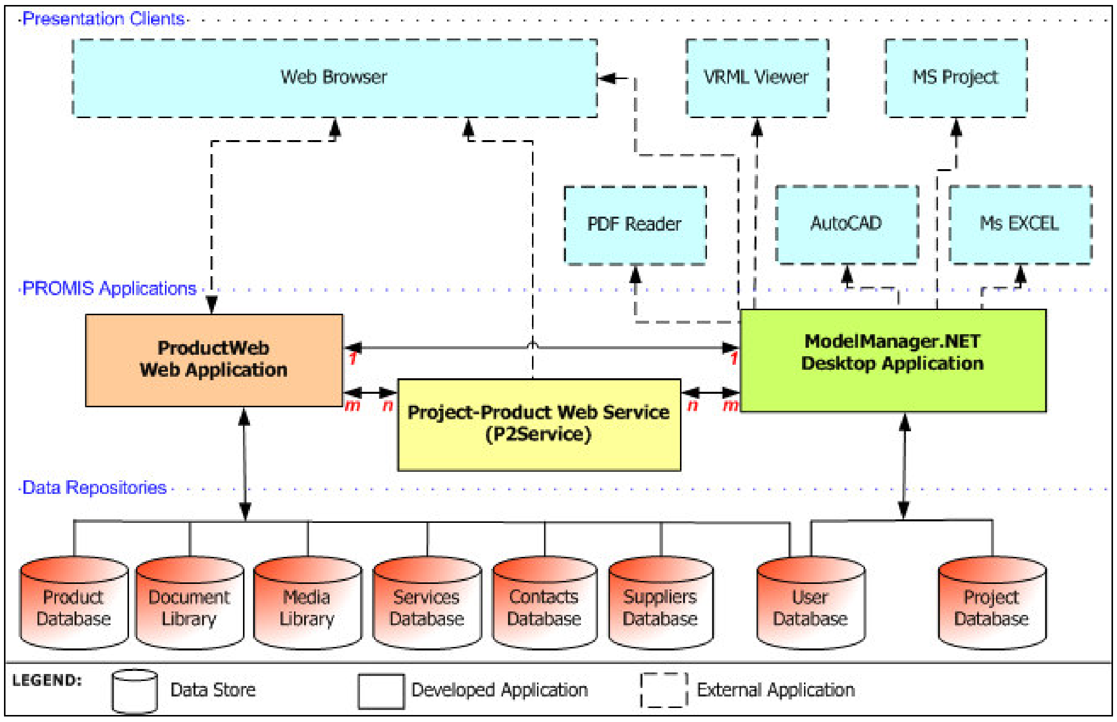


Figure 4: PROMIS - An Integrated Environment for Construction Product Information Management

Within the PROMIS framework, instances of ProductWeb or any other source of product information developed with a common product model can potentially interact dynamically with instances of ModelManager.NET (and vise versa) through instances of P2Service. Alternatively, a static link between a copy of ProductWeb and ModelManager.NET can be established in a stand-alone system, say for a product manufacturer.

Commonly used desktop applications (such as Internet Explorer, AutoCAD, Adobe Acrobat, etc.) are used as viewers for information obtained from the three subsystems. Web browsers are used for interaction with ProductWeb. They can also be used to view outputs from P2Service and ModelManager.NET. In this easily extensible framework, ModelManager.NET can provide views for other desktop applications commonly used in the industry including CAD, Virtual Reality (VR) and costing software.

3.3 COMMUNICATION TECHNOLOGIES

Communication between the developed applications is accomplished with standard Internet technologies. Ubiquitous HTTP Web protocol is used to transfer data using the universally accepted XML data format. This ensures that the different subsystems can communicate, even over the fire walls, irrespective of their locations on the globe.
In the past, integration projects, have employed tightly coupled, distributed computing protocol, such as Distributed Component Object Model (DCOM), Common Object Request Broker Architecture (CORBA), or Java Remote Method Invocation (Java RMI). These protocols are constrained by their dependence on vendor implementations, platforms, languages, and data-encoding schemes thereby limiting flexibility and interoperability.
Communication between ModelManager.NET and desktop applications has been implemented with the Component Object Model (COM). COM is a binary standard for software object interaction based on standardised interfaces. It allows software components to communicate with each other regardless of:

- What machine they are running on (as long as the machines are connected);
- What operating systems the machines are running (as long as they support COM); and
- What language the components are written in.

Zarli & Poyet (1999) suggest the best way to allow software applications to interoperate is to provide standard interfaces. More and more industry desktop applications are implementing COM interfaces to enable user customisation, communication and integration with other applications. Developers can now integrate applications from different vendors to share data and information more easily and seamlessly.
COM is currently limited to Windows applications but Microsoft is working on porting it to other platforms (Gui, 1998). This is presently not a problem as ModelManager.NET is developed for the Windows platform.

3.4 SECURITY SYSTEM

PROMIS implements a single security model supported by the User database. It uses an Authentication, Authorisation, and Accounting (AAA) system to:

1. identify users (authentication);
2. establish and restrict users' privileges (authorisation); and
3. maintain audit trail of users' activities (accounting)

Authentication information (username/password) are transported confidentially through Secure Sockets Layer (SSL) protocol. They are sometimes stored locally, with a specialised encryption algorithm, to allow users log-in automatically as desired.
It is recognised that the role of individuals and organisations can change as the project unfolds. These changes can be reflected through modification of authorisation details. Additional controls are implemented in individual systems to customise interfaces to suit users' roles.

3.5 PARAMETRIC PRODUCT SPECIFICATION

With parametric product specification, parameters provide definitions, values, properties and variations of a product which can be sufficiently detailed to support 2D and 3D representation, costing, quantity calculations, and so on. This method has been proven to offer considerable flexibility and storage efficiency in product library applications. GDL iCatalogues (Graphisoft 2000) and Trace Parts (*www.traceparts.com*) are ready examples. Both GDL iCatalogues and Trace Parts are built on parametric programming language, are accessible via the Web and integrate with most common CAD software for 2D and 3D viewing.
PROMIS building objects are parametric objects that can be built with any .NET language. Users can specify the object parameters in Model Manager.NET or load up actual manufacturer product parameters from ProductWeb. Object parameters contain enough details to support rendering in a number of views including 3D model, VR model, cost model, bill of materials, etc. Where the users machine has more than one installed application capable of rendering the views, users can choose their preferred application.
The three main components of PROMIS – ProductWeb, ModelManager.NET and P2Service – are presented in the next three sections of this paper.

4 PRODUCTWEB

ProductWeb is a Web-enabled database application for managing and disseminating product information. It was developed as a 3-tier, client-server, Web application using Microsoft Active Server Pages (ASP) technology. The objectives of the system are as follows:

- to enable suppliers of construction products and services
- to register, publish and manage their product and service information;
- to provide accurate and up-to-date information on registered products and services; and
- to manage all documents and media and their relationships to specific products.

4.1 PRODUCTWEB ARCHITECTURE

As shown in Figure 4, ProductWeb interacts with a number of data repositories which store documents and data tables. These include the product database, document library, media library, services database, contacts (persons/organisations) database, suppliers database, and user database. ProductWeb consists of a number of pluggable, fairly independent application modules that provide specific functionalities and manages exclusive data subsets within the data repositories as shown in Figure 5.

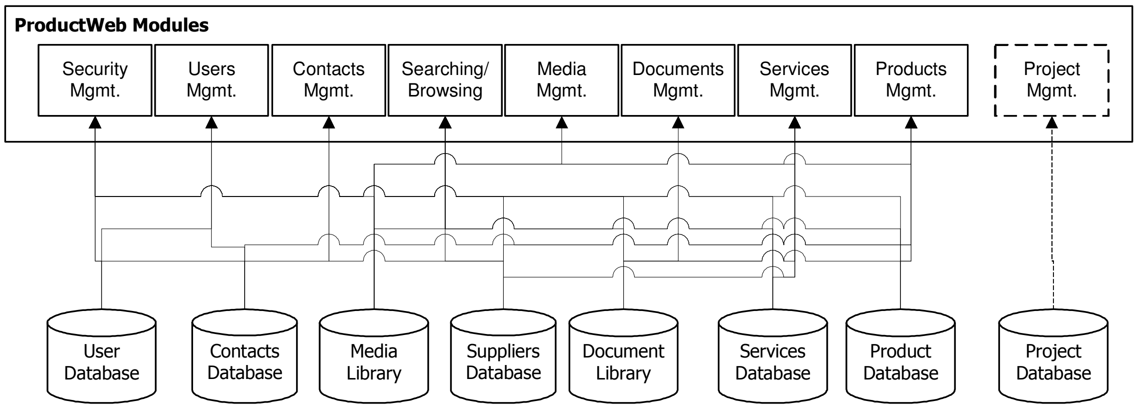
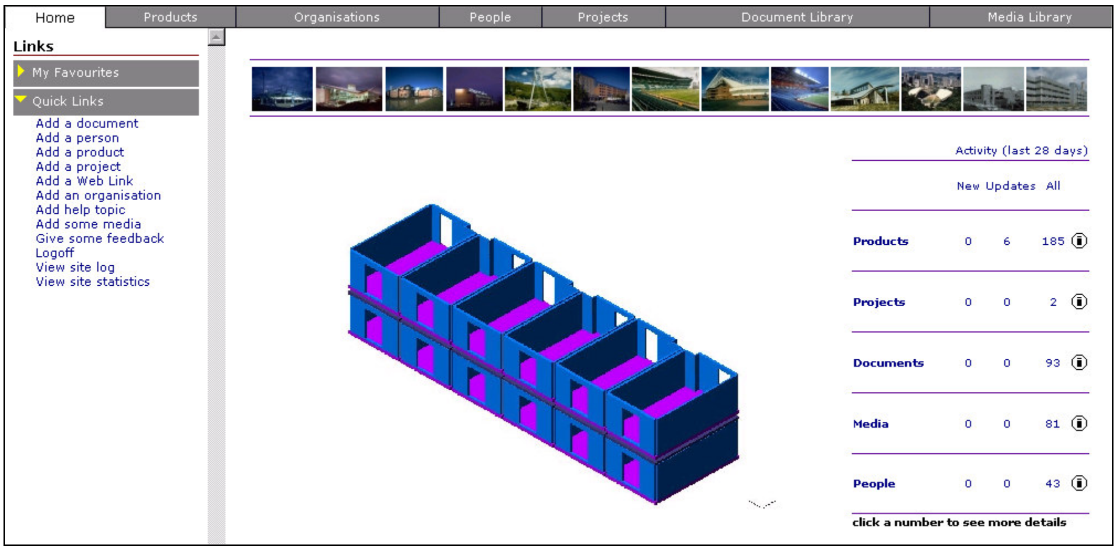


Figure 5: ProductWeb Modules and Data Repositories

This modular architecture allows graceful evolution, maintenance and flexible configuration. New versions of application modules can be introduced without affecting the rest of the system and the application can be configured to suit different product supplier's needs. The minimum configuration consists of the security management module, users management module, contacts management module, searching/browsing module and product management module. Figure 6 shows a screenshot of an installation of ProductWeb while Figure 7 shows the Web page for the Product Management module.



Activity (last 28 days)

	New	Updates	All
Products	0	6	185
Projects	0	0	2
Documents	0	0	93
Media	0	0	81
People	0	0	43

click a number to see more details

Figure 6: A Sample ProductWeb Installation

The Contacts management module handles registration of organisations and people for information and referencing purposes. All products, services, documents and media are linked to some organisation or person within an organisation. At least one user in a registered organisation is designated as *Domain Administrator*, by the *System Administrator* with privileges to add, activate, remove and deactivate users through the user management module. The domain administrator is also responsible for managing authorisations for users within his/her domain. Other users types, in order of system privileges, are *Advanced Author*, *Author*, *Project Manager*, *User*, and *Guest*. All users (except *Guest*s) can manage their own preferences, bookmarks, *quicklinks*, and other user-specific options. The Security Management module works in the background to enforce all user right specifications as well as logins, logouts, sessions, cookies, etc. All the other modules are linked to product information publishing.

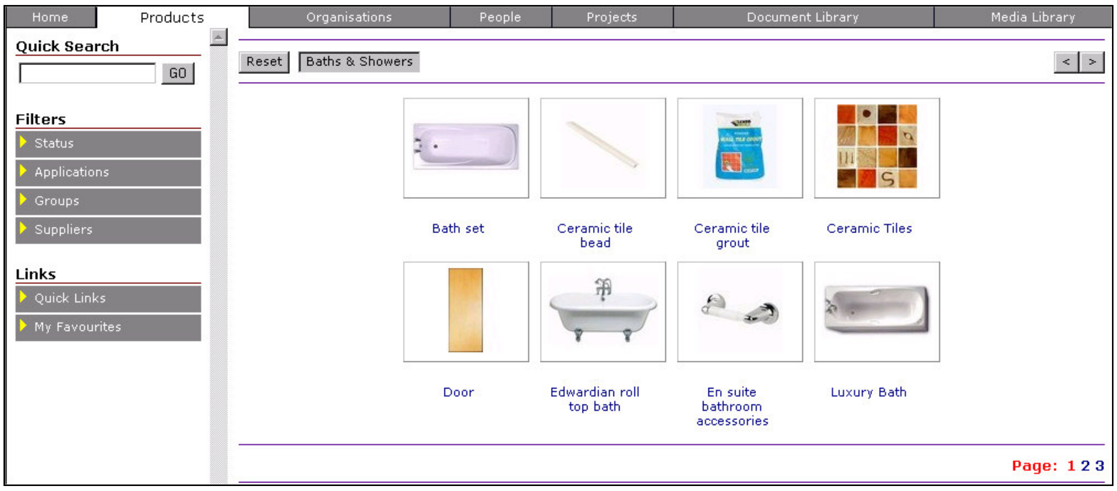


Figure 7: ProductWeb Product Management Web Page

An extension of ProductWeb to provide access to the Project database via the Project Management module is under development. This extension would provide limited access to ModelManager.NET projects as well as enable product manufacturers to expose assemblies that were created internally with ModelManager.NET, as modular products to the industry. Manufacturers would also be able to encapsulate or disclose internal details (for example, components) of the products as desired.

4.2 PUBLISHING PRODUCT INFORMATION

ProductWeb provides facilities for data creation through definition and population of product information as well as uploading and linking of documents and media to products. Services can also be associated with products. For example, a product like the Universal Beam would have processing services such as short blasting, priming, cutting, and drilling associated with it. Typical product information managed by ProductWeb includes product data sheets, technical literature, method statements, health and safety information, images, etc.

Product definition (or metadata) is provided by the supplier where there is no existing definition in ProductWeb. These definitions are non-standard and admittedly insufficient for the global industry. However, it is expected that this allowance to use existing definitions would encourage supplier and user buy-in and enable the developers to concentrate resources, in the meantime, on tackling the technical challenges. The IFC is being developed to address this problem of non-standard definition of *things* within the industry. Once completed, a business case for global interoperability can be made and appropriate translations for IFC-compliance can be implemented.

The main feature that distinguishes ProductWeb from traditional online product catalogues is its extension of product information to include, where applicable, parameterised product profile or section properties. These are included to provide a more detailed description of the products and sufficient information for drawing the

product representation in a 3D CAD or Virtual Reality modelling environment. Product costs and materials can also be calculated from the specified parameters.
Both product definition and information are stored in the *Product database*. Documents, media and services metadata are similarly stored in their respective databases as shown in Figure 5. Presently, the documents and media, created with standard editing software, are uploaded and stored on a file server but there are plans to move them to their respective databases for security and ease of administration.
Products, Documents and Media are globally uniquely identified for ease and persistence of reference. Broken-links are the bane of Web-based applications. The transient nature of today's online product information adversely affects usability and greatly limits their long-term application. Ideally, product information should exist for as long as the physical product exists *somewhere*. ProductWeb implements an incremental-growth document publishing system which provides a predefined workflow and revision process to ensure published information is never changed. New revisions can be released as required. The use of external Web links to legacy data is allowed where intellectual property is an issue but users are strongly encouraged to avoid them as much as possible.

4.3 BROWSING AND SEARCHING

ProductWeb implements software development best practices in Web information delivery to enhance the users' experience. Information filtering, drill-down search, indexed views, textual search, favourites, and quick links are some of the incorporated features. Interactive and intuitive information filtering reduces the amount of information presented to the user quickly to a manageable level thus preventing information overload. A context-sensitive information filtering system is implemented in ProductWeb. It loads filter sets depending on the user's past interaction with and current context in the application. Table 1 shows some filter sets and sample filters that are available to a user from the Product Management module's Web page.

Table 1: Product Management Module Filter Sets

Filter Set	Examples
Status	Draft, Awaiting Approval, Approved, Published
Applications	Architraves, Boards, Damp Proofing, Fasteners, Fixtures, Lighting, etc.
Groups	Accessories, Computing, Furniture, Hardware, etc
Suppliers	ABC Timbers Ltd.

These filters provide a drill-down information query system which allows stepped investigation of product information, from the general to specifics. The implemented filters are complemented with pictorial indexes to further guide user selections. Traditional textual search is also supported. Quick links and Bookmarks are some of the additional features implemented to speed up access to popular information. A context-sensitive help system along with a *How-do-I* quick reference manual is available to provide instructions on using the application.
Earlier research by Owolabi et al. (2003) into search methods for construction products identified five possible search methods including search by keyword (or textual search), search by classification, parametric search, performance-based search and knowledge-based search. The last two are still in the domain of theoretical research (Jain &

Augenbroe, 2002; Denzinger et al., 2002). ProductWeb has implemented textual search and classification by user-defined filter sets and filters. The implementation of parametric search in ProductWeb (and ModelManager.NET) is planned for the next version of the system.

5 MODELMANAGER.NET

ModelManager.NET is a configurable, extensible, product modelling, database-driven, desktop application for product prototyping, building modelling, client briefing, collaborative product development, cost analysis and construction applications integration. It is designed to facilitate the specification of product objectives through its decomposition into individual building elements required to meet those objectives. Its primary functions can be summarised as creation, extraction, visualisation and integration of project-related product information.
ModelManager.NET is aimed at the Construction Industry, particularly offsite modular construction although it can be re-used for other industries such as transport and shipbuilding. It is currently only appropriate for precision pre-engineered solutions and requires detailed engineering knowledge. The application development was motivated by the need to deliver client value through quality assured products; built with minimal waste, promptly and efficiently.

5.1 MODELMANAGER.NET ARCHITECTURE

ModelManager.NET is being developed with Visual Basic.NET. Its application structure comprises simply of a modelling engine (*MMCore*) and pluggable modules. Figure 8 shows the application structure or approach to ModelManager.NET development. The main functions of MMCore include:

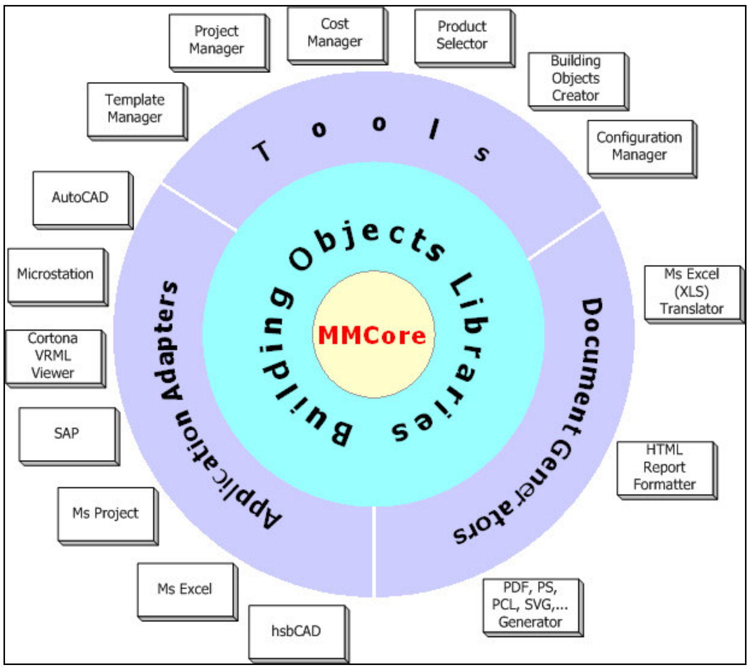


Figure 8: ModelManager.NET Application Structure

1. Creation of building objects;
2. Creation and management of relationship between building objects;
3. Building object persistence;
4. Authentication, authorisation and accounting management; and
5. Registration of and interfacing with pluggable modules;

5.2 BUILDING OBJECTS

Building products can range from simple fasteners to entire buildings. It is important to capture information about these simple and complex objects as well as their relationships and usage context. For example, the software behaviour of a joist on its own would differ from when it is incorporated into a floor assembly. The joist dimensions may need to change to accommodate changes in the dimensions of the floor. ModelManager.NET categorises building objects according to their composition and use. There are 5 groups including: *Building*, *Module*, *Assembly*, *Component*, *Fastener*, and *MiscItem*. Components, for example, Floor joist, are put together into *Assemblies* like Floor system, which in turn can be assembled into a *Module*. A building can bring together many different building objects. Fasteners, such as screws, rivets, etc., and Miscellaneous objects, such as lamp shades, mirrors, water taps, etc., can be added to any of the other groups as appropriate.
Simple objects such as *Component*, *Fastener* and *MiscItem* are treated as atomic. They can be associated with actual supplier products sourced from ProductWeb directly or through P2Service. ModelManager.NET uses *Templates* to provide definitions for complex objects such as *Assembly*, *Module* and *Building*. Templates can be dynamically composed to match the product complexity. For example, a bedroom *Module* template may reference a number of wall *Assembly* templates which in turns references simple components and so on. Figure 9 shows the composition of structural objects from simple rivets to wall joists, to wall assembly and a complete module.

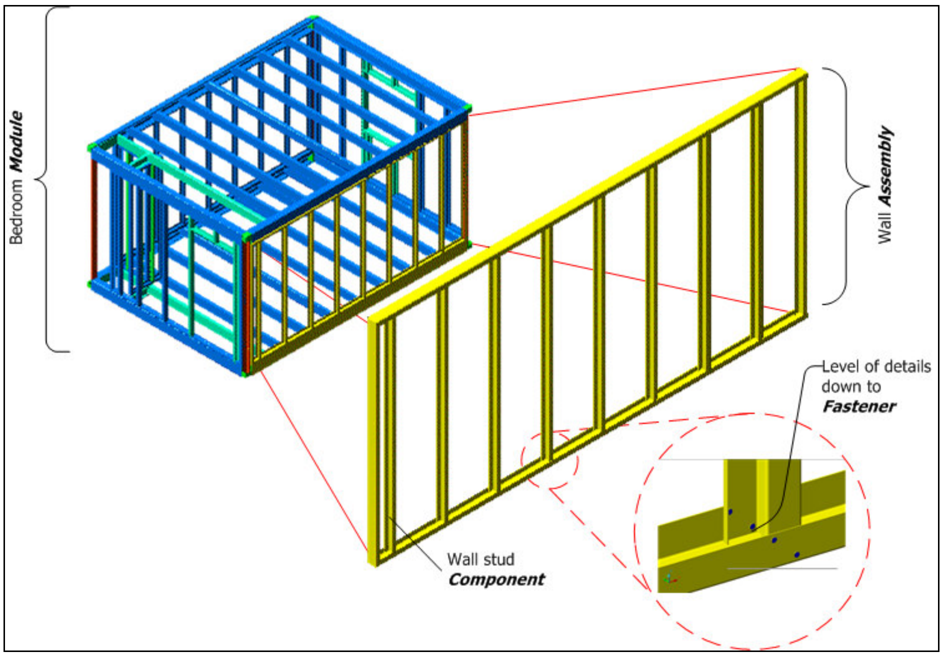


Figure 9: Building Object Composition

5.3 BUILDING OBJECTS RELATIONSHIPS

Building Objects are created within the context of a ModelManager.NET *Project* Object. A hierarchical object relationship is used to reflect the composition of the building objects. Simple objects are added as leaf nodes. Complex, *template-based*, objects create simple objects when the template is processed.
Processing a template-based object applies the encapsulated detailed engineering knowledge to create simple objects which may be grouped into folder-like containers to reflect the composition or for ease of organisation. A user may be allowed to add custom objects to processed templates, for example, an extra joist to a standard floor. Figure 10 shows this relationship as represented in the ModelManager.NET user interface.

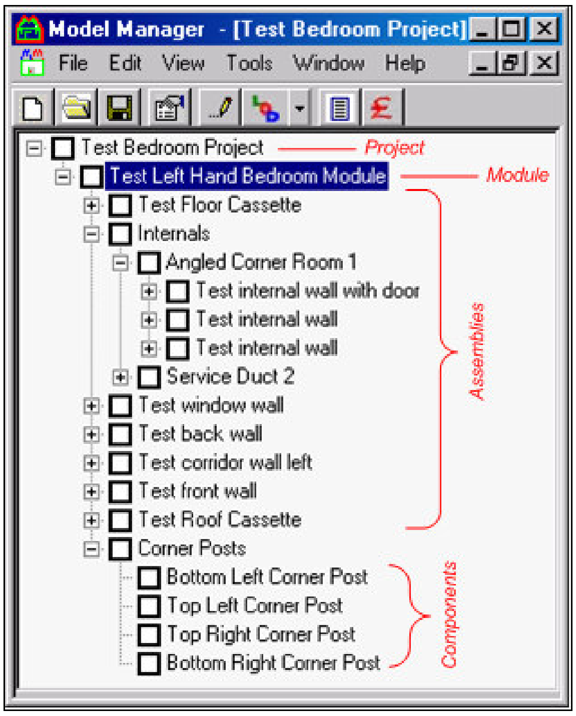


Figure 10: Hierarchical Building Object Relationships

Every object, including *Project* object is globally uniquely identified. This does not only guarantee that every building object in a project can be traced through its lifetime but it also facilitates model merging. Model merging enables project team members to work on parts of the model independently as the results can be merged later without conflicts.

5.4 OBJECT PERSISTENCE

ModelManager.NET is database-driven. Building objects and their relationships are stored in a central project database which can be accessed remotely by a number of ModelManager.NET instances. This is done to facilitate collaboration, concurrent engineering, sharing of model information and data security.
Where connection to a network is unavailable off-line working is supported with XML-structure project files. This is particularly useful for project archiving, client briefing

and marketing purposes. Local databases can be maintained alongside the live project databases and users can switch between or synchronise these persistence options (*Local DB*, *Live DB*, *and Off Line*) as required.

5.5 PLUGGABLE MODULES

ModelManager.NET implements a modular architecture which ensures the system can grow dynamically with minimal configuration rather than extensive redevelopment efforts. Possible extensions include:

1. Building Object libraries;
2. Document Generators;
3. Application Adapters; and
4. Tools.

BUILDING OBJECT LIBRARIES

There are hundreds of types of products available to the construction industry. Typically a construction business would reference a small subset of this. There are three ways to filter or extend the list of products that are available to a ModelManager.NET installation:

1. Building object library registration;
2. Objects configuration; and
3. Setting product status in ProductWeb.

Building object libraries can be registered to dynamically extend the application and provide additional building objects without the need for rebuilding the application. Since building objects are associated with actual products, all related products are consequently accessible to the ModelManager.NET. Unregistering building object libraries achieve the reverse.
On the other hand, an administrator can dynamically configure the system for specific users, as required by the business, to include or exclude any set of building objects. This is particularly important for enforcement of business decisions and management of the application interface as the number of implemented building objects grow.
ModelManager.NET can only access products that are *Approved* or *Published* in ProductWeb. A *Domain Administrator* can thus change the status of products to make it available or unavailable to ModelManager.NET.

APPLICATION ADAPTERS

Brown et al. (1996) described construction industry software packages in use then as monolithic legacy applications and suggested an implementation strategy that involves modelling and coding of *Object Wrappers*. Today, these applications have changed considerably. Commonly used industry software packages provide one or more interfaces for integration. Particularly, the COM interface has gained considerable popularity with software vendors and developers alike. It gives competitive advantage to the software vendor and developers can now integrate and customise standard applications more easily and virtually seamlessly.

There is however still a need for Object Wrappers or Application Adapters in this architecture because different industry applications expose different API. Application Adapters have been implemented to manage communication between ModelManager.NET and industry applications, translating the applications' function calls into ModelManager.NET function calls and vise versa. Currently, there are Application Adapters for AutoCAD and Cortona VRML Viewer for 3D model viewing and interaction respectively. Figure 11 shows the 3D model view of the structural elements of a Floor *Assembly* in AutoCAD and the same model view in Cortona. Application Adapters for other industry applications such as MicroStation for 3D model view, Ms Project for construction programme, hsbSoft for machine programming and SAP for stock management and material procurement are at different implementation stages.

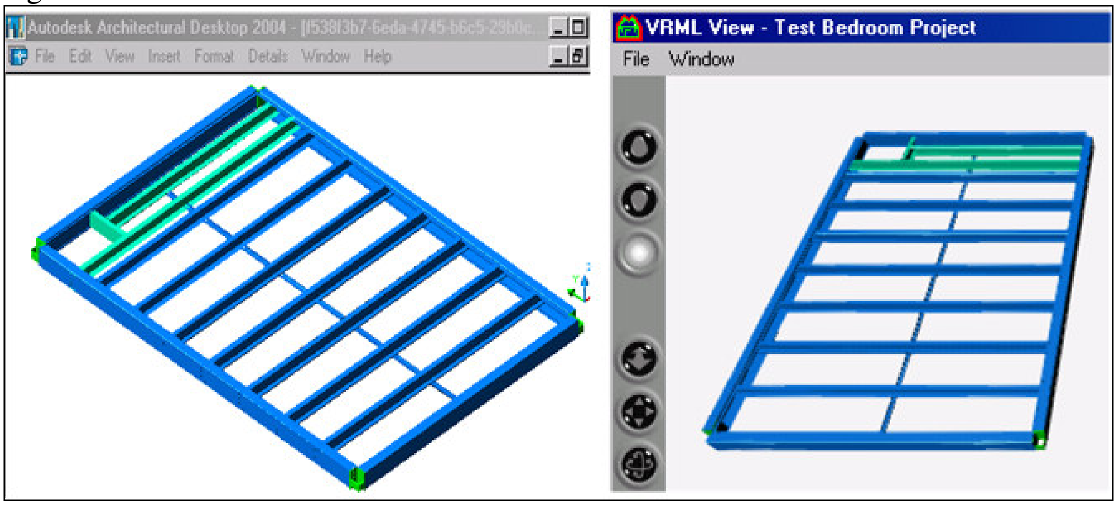

Autodesk Architectural Desktop 2004 - [f538f3b7-6eda-4745-b6c5-29b0c
File Edit View Insert Format Details Window Help
VRML View - Test Bedroom Project
File Window

Figure 11: 3D Model View in AutoCAD and Cortona VRML Viewer

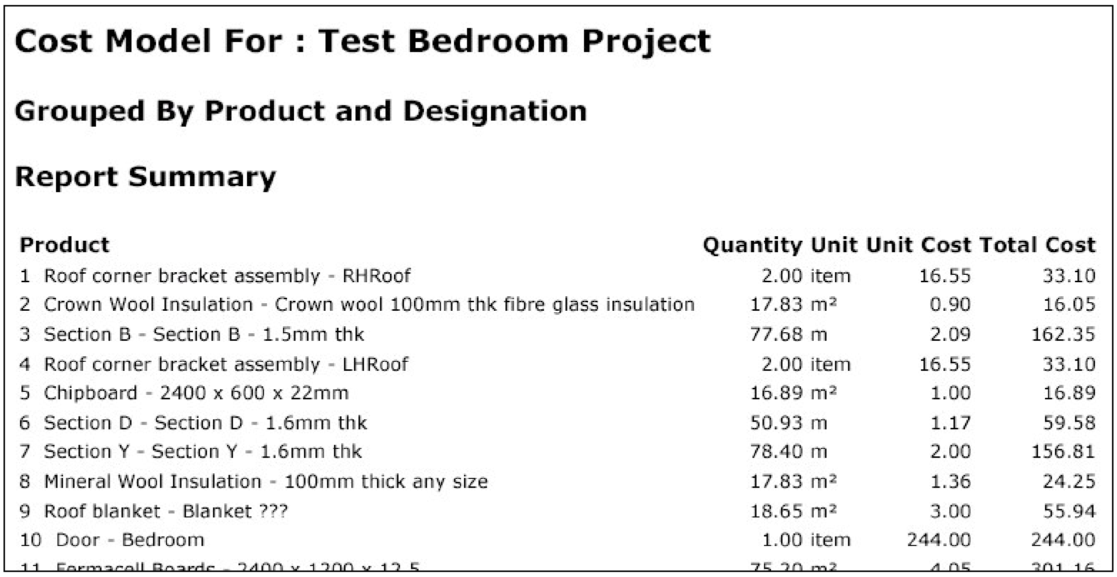
Information is transferred between the applications through function calls rather than files hence changes to information held in ModelManager.NET is automatically reflected in adapted applications in realtime. Adapted applications act as information viewers though users can interact with information presented in them as if the applications created the information. Changes made in adapted applications are however not communicated back to ModelManager.NET[7]. The idea is to make information changes deliberate and transparent to the users by ensuring information is changed in only one place.
During startup, ModelManager.NET searches the user machine for installed and applicable software and registers their adapters. User's preferences can be configured or changed as desired.

DOCUMENT GENERATORS

Document Generators support the reporting capabilities of ModelManager.NET by providing reports in standard file formats for viewing and printing. They receive XML data from ModelManager.NET and translate the data into required formats using XSL. The current implementation of ModelManager.NET includes a PDF, HTML and XLS

[7]Except SAP which may change pricing information and require ModelManager.NET to adjust its prices accordingly.

(MS Excel file) format generators. Figure 12 shows a sample HTML summary cost report (grouped by products).



Cost Model For : Test Bedroom Project

Grouped By Product and Designation

Report Summary

Product	Quantity	Unit	Unit Cost	Total Cost
1 Roof corner bracket assembly - RHRoof	2.00	item	16.55	33.10
2 Crown Wool Insulation - Crown wool 100mm thk fibre glass insulation	17.83	m^2	0.90	16.05
3 Section B - Section B - 1.5mm thk	77.68	m	2.09	162.35
4 Roof corner bracket assembly - LHRoof	2.00	item	16.55	33.10
5 Chipboard - 2400 x 600 x 22mm	16.89	m^2	1.00	16.89
6 Section D - Section D - 1.6mm thk	50.93	m	1.17	59.58
7 Section Y - Section Y - 1.6mm thk	78.40	m	2.00	156.81
8 Mineral Wool Insulation - 100mm thick any size	17.83	m^2	1.36	24.25
9 Roof blanket - Blanket ???	18.65	m^2	3.00	55.94
10 Door - Bedroom	1.00	item	244.00	244.00
11 Formacell Boards - 2400 x 1200 x 12.5	75.20	m^2	4.05	301.16

Figure 12: Sample HTML Cost Report

The PDF Generator uses Apache FOP (Formatting Objects Processor, *xml.apache.org/fop*), an output independent print formatter driven by the standardised XSL formatting objects (XSL-FO). Apache FOP is a Java application which can produce other formats including PCL, PS, SVG, AWT, MIF, and TXT from the same inputs. The PDF Generator translates the XML Data from ModelManager.NET to create a formatting object (FO) tree which is read by Apache FOP and rendered as a specified output, PDF in this case. The PDF Generator provides a number of XSL transformations for different reports. New transformations can be added or existing ones modified to suit specific users' needs. Additional generators or even third-party generators can be incorporated as required.

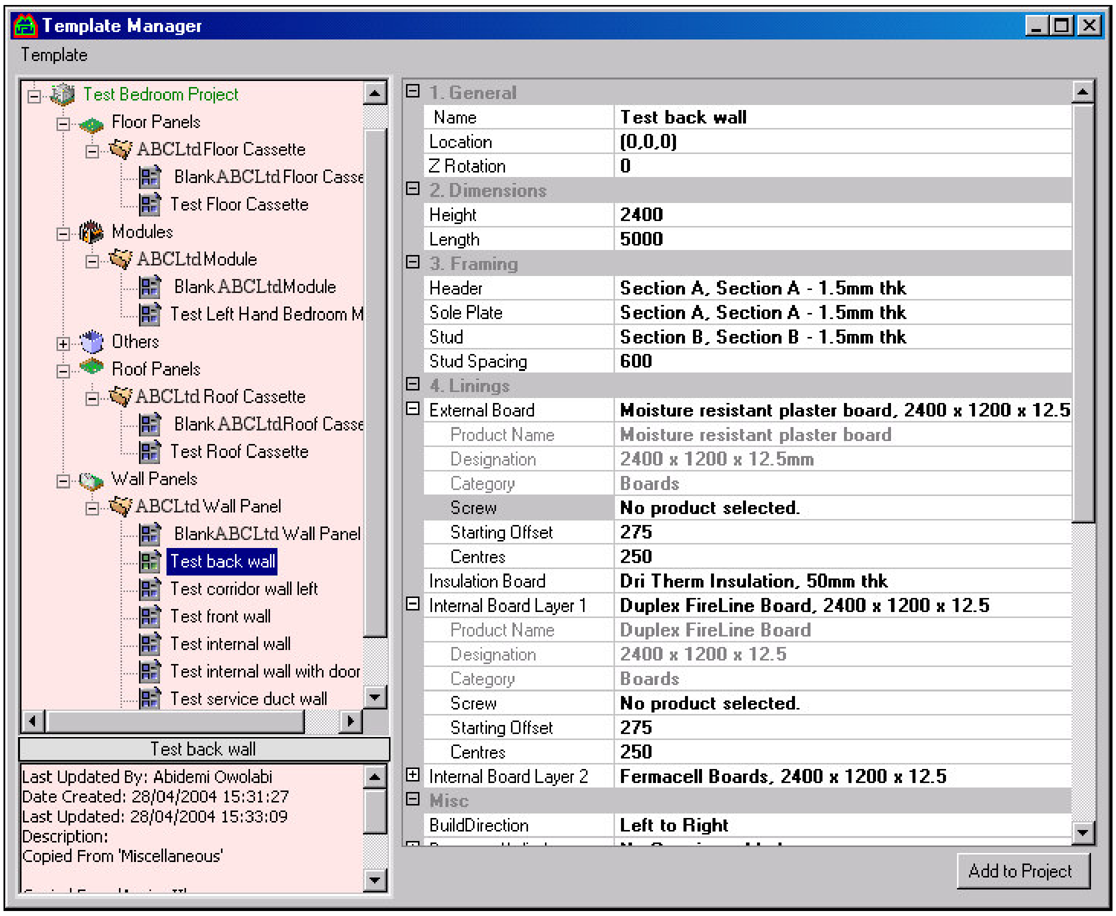
TOOLS

Tools add considerable value to ModelManager.NET. Currently, the implemented tools include: Template Manager, Project Manager, Cost Manager, Product Selector, Building Object Manager, and Configuration Manager. These tools can be replaced with newer versions or third-party tools as required provided the appropriate interfaces are implemented. Although ModelManager.NET can be configured to exclude any tool, some tools provide functionalities that are primary to the application. Examples of such tools are Template Manager, Project Manager, and Product Selector.
Template Manager is an essential tool which facilitates the creation and management of template-based, complex building objects. It allows users to:

- Define new templates;
- Validate a template;
- Save and Retrieve templates;
- Import templates from other projects;

- Link templates for complex objects; and
- Generate and add templates to projects.

Template Manager cooperates with Project Manager for template imports and with Product Selector for selection and association of products with templates. Figure 13 shows a screenshot of the Template Manager.



Test Bedroom Project
- Floor Panels
  - ABCLtd Floor Cassette
    - BlankABCLtd Floor Casse
    - Test Floor Cassette
- Modules
  - ABCLtdModule
    - BlankABCLtdModule
    - Test Left Hand Bedroom M
- Others
- Roof Panels
  - ABCLtd Roof Cassette
    - BlankABCLtdRoof Casse
    - Test Roof Cassette
- Wall Panels
  - ABCLtd Wall Panel
    - BlankABCLtd Wall Panel
    - Test back wall
    - Test corridor wall left
    - Test front wall
    - Test internal wall
    - Test internal wall with door
    - Test service duct wall

Test back wall

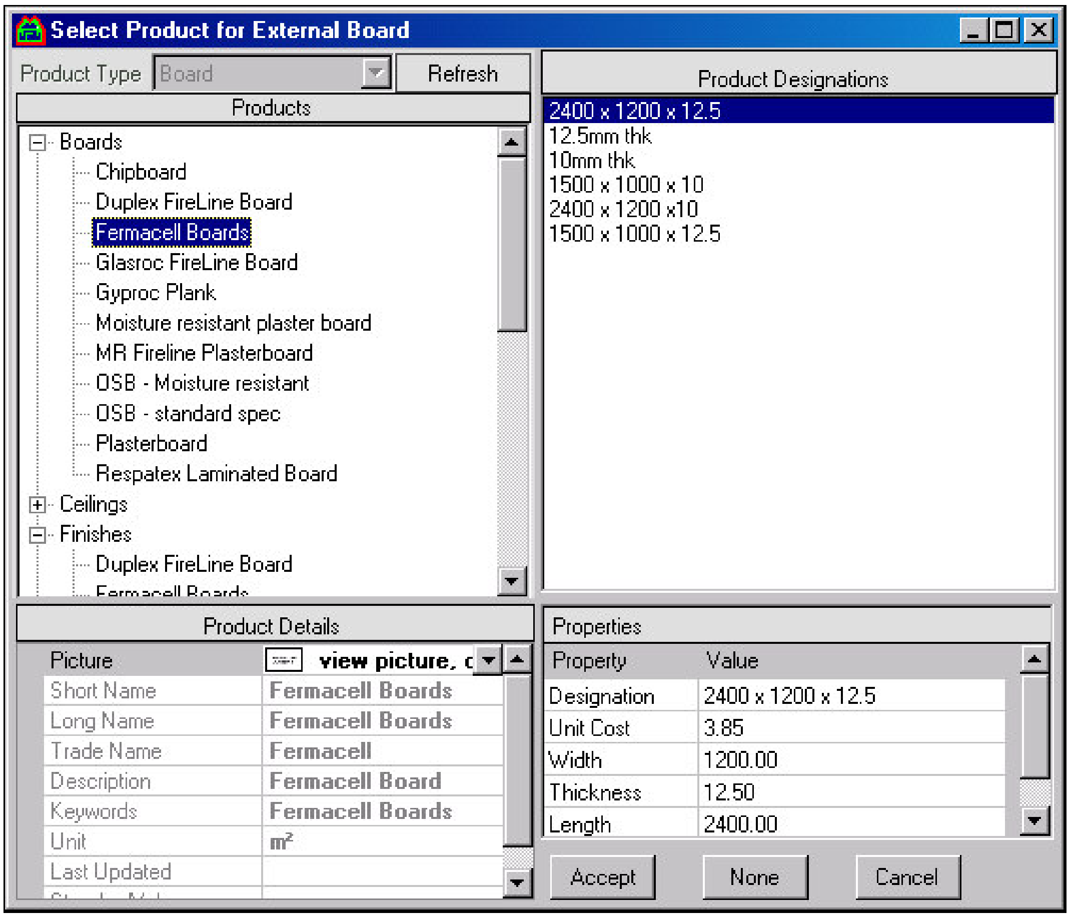
Last Updated By: Abidemi Owolabi
Date Created: 28/04/2004 15:31:27
Last Updated: 28/04/2004 15:33:09
Description:
Copied From 'Miscellaneous'

1. General	
Name	Test back wall
Location	(0,0,0)
Z Rotation	0
2. Dimensions	
Height	2400
Length	5000
3. Framing	
Header	Section A, Section A - 1.5mm thk
Sole Plate	Section A, Section A - 1.5mm thk
Stud	Section B, Section B - 1.5mm thk
Stud Spacing	600
4. Linings	
External Board	Moisture resistant plaster board, 2400 x 1200 x 12.5
Product Name	Moisture resistant plaster board
Designation	2400 x 1200 x 12.5mm
Category	Boards
Screw	No product selected.
Starting Offset	275
Centres	250
Insulation Board	Dri Therm Insulation, 50mm thk
Internal Board Layer 1	Duplex FireLine Board, 2400 x 1200 x 12.5
Product Name	Duplex FireLine Board
Designation	2400 x 1200 x 12.5
Category	Boards
Screw	No product selected.
Starting Offset	275
Centres	250
Internal Board Layer 2	Fermacell Boards, 2400 x 1200 x 12.5
Misc	
BuildDirection	Left to Right

Figure 13: Screenshot of Template Manager Pluggable Tool

Project Manager provides the interface for managing and reporting projects. It is the broker between MMCore and every pluggable module that accesses stored projects including Template Manager and ProductWeb Project Manager. It provides facilities for opening, archiving, synchronising, deleting, copying and moving projects.

Product Selector provides an interface to ProductWeb or P2Service for product selection. It filters the available product list according to the type of building object sought and presents information necessary for product selection including technical specification, costs, and pictures. Selected products are automatically linked and uniquely identified with the building objects. Figure 14 shows the Product Selector screenshot opened for *External Board* selection in the *Test back wall* template shown in Figure 13



Product Type: Board

- Boards
  - Chipboard
  - Duplex FireLine Board
  - Fermacell Boards
  - Glasroc FireLine Board
  - Gyproc Plank
  - Moisture resistant plaster board
  - MR Fireline Plasterboard
  - OSB - Moisture resistant
  - OSB - standard spec
  - Plasterboard
  - Respatex Laminated Board
- Ceilings
- Finishes
  - Duplex FireLine Board
  - Fermacell Boards

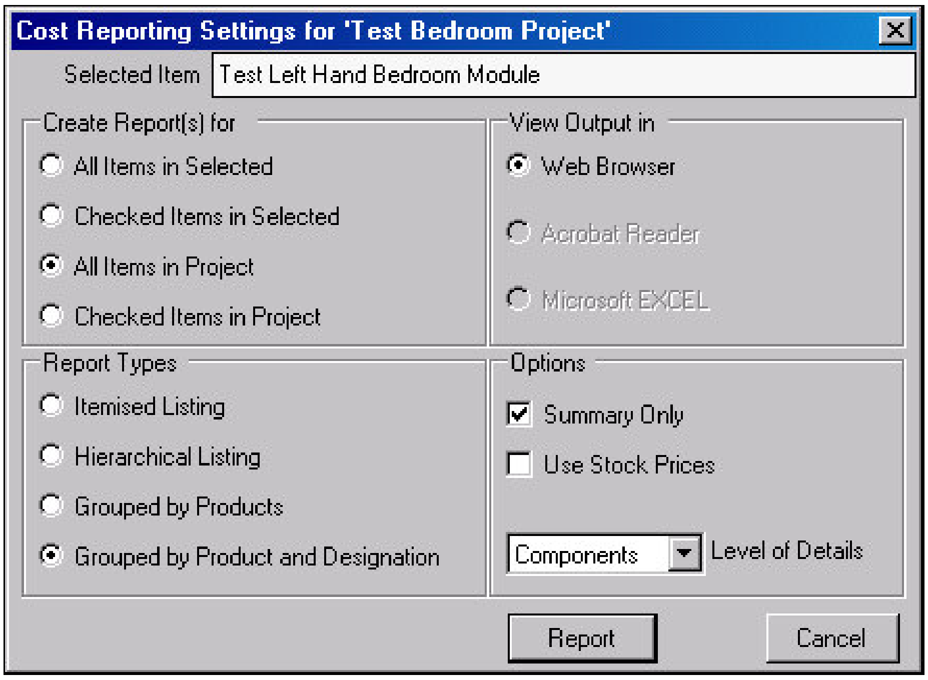
2400 x 1200 x 12.5
12.5mm thk
10mm thk
1500 x 1000 x 10
2400 x 1200 x10
1500 x 1000 x 12.5

Picture	view picture, c
Short Name	Fermacell Boards
Long Name	Fermacell Boards
Trade Name	Fermacell
Description	Fermacell Board
Keywords	Fermacell Boards
Unit	m^2
Last Updated	

Property	Value
Designation	2400 x 1200 x 12.5
Unit Cost	3.85
Width	1200.00
Thickness	12.50
Length	2400.00

Figure 14: Screenshot of Product Selector Pluggable Tool

Cost Manager offers cost analysis facility. It recognises the complexities involved in costing and allows users to perform sophisticated *what-if* analysis. Prices in different projects can be reviewed, compared, copied and pasted for different costing scenarios. Sources of product prices including ProductWeb, and master materials databases can also be queried for up-to-date prices. With this tool, the effect, on projects, of a change in price for a specific product, material or labour can be seen almost instantaneously.
MMCore includes a *Cost Reporter* which generates cost reports for *costable* building objects based on users selections and building objects prices within the project. Figure 15 shows the cost report settings dialog used to generate the HTML cost report shown in Figure 12.
Building Object Creator is a wizard-like tool provided for ModelManager.NET tool developers to quickly define building objects. It generates Visual Basic.NET classes and appropriate database tables based on parameters, profiles and extrusions defined by the user. It also provide a visual aid to help users verify the building objects profiles with test parameter values supplied interactively by the users. Facilities for automatic compilation of building objects into libraries and dynamic registration of the libraries with ModelManager.NET installations are being implemented.
Configuration Manager allows a *Domain Administrator* or *System Administrator* to specify building objects that are applicable to different ModelManager.NET installations and for specific users. It can be used to complement the user authorisation system for specifying user's access and editing rights and available building objects and tools.


Cost Reporting Settings for 'Test Bedroom Project'

Selected Item: Test Left Hand Bedroom Module

Create Report(s) for: All Items in Selected / Checked Items in Selected / [✓ All Items in Project] / Checked Items in Project

View Output in: [✓ Web Browser] / Acrobat Reader / Microsoft EXCEL

Report Types: Itemised Listing / Hierarchical Listing / Grouped by Products / [✓ Grouped by Product and Designation]

Options: [✓ Summary Only] / Use Stock Prices

Level of Details: Components

Figure 15: Cost Report Settings Dialog

5.6 MODELMANAGER.NET EXTENSIONS

ModelManager.NET is in active development. Some of the features reported here are still in prototype stage. Future extensions would incorporate supplier management, facility management, design rules verification, thermal and acoustic analysis, electronic commerce and fully-featured Web interface. Presently, ModelManager.NET is being deployed for a modular construction business and feedback from users is being reviewed and incorporated into the application development process.

6 PROJECT-PRODUCT WEB SERVICE (P2SERVICE)

P2Service is a Web Service which primarily acts as product information broker between ProductWeb and ModelManager.NET. Like ModelManager.NET, it is being developed mainly with Visual Basic.NET although it incorporates a batch file uploader service developed with C#.

P2Service provides many-to-many integration of product information sources and ModelManager.NET. Although this system has been tested with ProductWeb as a product information source, theoretically it can be adapted for any other source of product information with a shared product model. Again, the IFC can play a very important role here.

Web Services provide a powerful mechanism for integrating disparate IT systems because they employ commonly adopted standards and technologies. Unlike the component-based distributed computing protocols like CORBA and DCOM, Web Services expose application interfaces using platform-independent, language-neutral, ubiquitous Web protocols, HTTP. The application interfaces and messages are further encoded in the universal-accepted Extensible Markup Language (XML) to ensure ease of integration of heterogeneous platforms. Web Services provide modular, message-oriented, loosely-coupled, and easily scalable architectures for todays application-

application integration. Detailed introduction to Web Services and comparison with other distributed computing technologies can be found in Basiura et al. (2001).
P2Service registers sources of product information managing the connection information and their product metadata. Required product metadata can be derived from building objects using .NET *Reflection*. Reflection is the ability of a program to observe and/or manipulate the inner working of its environment programmatically and dynamically through an application programming interface (API). Product data are also validated through the building objects' *Validate* function to ensure required information is present in the data stream.
Using its Product Selector tool, ModelManager.NET contacts P2Service with an XML-formatted request for product information. P2Service finds the relevant product information sources, collates the matching product data into an XML response and forwards it back to ModelManager.NET. Given the amount of data involved, the response is cached until the user manually requests a refresh. Subscription/publishing features for automatic updates are under development. In a partnered environment, this service architecture can be extended to include publishing of RFI (Request for Information), RFQ (Request for Quotation), and RFP (Request for Proposal).
Early trials suggest there is noticeable performance degradation when a single instance of ModelManager.NET is connected to an instance of ProductWeb over P2Service. Research to determine the optimal configuration or participation to justify the presence of the P2Service broker in a dynamic configuration is being carried out.

7 DISCUSSION - PROMIS AND THE CONSTRUCTION INDUSTRY

PROMIS, as developed, suited a particular business need, the main requirement of which was to develop an end-to-end solution for a business manufacturing modules for use in the construction industry. This need was satisfied by developing a bespoke solution to the problem at hand.
The building objects developed were all designed with the particular need of the manufacturing business concerned and not necessarily that of any other manufacturing business or indeed of the wider construction industry. Similarly the templates developed, which represent more complex type objects such as walls, floors and modules were designed with the particular business need in mind. Using this approach it was possible to demonstrate that many of the industry's issues in attempting to define and improve its processes and to improve efficiency in design and construction can be achieved, albeit on a tightly defined scope. Clearly some of the system developed is private to the business that supported the development, however much of the solution is applicable to the wider construction industry. The methodology is highly appropriate, as is the ProductWeb, the core of the ModelManager.NET, and the integration (or application) adapters to interact with industry standard applications. The rest of this discussion is focused on how pervasive use of this system within the construction industry could be achieved.
The construction industry is characterised by its diverse, disparate and fragmented nature. This poses many problems when attempting to produce solutions that have industry wide application. We see this clearly in the industry's attempt to create an information standard, where much progress has been made but still the ubiquitous solution is far off. Part of the problem lies in the timescale required to agree on any

standard, which can run into decades. This does not sit well with the dynamic nature of the industry.
Another issue is the construction industry's propensity for in-house solutions, which by definition are generally to satisfy the needs of an individual organisation or a small section of the industry. In developing the framework for PROMIS, we have specifically kept in mind the needs of the wider construction industry because therein lies the key to the ongoing success of the system and the business for which it was initially developed. The evolutionary path needed to move the PROMIS system to an industry-wide solution may well provide the key to a general approach to system development that the industry can use.
The architecture of the solution as described in Section 2 is the key to the deployment methodology within the industry. It consists of the core application, which deals with the integration and interaction of the tools created, information brokering between applications, implementation of the security policies, and management of access to the core databases. The toolsets, templates and building components exist outside of this core and as long as future extensions are developed with the correct interfaces to the core they can be developed independently of the original developers.
This leads immediately to a significant advantage over traditional software development within the industry in that it allows many developers to develop tools and templates to interact with many others via the core and to have their tools and templates immediately accessible to the application. The developer would have the option to make his or her contribution available to the core or keep it private should this be required for business reasons. This would lead to situation similar to the Linux development where maintenance of the core program is the responsibility of a specific organisation. This organisation would also offer validation and verification services to the industry should the developments be moved into the core application.
This methodology could improve the development of standards such as the IFCs by allowing a staged development of building objects (such as cladding sheets, beams and bolts etc), templates for complex assemblies of building components (such as walls, floors cladding etc) and tools. In the initial stage the developer can *ignore* standards either because of the time required to implement or because the standard does not yet cover the scope of the problem. This gives immediate business benefit as the time frame for the business solution is minimised. The second stage would be for the development to be exposed to industry scrutiny should the developers wish it. The third stage would be for the development to be incorporated into the core of the application and the standard adopted.

8 CONCLUSION

The PROMIS system is currently being evaluated for efficiency, effectiveness and performance by a construction product manufacturer and full deployment is expected before the end of the year. When fully deployed, it would be the first commercial system of its kind in the construction industry offering an end-to-end solution. Some expected benefits and motivation for the target product manufacturer are:

- Reduced marketing costs, IT costs, rework, planning times, lead times, operating costs, staff requirement and waste;
- Faster response to enquiries and change requests;
- Faster access to information and reuse of data;

- Improved product quality, productivity, and business flexibility; and
- Improved capture of design decisions and engineering knowledge.

The authors are confident PROMIS can deliver these benefits and industry-wide integration of product information. Considerable development effort has been invested in PROMIS but much more is required to achieve the goal of cradle to grave product information management.

9 REFERENCES

Anumba, C. J., Bouchlaghem, N. M., Whyte, J. & Duke, A. (2000), 'Perspectives on an integrated construction project model', *International Journal of Cooperative Information Systems* (3), 283–313.

Basiura, R., Batongbacal, M., Bohling, B., Clark, M., Eide, A., Eisenberg, R., Hoffman, K., Loesgen, B., Miller, C., Reynolds, M., Sempf, B. & Sivakumar, S. (2001), *Professional ASP.NET Web Services*, Wrox Press Ltd, Arden House, 1102 Warwick Road, Acocks Green, Birmingham B27 6BH, UK. ISBN 1-861005-45-8.

Best, R., de Valence, G. & C, L. (2002), *Design and Construction*, Building in Value, Butterworth Heinemann, Oxford, chapter 15, pp. 291–305. ISBN: 0750651490. Chapter contributed by Rabee M Reffat.

Brown, A., Rezgui, Y., Cooper, G., Yip, J. & Brandon, P. (1996), 'Promoting computer integration construction through the use of distribution technology', *ITcon*. **Vol. 1**

Coulter, E. (1998), Construction modelling methodologies for intelligent information integration (COMMIT), Technical Report GR/K15459-ITE, Construct I.T.

Cyon Research (2003), *The Building Information Model: A Look at Graphisoft's Virtual Building Concept*, Cyon Research Corporation, 8220 Stone Trail Drive Bethesda, MD 20817-4556 USA, www.cyonresearch.com. Research Whitepaper.

Denzinger, J., Sinz, C., Avenhaus, J. & Uchlin, W. K. (2002), Teamwork-PaReDuX: Knowledge-based search with multiple parallel agents, *in* 'Proceedings of International Conference on Massively Parallel Computing Systems, Ischia, Italy'.

DoE (1996), Feasibility of a library of building design objects, Technical report, Construction Sponsorship Directorate, Department of Environment, UK.

Eastman, C. M. (1999), *Building Product Models : Computer Environments Supporting Design and Construction*, CRC Press LLC, 2000 N.W. Corporate Blvd., Boca Raton, Florida 33431, USA. ISBN 0-8493-0295-5.

Ekholm, A. (2002), Principles for classification of properties of construction objects, *in* K. Agger, P. Christiansson & R. Howard, eds, 'Distributing Knowledge in Building', CIB w78 Conference, Aarhus School of Architecture, Denmark. ISBN 87-90078-36-5.

Ekholm, A. & Fridqvist, S. (1997), Design and modelling in a computer integrated construction process - the BASCAAD project, *in* R. Junge, ed., 'CAAD Futures '97', Kluwer Academic Publishers.

Ekholm, A., Häggström, L., Tarandi, V. & Thåström, O. (2002), Application of IFC in sweden, Technical Report Working Report A15 - Project 98309, The Swedish Building Centre.

Graphisoft (2000), The online marketing and e-commerce of building components with graphisoft's GDL technology, Technical report, Graphisoft. White Paper.

Gui, D. (1998), 'Components, COM, and ATL', Published at msdn.microsoft.com. Last accessed April, 2004.

Harper, C. (2003), ICT at work for the LSE ProCurement chain: Putting information technology and communications technology to work in the construction industry - good practice guidelines, Technical Report WP3-T3500-R3501, ProCure - Esprit 29948.

Howard, R. (2001), Classification of building information - european and IT systems, *in* 'Construction Information Technology, International Conference: IT in Construction in Africa', CSIR, Building and Construction Technology, pp. 9–1 to 9–14.

IAI-ISG (2003), *International Overview of IFC-Implementation Activities*, IAI-Implementation Support Group, Organisation website: http://www.iai.fhm.edu/iai_isg/. Last visited 10 June, 2004.

Ibrahim, M. M., Krawczyk, R. J. & Schipporiet, G. (2004), A web-based approach to transferring architectural information to the construction site based on the BIM object concept, *in* 'CAADRIA 2004 Conference: Culture, Technology and Architecture' Seoul, Korea.

Jain, S. & Augenbroe, G. (2002), Performance-based electronic product catalogues for the building industry, *in* 'Proceedings of e-2002, eBusiness and eWork Conference in Prague'.

Liebich, T. & Wix, J. (2002), Standard analysis - current AEC situation - building models, Technical Report IST-2001-32035, prodAEC - European Network for IT in Architecture, Engineering, and Construction.

Owolabi, A., Anumba, C., El-Hamalawi, A. & Harper, C. (2004), 'Implementation of an IFC assembly viewer', *ASCE Journal of Computing in Civil Engineering*. Accepted for publication (in press).

Owolabi, A., Anumba, C. J. & El-Hamalawi, A. (2003), 'Architecture for implementing IFC-based online construction product libraries', *ITcon* **Vol. 8**, 201–218. Special Issue IFC - Product models for the AEC arena, Copy available online at http://www.itcon.org/2003/15.

Rezgui, Y. & Cooper, G. (1998), 'A proposed open infrastructure for construction project document sharing', *Electronic Journal of Information Technology in Construction (ITCon)* , **Vol. 3**, 11–24.

Zarli, A. & Poyet, P. (1999), Distributed architectures and components for the virtual enterprises, Technical report, Centre Scientifique et Technique du Batiment (CSTB), Sophia Antipolis, France (zarli/poyet@cstb.fr).

Corus UK Ltd
Swinden Technology Centre
Moorgate, Rotherham
South Yorkshire S60 3AR

Centre for Innovative Construction Engineering (CICE)
Department of Civil & Building Engineering
Loughborough University
Loughborough
Leics, LE11 3TU

Owolabi A. A. Development of an Integrated Product Information Management System 2004 CICE