Integrated Systems for Maintenance Management

By

Mohammad Ahmad Hassanain

B.Sc. (Honors) Architectural Engineering, King Fahd University of Petroleum and Minerals, 1993

A Thesis Submitted in Partial Fulfillment

of the Requirements for the Degree of

Doctor of Philosophy

in

Faculty of Graduate Studies
Department of Civil Engineering
Construction Engineering and Management

We accept this thesis as conforming to the required standard

The University of British Columbia

February 12, 2002

© Mohammad Ahmad Hassanain, 2002
In presenting this thesis in partial fulfillment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Department of Civil Engineering
The University of British Columbia
Vancouver, Canada

Date February 12, 2002
Abstract

The research presented is entitled Integrated System for Maintenance Management. It explores the intersection of two topic areas: Information Technology (IT), which is providing significant improvement to business practices throughout many industries, and Facilities Management (FM), which is dealing primarily with sustainability of capital investments.

The research recognizes the fragmentation in the practice of FM, and proposes a framework that aims at being a systematic and generic reference to the practice of maintenance management. The framework served to identify avenues throughout the practice of maintenance management where strategies and procedures could be implemented to improve operations and increase efficiencies. The framework is proposed to standardize business processes description, the activities that need to be undertaken and the methodology of how and what information needs to be communicated between processes.

The dissertation focuses on integrating data and knowledge through the development of shared project models that maintain a combination of product and process views of the project. Development of these models followed a methodology similar to that of the International Alliance for Interoperability (IAI) to develop data standards in the form of Industry Foundation Classes (IFCs). The research, while examining the existing IFCs within the available releases of the IFC model, recommends a set of extensions to the model to be included in future releases.

The research explores the implementation of these models through the development of an Asset Management Tool prototype integrated application, as a proof-of-concept for implementing the view of generic asset management in a distributed, model-based, integrated systems architectures environment named Jigsaw Distributed System. An interface for representing the hierarchical breakdown of project information was developed and tested. The Interface enables users to relate multiple kinds and sources of information and initiate data exchanges with other desktop software applications, hence demonstrating software interoperability, which in turn reduces dependency on paper-based views of project information.

The resulting contributions to the research have demonstrated that a distributed, model-based approach to systems integration is feasible in the FM domain and that resulting applications hold the promise of exhibiting flexibility and significant integration.
Table of Contents

ABSTRACT .................................................................................................................................................. ii
TABLE OF CONTENTS .............................................................................................................................. iii
LIST OF TABLES ........................................................................................................................................ vi
LIST OF FIGURES ....................................................................................................................................... vii
ACRONYMS .................................................................................................................................................. ix
DEFINITIONS ................................................................................................................................................ x
ACKNOWLEDGMENTS .............................................................................................................................. xii

Chapter 1 Introduction .............................................................................................................................. 1
1.1 STATEMENT OF THE PROBLEM ....................................................................................................... 1
1.2 MOTIVATION ......................................................................................................................................... 3
1.3 RESEARCH HYPOTHESES, OBJECTIVES AND CHALLENGES ..................................................... 3
1.4 SCOPE .................................................................................................................................................. 5
1.5 RESEARCH METHODOLOGY .............................................................................................................. 5
1.5.1 Domain Analysis and Description .................................................................................................... 5
1.5.2 Development of a Conceptual Framework and Model ................................................................. 6
1.5.3 Prototype Development and Implementation .................................................................................. 7
1.6 READER’S GUIDE ............................................................................................................................... 7

Chapter 2 Points of Departure .................................................................................................................... 10
2.1 MAINTENANCE MANAGEMENT – A BACKGROUND ....................................................................... 10
2.1.1 Maintenance Information Management .......................................................................................... 10
2.1.2 Making Decisions in Maintenance Organizations ........................................................................... 11
2.1.3 Classification of Maintenance Operations ....................................................................................... 11
2.1.4 Multimedia Technology in Maintenance Management .................................................................... 11
2.2 FACILITY MANAGEMENT FUNCTIONS .......................................................................................... 12
2.3 PREVIOUS RESEARCH ON CONCEPTUAL FACILITIES MANAGEMENT MODELS ..................... 15
2.3.1 Process and Data Models for Particular Building Systems ............................................................. 15
2.3.2 Process and Data Models for Generic Facilities Management ....................................................... 18
2.4 THE INTERNATIONAL ALLIANCE FOR INTEROPERABILITY .................................................... 25
2.4.1 IFC Model Development Methodology ............................................................................................ 25
2.4.1.1 Process Models ............................................................................................................................ 26
2.4.1.2 Usage Scenarios .......................................................................................................................... 26
2.4.1.3 Object Models ............................................................................................................................ 26
2.4.2 IFC Model Architecture .................................................................................................................. 27
2.4.2.1 Model Requirements .................................................................................................................... 27
2.4.2.2 IFC Model Architecture Decomposition .................................................................................... 27
2.5 TOTAL PROJECT SYSTEMS ................................................................................................................. 28
2.6 MAINTENANCE MANAGEMENT SOFTWARE REVIEW .................................................................... 29
2.6.1 MicroROOFER ................................................................................................................................. 29
2.6.2 BUILDER ........................................................................................................................................... 30
2.6.3 MAXIMO Enterprise ....................................................................................................................... 32
2.6.4 RECAPP .......................................................................................................................................... 33
2.7 ROOFING SYSTEMS AS AN APPLICATION DOMAIN ....................................................................... 35
2.8 DISCUSSION ......................................................................................................................................... 36
5.3.1.1 Jigsaw Asset Management Tool ................................................................. 82
5.3.1.2 Jigsaw Application Objects ........................................................................ 82
5.3.2 Data Server Software Components ................................................................. 84
5.3.2.1 Jigsaw Server Document Object Model ....................................................... 84
5.3.2.2 Jigsaw XML File ......................................................................................... 85
5.3.2.3 Jigsaw BLIS Data Source ........................................................................... 85
5.3.2.4 Jigsaw MicroROOFER Data Source .......................................................... 85
5.3.2.5 Jigsaw Microsoft Project Data Source ......................................................... 85
5.4 Discussion ........................................................................................................... 86

CHAPTER 6 IMPLEMENTATION AND EVALUATION ........................................................................ 87
6.1 Components of User Interface ........................................................................ 88
6.1.1 Tree View ........................................................................................................ 88
6.1.2 Multiple-Document Interface Forms .............................................................. 92
6.2 Functional Requirements of User Interface ...................................................... 92
6.2.1 Browsing ......................................................................................................... 92
6.2.2 Creating and Editing Object Data .................................................................. 92
6.2.2.1 Creating/Editing a Project Object ................................................................. 92
6.2.2.2 Creating/Editing a Product Object ............................................................... 93
6.2.2.3 Creating/Editing an Asset Object ................................................................. 94
6.2.2.4 Creating/Editing a Task Object .................................................................. 94
6.2.3 Importing/Exporting Data from and to Data Servers ...................................... 95
6.3 Evaluations and Testing Scenario ..................................................................... 98
6.3.1 Starting from Scratch ..................................................................................... 98
6.3.2 Creating Association Relationships between Objects .................................... 99
6.3.3 Exporting Data to MicroROOFER Application ............................................ 101
6.3.4 Importing/Exporting Data to Microsoft Project Application ...................... 102
6.4 Discussion ........................................................................................................ 104

CHAPTER 7 CONTRIBUTIONS AND CONCLUSIONS .................................................................... 107
7.1 Summary .......................................................................................................... 107
7.2 Research Contributions and Deliverables ........................................................ 108
7.2.1 Contribution to Facility Management Practice .............................................. 108
7.2.2 Contribution to Development of Domain Models ........................................ 109
7.2.3 Contribution to Development of Integrated Applications ............................ 109
7.2.4 Contribution to Domain Analysis and Description ....................................... 109
7.3 Future Work ...................................................................................................... 109
7.3.1 Facilities Management Domain ..................................................................... 110
7.3.2 Development of Domain Models .................................................................. 110
7.3.3 Development of Integrated Applications ...................................................... 110
7.4 Conclusions ...................................................................................................... 110

REFERENCES ............................................................................................................. 112

APPENDIX A – IDEF6 PROCESS MODELING NOTATION GUIDE ........................................... 118

APPENDIX B - PROPOSED EXTENSIONS TO IFC MODEL .................................................. 121
List of Tables

Table 3.1: MRR options corresponding to values of roofing condition index .................45
Table 3.2: Subjective listing of condition rating categories .............................................45

Table A-1: Data entity descriptions ..................................................................................119
List of Figures

Figure 2.1: Types of maintenance actions (Vanier 1996) ................................................................. 11
Figure 2.2: An overview of the scope of FM activities (Quah 1992) .................................................. 13
Figure 2.3: Functions of planning and control (Yu et al. 2000) ........................................................... 13
Figure 2.4: Identifiable FM Functions (Yu et al. 2000) ..................................................................... 14
Figure 2.5: FM elements (Yu et al. 2000) ....................................................................................... 15
Figure 2.6: Portion of roofing system product model (Vanier 1998) .................................................... 16
Figure 2.7: Architectural design process for precast concrete facades (Karhu 1997) ...................... 17
Figure 2.8: Main decomposition relationship of a facade (Karhu 1997) ............................................ 17
Figure 2.9: Portion of the object model of FMIS (Bos 1995) ............................................................. 18
Figure 2.10: Generate maintenance forecast valuation (Underwood and Alshawi 1999) ............. 19
Figure 2.11: Node A1, prepare maintenance forecast valuation for replacement remedies (Underwood and Alshawi 1999) ................................................................. 19
Figure 2.12: Node A2, prepare maintenance forecast valuation for repair remedies (Underwood and Alshawi 1999) ................................................................. 20
Figure 2.13: EXPRESS-G model for MAINCAST (Underwood and Alshawi 1999) ....................... 21
Figure 2.14: An overview of the FM process model (Svensson 1998) ............................................. 21
Figure 2.15: Node A1, core processes (Svensson 1998) ................................................................. 22
Figure 2.16: Node A2, FM configuration process (Svensson 1998) ................................................ 22
Figure 2.17: Node A3, FM execution process (Svensson 1998) ........................................................ 23
Figure 2.18: Node A34, controlling of FM task (Svensson 1998) ...................................................... 23
Figure 2.19: Node A4, FM control process (Svensson 1998) ............................................................ 23
Figure 2.20: The schematic structure of the KBS model (Svensson 1998) ....................................... 24
Figure 2.21: IFC Model layers ........................................................................................................ 27
Figure 2.22: A screen capture from MicroROOFER application ..................................................... 30
Figure 2.23: A screen capture from BUILDER application ............................................................. 31
Figure 2.24: A screen capture from MAXIMO application ............................................................. 33
Figure 2.25: A screen capture from RECAP application ............................................................... 34
Figure 2.26: Conventional and protected membrane roofing systems (Hedlin 1989) .............. 36
Figure 3.1: General processes involved in maintenance management model .................................. 39
Figure 3.2: Node A, identify assets ................................................................................................ 40
Figure 3.3: Node R, identify performance requirements ............................................................... 42
Figure 3.4: Node P, assess performance ....................................................................................... 44
Figure 3.5: Node P.3, identify distress (anomaly) ........................................................................... 45
Figure 3.6: Node M, plan maintenance ........................................................................................ 47
Figure 3.7: Node M.2, estimate cost of maintenance ................................................................... 48
Figure 3.8: Node O, manage maintenance operations .................................................................... 49
Figure 3.9: Node O.1, request maintenance work order ............................................................. 51
Figure 3.10: Node O.2, plan maintenance ...................................................................................... 52
Figure 3.11: Node O.22, define maintenance activities ................................................................. 52
Figure 3.12: Node O.23, identify precedence relationship ............................................................ 53
Figure 3.13: Node O.3, schedule maintenance activities ............................................................... 54
Figure 3.14: Node O.4, accomplish maintenance workload .......................................................... 55
Figure 3.15: Node O.5, record maintenance .............................................................................. 56
Figure 3.16: Node O.52, report resources consumed ..................................................................... 57
Figure 4.1: Overview of roofing maintenance management data model ........................................ 62
Figure 4.2: IFC Model for identifying roofing system components ............................................. 64
Figure 4.3: IFC Model for identifying performance requirements of roofing systems .............. 65
Figure 4.4: Existing property classes in IFC release 2X model .................................................... 65
Figure 4.5: IFC Model for performance assessment ....................................................................... 66
Figure 4.6: IFC Model for maintenance planning of roofing systems ......................................... 67
Figure 4.7: IFC model for roofing system maintenance operations management ...................... 68
Figure 4.8: UML diagram for the scenario of identifying roofing system components ............ 70
Figure 4.9: UML diagram for identifying performance requirement of roofing systems ............ 71
<table>
<thead>
<tr>
<th>Acronyms</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AEC/FM</td>
<td>Architecture, Engineering, Construction and Facilities Management</td>
</tr>
<tr>
<td>AMT</td>
<td>Asset Management Tool</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>BELCAM</td>
<td>Building Envelope Life Cycle Asset Management</td>
</tr>
<tr>
<td>BLIS</td>
<td>Building Lifecycle Interoperable Software</td>
</tr>
<tr>
<td>CAS</td>
<td>Condition Assessment Survey</td>
</tr>
<tr>
<td>DC</td>
<td>Data Client</td>
</tr>
<tr>
<td>DS</td>
<td>Data Server</td>
</tr>
<tr>
<td>DOM</td>
<td>Document Object Model</td>
</tr>
<tr>
<td>EMS</td>
<td>Engineered Management System</td>
</tr>
<tr>
<td>FCI</td>
<td>Flashing Condition Index</td>
</tr>
<tr>
<td>FM</td>
<td>Facilities Management</td>
</tr>
<tr>
<td>FMC</td>
<td>Facility Management Class</td>
</tr>
<tr>
<td>IAI</td>
<td>International Alliance for Interoperability</td>
</tr>
<tr>
<td>ICI</td>
<td>Insulation Condition Index</td>
</tr>
<tr>
<td>IDEF</td>
<td>Integrated Definition for Function Modeling</td>
</tr>
<tr>
<td>IFC</td>
<td>Industry Foundation Class</td>
</tr>
<tr>
<td>ISO</td>
<td>International Organization for Standardization</td>
</tr>
<tr>
<td>IT</td>
<td>Information Technology</td>
</tr>
<tr>
<td>ITAC</td>
<td>Information Technologies in AEC/FM in Canada</td>
</tr>
<tr>
<td>JDS</td>
<td>Jigsaw Distributed System</td>
</tr>
<tr>
<td>JMT</td>
<td>Jigsaw Modeling Tool</td>
</tr>
<tr>
<td>KBS</td>
<td>Knowledge Based System</td>
</tr>
<tr>
<td>MCI</td>
<td>Membrane Condition Index</td>
</tr>
<tr>
<td>MRR</td>
<td>Maintenance, Repair and Renewal</td>
</tr>
<tr>
<td>NRCC</td>
<td>National Research Council Canada</td>
</tr>
<tr>
<td>PWGSC</td>
<td>Public Works and Government Services Canada</td>
</tr>
<tr>
<td>RCI</td>
<td>Roofing Condition Index</td>
</tr>
<tr>
<td>STEP</td>
<td>Standard for the Exchange of Product Model Data</td>
</tr>
<tr>
<td>TOPS</td>
<td>Total Project Systems</td>
</tr>
<tr>
<td>UI</td>
<td>User Interface</td>
</tr>
<tr>
<td>UML</td>
<td>Unified Modeling Language</td>
</tr>
<tr>
<td>XML</td>
<td>Extensible Markup Language</td>
</tr>
</tbody>
</table>
Definitions

**Asset**: A uniquely identifiable element or group of elements which has a financial value and against which maintenance actions are recorded (IAI 1999).

**Asset Manager**: The person responsible for major maintenance, repair, renewal decisions, as well as the long-term strategic planning (beyond five years) of a corporate asset portfolio (Vanier 2000).

**Asset Register**: A table which records the assets of an organization, within which the value of all assets can be determined (IAI 1999).

**Control**: An entity which influences or determines the process of converting inputs to outputs. In a process model diagram, a control is shown entering the top side of the box (Federal 1993).

**Function**: An activity, action, process, operation, or transformation, which is described by an active verb. A function is shown as a box in a process model diagram (Federal 1983).

**Input**: An entity, which undergoes a process or operation, and is typically transformed. In a process model diagram, it enters the left of the box, and may be any information or material resource (Federal 1983).

**Maintenance**: The combination of all technical and administrative actions intended to retain an item in, or restore it to, a state in which it can perform its required function (Glossary 1984).

**Mechanism**: An entity such as a person or a machine, which perform a process or operations. In a process model diagram, a mechanism is shown entering the bottom side of the box (Federal 1983).

**Model**: A representation in conceptual form of the flow and content of information that is contained within a defined area of interest (IAI 1999).

**Node**: A unique identifier to every function. In a process model diagram, a node number is shown in the bottom right hand corner of the box (Federal 1993).

**Object Model**: A representation of the information content and structure that will be exchanged or shared (IAI 1999).

**Output**: An entity which results from a process or objects which are created by a function. In a process model diagram, an output is shown existing the right side of the box (Federal 1993).

**Performance**: The behavior of a product related to use (ISO-6241 1984).


Performance Values: Upper and lower limits of acceptable solutions to fulfill the performance requirements (ISO-6241 1984).

Planned Maintenance Work: Maintenance work organized and carried out with forethought, control and the use of records to a pre-determined plan (Then 1982).


Property Manager/Facility Manager: The person responsible for day-to-day accommodation issues and the implementation of the strategic plan (Vanier 2000).

Service Life: The actual period of time during which the asset, or any of its components, performs without unforeseen costs of disruption for maintenance and repair (CSA-S478 1995).

Unplanned Maintenance Work: Maintenance work carried out outside planned maintenance and for which there is no regular pattern of inspections (Then 1982).

Usage Scenarios: Textual descriptions of situations that show the use of IFCs to carry out the selected process (IAI 1999).
Acknowledgments

All praise be to Allah, the Lord of the two worlds, the Almighty with whose gracious help it was possible to accomplish this work.

I wish to thank my supervisor, Professor Thomas Froese for his numerous suggestions and input throughout the duration of this research, and his expert guidance and help with the development of the prototype application as part of this project. I wish to thank my co-supervisor Dr. Dana Vanier, for his valuable input during the research, the thorough, yet prompt review of the thesis, and suggestions, which contributed to improving the quality of this manuscript. Also, I wish to thank the supervisory committee members Professors Alan Russell and Barbara Lence for their review of the research, and their comments and inputs. Thanks to Dr. Mahmoud Halfawy for the various useful discussions we had.

Thanks for the inputs provided by the university examiners, Dr. Carlos Ventura and Dr. Jerzy Wojitowicz, and the external examiner Dr. Michael Case for their challenging and constructive reviews.

Thanks to many friends and colleagues at the University of British Columbia for providing a rich international atmosphere during my stay in Vancouver. They include Arezou Pouria, Abdolhamid Ghanbari, Branka Kosovac, Frank Lin, Heli Torres, and many others. Additionally, I wish to thank Dr. Edward Harkness for the fruitful discussions we had and for his guidance and encouragement.

Deep gratitude is extended to my parents and brothers, who have been always an important source of support and encouragement.

I have received financial support for this work from the National Research Council Canada, and Public Works and Government Services Canada, through the Building Envelope Life Cycle Asset Management project, and the National Science and Engineering Research Council of Canada.
Chapter 1
Introduction

Chapter Abstract: This chapter introduces the topic area researched in this dissertation. It provides a statement of the problem, followed by a description of the motivation behind conducting the research. It presents a detailed statement of the hypotheses, research objectives and the intellectual challenges of the research, followed by a description of the scope of the research, then the methodology applied in the dissertation.

This dissertation discusses research in the field of integrated systems for maintenance management. Integration of data and knowledge, through shared project models, among participants in a maintenance project promises increased efficiencies in the facilities management (FM) industry. The research focuses on the development of project models: a combination of product and process views of the product being maintained. It also focuses on a prototype application developed to provide proof-of-concept for data interoperability among users and applications.

1.1 Statement of the Problem

A building can be considered as an asset or an investment that needs to be maintained to ensure its optimal value over its life cycle. Building systems, such as roofing, mechanical, or electrical, usually have a shorter life span than their supporting structures. These services are in constant need of regular maintenance to ensure that they continue to function properly and that they retain their value and good appearance. Maintenance, as per British Standard 3811, may be defined as “the combination of all technical and administrative actions intended to retain an item in, or restore it to, a state in which it can perform its required function” (Glossary 1984).

The construction industry is often described as a fragmented industry. The author argues that the asset management industry shares this characteristic with the construction industry. One aspect of this can be seen in the fact that the asset management industry is witnessing a proliferation of information technology (IT) tools (Vanier 2001), mainly relational databases, each capable of providing standalone solutions to a multitude of problem areas. This proliferation of IT tools is, however, leading to large volumes of loosely-structured data (Kyle et al. 2000; Peters and Meissner 1995) with poor interoperability.

The industry has became aware of data model standards as a way of representing technical and administrative information content, leading to the development of data structures that allow information to be exchanged among various computer applications (Eastman 1999). A requirement to achieve integration is conformance to some degree of standardization of the information representation approaches (Russell and Froese 1997). Efforts in this line of research include those under ISO International Standard 10303,
STEP (Standard for the exchange of product model data, ISO 1994), or the Industry Foundation Classes (IFCs) developed within the International Alliance for Interoperability (IAI) (IAI 1999).

In the facilities management field of research, little is found on developing object-oriented models for building maintenance, especially the area of operations management, as compared to other areas within the facilities management domain, such as modeling technical, spatial, and construction systems. Examples of which are cited in Chapter 2 (Points of Departure) of the dissertation (Vanier 1998; Karhu 1997; Bos 1995; and Svensson 1998).

This dissertation’s challenge in achieving integration was the development and standardization of high level, generic maintenance information models. The approach maintained while developing these maintenance information models was not only focusing on what product is maintained (product model), but also on the procedural context in which the products are maintained (process model). Models combining product and process are referred to as project models (Froese 1995).

In developing the maintenance information models presented in this dissertation, and in keeping with the notion of developing integrated maintenance project models, the author chose to conform to the development methodology set by the IAI. The IAI is a global, non-profit, industry-based consortium for the Architecture, Engineering, Construction and Facilities Management (AEC/FM) industry (IAI 2000). The goal of the IAI is to enable interoperability among industry processes of all different professional domains in AEC/FM projects by allowing the computer application used by all project participants to share and exchange project information. The scope of the IAI is the entire life cycle of the building project, covering strategic planning, design and engineering, construction, and building operation. The IAI was established with the objective of defining, publishing and promoting a specification (the IFCs) which describes building project objects in a neutral computer language that represents information requirements common to all industry processes.

The maintenance project models developed in this dissertation provide a direction for the implementation of IFCs in a distributed model-based application. This application implements the developed process and data models and delivers the capability to import and export (read and write object data) IFC information from, and to, other legacy applications, thus facilitating data exchange and achieving software interoperability. Such capability holds the promise of significantly impacting AEC/FM industries through communicating the critical relationships between project information and greatly reducing the dependency on paper-based views of project information. It is envisaged that the intended user of the developed application would be representative middle management staff members (referred to as operational staff) in a facilities management organization engaged in managing several assets. Their scope of work includes implementing the decisions made at the strategic, tactical and operational planning windows. The beneficiaries of the thesis include building investors, and building occupants due to the more thorough and efficient use of knowledge and data.
1.2 Motivation

Industry concerns in Canada are the driving factors behind the research developed within the dissertation. Reasoning to support this view stems from the fact that in developed countries, the total value of building and works account for approximately two thirds of a nation’s capital stock. This does not only represent wealth accumulated over many years, but it also represents an important factor in the production of new wealth (Lee 1987). Thus the preservation of the value and utility of the stock of building is essential to the economic well-being of the country.

Canada has an approximate value of $5.5 trillion worth of built assets, and the condition of these assets is deteriorating due to age, climate, and change in use (Vanier 2001). It was found that because of lack of adequate funding and appropriate support technologies, certain components of the infrastructure have been neglected and receive only remedial treatments. Consequently, these assets will not last their originally predicted life unless there are major maintenance and rehabilitation investments (Vanier et al. 1997).

Commenting on the state of Canadian government assets, the Office of the Auditor General of Canada stated that, “The consequences of not carrying out adequate building maintenance and repairs are loss of asset value, poor quality of working space, potential health and safety problems, the probability of higher repair costs in the future, and increasing reliance on more costly leased accommodation. Without proper maintenance, facilities will deteriorate to the point that extensive investment is required to restore or replace them. The result, ultimately, will be increased federal costs” (OAG 1994).

A proposal was prepared by the Institute for Research in Construction (IRC), National Research Council Canada (NRCC) and Public Works and Government Services Canada (PWGSC) for a project investigating service life prediction and maintenance management of building envelope components. The project was named “Building Envelope Life Cycle Asset Management” (BELCAM), where roofing systems, specifically low-slope conventional assemblies, were chosen as a building system that is representative of the built-asset maintenance domain. The BELCAM project identifies maintenance management as one of the essential “enabling” technologies to permit asset managers to operate assets efficiently (Vanier and Lacasse 1996).

The research presented in this dissertation is interested in exploring the intersection of two domain areas: IT, which is providing significant improvement to business practices throughout many industries, and FM, an area dealing primarily with sustainability of capital investment, which is gaining attention as a large industry with many opportunities for improved business practices, methodologies and software tools.

1.3 Research Hypotheses, Objectives and Challenges

The main objective of the research is to apply the technology of model-based integrated systems to the domain of facilities maintenance management. Working towards this
objective, a series of principle hypotheses were formulated based on the current state of the practice and industry. These hypotheses are that:

1. For the primary purpose of supporting the development of IT solutions, FM knowledge can be formalized in a manner similar to that of project and construction management knowledge.
2. The development of data standards facilitates the integration of data and knowledge of FM projects. The emerging IFC standards span much of required information, but additional exploration and development is required.
3. Interoperable software that integrates data and knowledge can be developed for FM applications.

With the main objective in mind, the following sub-objectives were formulated to investigate and prove these principle hypotheses:

1. To develop a generalized analysis and description of the problem of facilities maintenance management.
2. To develop a conceptual, integrated, process-oriented framework for a facilities maintenance management system.
3. To develop data models for managing maintenance activities, through comparisons with existing data models for managing construction activities.
4. To develop a prototype integrated, interoperable software application that implements the view of generic asset management in the context of distributed, model-based, integrated AEC/FM systems architecture.

These objectives follow integration technologies that are emerging within the AEC/FM industry. Most work to date has focused on the areas of design and project management. In general, integration approaches within the FM domain can be expected to be similar to those within design and project management. However, relatively little attention has been paid to FM and there are enough differences between the disciplines to create significant research challenges. Examples of the differences between FM and design or construction phase work include the fact that in FM, full knowledge of the product is available (though it may not exist electronically), whereas the overriding issue in design and construction is the creation of the product (both the creation of the "product model" or facility design, and the creation of the physical product), and it is generally assumed that this information is available electronically as it is produced. Further, in design and construction, there is little focus on deciding what work is to be carried out (all of the work necessary to construct the building must be completed, decisions lay in how best to do this). In FM, decisions about what work should be carried out within limited funding availability is a significant factor. Thus, there are enough significant differences between the practice of FM and the practice of facility design and construction that it cannot be assumed that integration solutions for one are directly appropriate for the other, and a research challenge exists to explore the degree to which technological solutions for the two disciplines overlap and to address the differences. Indeed, there appears to be relatively little transfer of IT technology between conventional AEC software and the FM industry at present.

One main intellectual challenge faced in proving the research hypotheses and carrying out the stated research objectives, specifically the development of the generic framework
of maintenance management, was the formalization and representation of FM data from a large and unstructured body of knowledge. This body of knowledge appears to be less well developed than areas such as project and construction management. Another intellectual challenge faced in the development of a maintenance management data model is the fact that there have been some efforts to formalize FM knowledge and practice in models, but these were found to be few and partial in their coverage of the breadth and depth of FM concerns. Further, there appears to be virtually no implementation or assessment of these models for FM applications from the standpoint of data exchange.

1.4 Scope

The following describes the scope of the dissertation:

1. The maintenance management framework developed was generic; limited neither by the complexity of asset types being examined, scale of FM projects, nor scale of FM organizations applying the framework.

2. The maintenance management data model developed was a high-level, conceptual model. If the proposed model were to be adopted by the IAI as extensions to the IFCs, the model would receive additional vetting and validation by committees of domain experts, through implementation by commercial software developers, by end users of software systems, and then the entire cycle would be repeated in future release cycles. These validation mechanisms, beyond the initial prototype, are beyond the scope of this dissertation.

3. The scope of application development was limited to allowing data exchanges and facilitating the development of inventory lists of assets and the creation of maintenance, repair or renewal tasks.

4. The scope of data exchange, facilitated through the implementation of the Asset Management Tool prototype application as a data client component in the Jigsaw Distributed System, was limited to a specific set of data source types (two desktop legacy applications, XML files, IFC files, and remote databases).

1.5 Research Methodology

The research plan set to achieve the objectives of this dissertation consists of three main phases. These phases are defined as follows:

1.5.1 Domain Analysis and Description

The domain analysis and description phase in this research implies:

1. Reviewing the existing state-of-the-art, to achieve a thorough understanding of the area of interest (facilities maintenance management), and to analyze the approaches taken in developing facilities maintenance management information systems.
2. Reviewing some of the available commercial software packages on maintenance management by studying their operating characteristics and functionalities.

3. Conducting interviews and discussions with facility management professionals to solicit knowledge from their experience and feedback. Opportunities to achieve this activity include:
   a. Attending the facilities management domain experts group at the technical summit #28 of the North American Chapter of the International Alliance for Interoperability (IAI) in Seattle, Washington, USA, 16-17 September 1997. While attending this summit, the framework for facilities maintenance management of infrastructure built-assets was conceptualized by the author (along with the research supervisor Professor Thomas Froese, Construction Engineering and Management Group and Kevin Yu, a graduate student from the University of British Columbia). The framework was subsequently developed by the author as part of the research for this dissertation.
   b. Working as a guest worker with the Building Envelope Life Cycle Asset Management (BELCAM) project’s team of researchers in the summer of 1998, at the Institute for Research in Construction, at the National Research Council Canada, Ottawa, Ontario, Canada. The author was also involved in participating in the first regional condition assessment surveys of roofing systems (as one of the planned activities in the BELCAM project) for a number of buildings in the campus of National Research Council Canada, Ottawa Region.
   c. Holding discussion groups with technical staff of Plant Operations, University of British Columbia.
   d. Participating at technical venues related to facilities management such as the 8th International Conference for Durability of Building Materials and Components, Vancouver, Canada, 29 May – 2 June 1999; the 8th International Conference for Computing in Civil and Building Engineering, California, USA, 14-17 August 2000; and the 4th Construction Specialty Conference of the Canadian Society for Civil Engineering, Victoria, Canada, 30 May – 2 June 2001.

1.5.2 Development of a Conceptual Framework and Model

The development phase of the conceptual, integrative, process-oriented framework for the facilities maintenance management system included:

1. Investigating facilities maintenance management functions, and studying activity linkages between construction projects and maintenance operations.
2. Performing a system design process involving the following tasks:
   a. Feasibility study and analysis to reach a more precise scope statement of the system based on investigating and analyzing the current situation and practices in the field of facilities maintenance management, and to state the requirements of the system. This study would aid in the preparation of the preliminary specification and criteria for the proposed developed application.
   b. Data modeling, following a methodology similar to that adopted by the IAI, to represent high-level information within the proposed framework of facilities
maintenance management (IAI 1999). IAI projects follow a standard process-oriented methodology to develop data standards in the form of IFCs, comprised of the following steps (Froese et al. 1998):

- **Process Selection**: includes proposals to develop new industry processes and to define the scope of selected processes.
- **Usage Scenarios**: include descriptions of situations that show the use of the IFCs to carry out the selected process.
- **Process Definitions**: include a description of the steps required to carry out the selected process, their logical sequence, and their information requirements in a formal process modeling language. IDEF0 graphical notation (Federal 1993) was used to represent visually the selected processes and the steps within.
- **Information Analysis**: includes using the high level information input and output requirements of each process to define modifications and additions to the IFCs in terms of detailed data elements.
- **Information Modeling and Validation**: transfers the results of the information analysis into the formal modeling language EXPRESS-G (ISO 10303-11 1994).

### 1.5.3 Prototype Development and Implementation

Prototype development and implementation involves definition of architectures and user interfaces for the integrated system. The prototype was developed using Microsoft Visual Basic. It demonstrates an application to low-slope conventional roofing system in accordance with the objectives set by the BELCAM project.

### 1.6 Reader's Guide

This dissertation is organized as follows:

- **Chapter 2: Points of Departure**

This chapter presents a background to the research work through providing a description of the relevant knowledge areas in maintenance management and presenting the state-of-the-art in research of conceptual facilities maintenance management models. The chapter then presents the methodology and architecture for developing data standards in the form of IFCs. An overview of Total Project Systems as a collection of applications, within which the prototype application described in Chapter 6 is part of, is described. The chapter presents a review of some of the available maintenance management software. Finally, the chapter describes low-slope roofing systems as the application domain for the developed prototype. The chapter constitutes a domain analysis and description, providing the foundation for the research work described in the succeeding chapters.
• Chapter 3: Maintenance Management Process Model

This chapter provides a description of the proposed asset maintenance management framework. The main challenge faced in the development of the framework is the formalization and representation of FM data from a large and unstructured body of knowledge. Presentation of the framework in this chapter is in the form of a process model. IDEF0 notation was used to visually represent the selected management processes and their supporting functions with their logical sequence and information requirements. A discussion as to the advantages and disadvantages of using the process modeling technology to describe the proposed framework is included at the end of the chapter.

• Chapter 4: Maintenance Management Data Model

This chapter presents the development of an object model for maintenance management of roofing systems as a case study to demonstrate the applicability of the proposed generic framework presented in Chapter 3, to a type of a building product that has been treated as an asset. A challenge faced in the development of the maintenance management data model is the fact that there has been some, but relatively little, FM input to the IFC standards to date, and virtually no implementation or assessment of these standards for FM applications. The model was developed to be consistent with IAI standards. The model builds upon the existing IFCs in Releases 2.0 and 2X of the IFC model to define object requirements and relationships for the exchange and sharing of maintenance information between applications. The chapter proposes several extensions to the IFCs, including the representation of functional requirements, condition assessment of objects, inspection and maintenance tasks, and libraries of non-project specific information.

• Chapter 5: Distributed Systems and Architecture

This chapter describes the system architecture of the prototype Asset Management Tool (AMT) application, a distributed model-based integrated system. It presents the set of inter-connecting components forming a typical or reference architecture for integrated distributed systems. It describes the Jigsaw Distributed System version 0.6, as the implementation environment of the reference architecture, which facilitates wide ranges of data exchanges and software interoperability.

• Chapter 6: Implementation and Evaluation

This chapter describes the development of a generic Asset Management Tool prototype application that can initiate data exchanges with a number of desktop applications (MicroROOFER and Microsoft Project), thus demonstrating software interoperability in the FM domain. The Asset Management Tool prototype application served as a validation for the structure of the developed process and data models. The chapter starts by describing the components of the user interface and the specific set of actions through which the user interacts with the prototype application. The chapter then presents the set of functional requirements and the implementation
environment of the user interface. Finally, the chapter presents an evaluation and testing scenario for the prototype application.

- Chapter 7: Contributions and Conclusions

This chapter concludes this dissertation. It summarizes the research findings and development areas resulting from pursuing and achieving the objectives of the research. The chapter lists the research contributions and the deliverables made to the appropriate societies and the FM community at large. These contributions include contributions made to FM practice, development of FM domain models, development of integrated applications, and FM domain analysis and description. Finally, the chapter discusses future research topics that could follow from this dissertation, and that hold significant potential to improve the FM industry.
Chapter 2
Points of Departure

Chapter Abstract: This chapter presents a background to the research work through providing a description of the relevant knowledge areas in maintenance management, and presenting the state-of-the-art in research of conceptual facilities maintenance management models. The chapter then presents the methodology and architecture for developing data standards in the form of Industry Foundation Classes. An overview of TOPS as a collection of applications, within which the Asset Management Tool prototype application described in Chapter 6 is part, is described. The chapter presents a review of some of the available maintenance management software. Finally, an introduction to low-slope roofing systems as the application domain for the developed prototype is presented. The chapter constitutes a domain analysis and description, providing the foundation for the research work described in the succeeding chapters.

2.1 Maintenance Management – a background

2.1.1 Maintenance Information Management

Mottonen and Niskala (1992) reported on the categories of information needed in maintenance management from the findings of research aimed at creating an information system for housing maintenance and management in Finland. These categories include:

1. Information on the building and the whole property. This information is created at the planning, design and construction stages of the project. In the case of new buildings, it is practical to transfer these data directly from the information system maintained by the planner to the system that would be employed in facilities management. This includes information on the site and buildings, spatial system (room, floor, etc.) and structural and technical systems.

2. Information on the operation and maintenance of the technical systems such as the estimated service life of equipment. This information is obtainable from designers, manufacturers and the literature in general. Some of the information is created or supplemented in the course of maintenance such as information concerned with the operation and condition of devices and structural elements, user experiences, or operating and maintenance costs.

3. Information on the maintenance organization, staff, goals and principles of the organization, maintenance routines and monitoring procedures.
2.1.2 Making Decisions in Maintenance Organizations

Having all the information needed in hand, maintenance management decisions are taken at three levels as reported by Aikivuori (1992); and Gordon and Shore (1998). It can be seen that the need for information is different at different levels. These levels are as follows:

1. Strategic level: involves the long term planning of facilities, usually beyond a five-year horizon, where decisions are concerned with the long term goals of the organization, their changes and the resources available for achieving them.
2. Tactical (Managerial) level: involves the tactical positioning of the organization in the two to five year frame, where decisions are concerned with the acquisition of resources and their effective and productive use.
3. Operational level: concerns itself with the implementation of the strategic and tactical level in day-to-day operations, normally within the current budget year.

2.1.3 Classification of Maintenance Operations

Vanier and Lacasse (1996) reported on the various types of maintenance actions, as shown in Figure 2.1 below. Preventive maintenance inspection carried out annually or cyclically is termed condition-based inspection; whereas condition-independent inspection, for example, is undertaken in accordance with manufacturer's recommendations or in relation to known deterioration profiles. The inspection may be initiated by a reported failure (response) or by the two preventative maintenance inspection types described earlier. These inspections will necessitate carrying out either normal or emergency action. Emergency maintenance occurs in the event that inaction would create serious safety or health problems. Programmed (or phased) corrective maintenance is needed for failures requiring extensive repairs or where budgets do not permit immediate remedial action.

![Figure 2.1: Types of maintenance actions (Vanier 1996)](image)

2.1.4 Multimedia Technology in Maintenance Management

Usually, the as-built drawings, specifications, and specialty equipment product literature are the only documents that the maintenance organization receives after the completion of the construction stage. However, these documents are not exactly what the maintenance organization needs to carry out maintenance operations. Mottonen and Niskala (1992) indicated that the lack of simple and illustrative maintenance documents is probably one
important reason for the poor maintenance of buildings. They added that computer-aided building design and the development of product models make it easy to use the potential of hypermedia in documentation. Hyperdocuments are interlinked pieces of information that contain text, diagrams, images and possibly sound, video, animation and computer programs. A building with its spatial, structural and technical systems can be documented by numerical information (volume and area), text and scanned drawings and images. The operation principles, location plans and coverage area of technical information can be visualized by hypermedia.

Without focusing directly on the application of hypermedia in maintenance engineering, Turk and Vanier (1995) indicated that the World Wide Web offers capabilities to:

1. Encompass a number of searching and indexing tools through which the user may find a set of technical information that apply to a given problem.
2. Accommodate interfaces to thesauri data. Thesauri are a technique that can easily assist searches by providing controlled vocabulary for both the user and the writer of the technical information.
3. Accommodate annotation systems that allow the users to add comments and note to the technical information they use or study.

Kyle et al. (2000) reported on the development of a prototype application, developed within the BELCAM project, named the BELCAM Visualizer. The concept behind the development of Visualizer is the ability to display non-visual data for large asset inventories through color-coding. The main objective is to treat a multi-objective optimization process for maintenance activities of roofing systems. This process involves the probabilistic predictions of system performance, the assessment of risk and cost, and the ability to visualize various “what-if” scenarios, i.e. change of inventory condition or other properties over time with varying maintenance strategies. Data required for Visualizer to function include: visual representation (map) of the inventory, stored data about the inventory, external data such as current maintenance costs and user input.

2.2 Facility Management Functions

Quah (1992) presented an overview of the scope of FM activities. Figure 2.2 illustrates the realm of FM work can be grouped under four main areas of management: financial, space, behavioral and operational management. Operational management aspect involves maintenance and modernization of building enclosures, building services, building environment and building grounds. Quah (1992) indicated that differential emphasis is expected to be placed in these four areas of management and would be largely dependent on building type and projected life. However, the operational management aspect (maintenance and/or modernization) would normally receive the highest level of attention as it directly contributes to efficient functioning of the building.
Figure 2.2: An overview of the scope of FM activities (Quah 1992)

A FM function hierarchy was identified by the North American Facilities Management Domain Committee of the IAI as a guideline for developing IAI-FM projects. In this hierarchy, FM functions and their processes are considered to have two main aspects: a general management planning and control function as illustrated in Figure 2.3, and specific or identifiable FM functions.

Figure 2.3: Functions of planning and control (Yu et al. 2000)

The identifiable FM functions as illustrated in Figure 2.4, are classified into three categories. These categories are maintenance and operation management, property management, and services. Further, Figure 2.4 also illustrates that maintenance and operation category is classified into three interrelated functional areas. These areas are monitoring and tracking, maintenance, alteration and repairing, and space management.
The figure also illustrates the set of sub-functions within each of the three functional areas (Yu et al. 2000).

Moreover, the North American Facilities Management Domain Committee of the IAI indicated that FM functions are carried out with respect to specific elements within a facility as illustrated in Figure 2.5. Two classifications of elements were identified. These include building system elements and non-building system elements (Yu et al. 2000).
2.3 Previous Research on Conceptual Facilities Management Models

A review of literature on conceptual models for the facilities management industry indicates two streams of work which share the objective of providing better management of large volumes of loosely-structured data produced by all kinds of independently-operating facility applications. These streams are: process and data models for particular building systems that correspond to specific asset types, and process and data models for generic facilities management that correspond to non-specific asset type.

2.3.1 Process and Data Models for Particular Building Systems

Examples of data models for particular building systems include work by Vanier (1998) to develop a STEP-based roofing system product model. The objective behind this model
development centers on mapping input data from MicroROOFER (Bailey et al. 1990), a relational database-Engineered Management System (EMS), to an object-oriented model (as illustrated in the portion of the model shown in Figure 2.6). A goal is to contribute to the process of standardizing the format of data collected from a number of roofing surveys to take place in North America (Vanier and Lacasse 1996). The model addressed the development of new entities and attributes that were not available within MicroROOFER, but were needed to satisfy the objectives of the BELCAM project.

Karhu (1997) developed process and product models for precast concrete facades, aiming at enhancing the exchange of data between architects, structural engineers and precast element manufacturers. Central data structures in this conceptual schema defined how a precast concrete facade consists of precast concrete elements. An element, then, consists of structural layers, which may be an outer wall panel or an insulation layer. Structural layers and openings have edges. A precast facade also has element joints and element divisions.

The process model for designing a precast concrete facade is illustrated in IDEF0 notation in Figure 2.7. The process is broken down into the following sequential functions:

1. Define shape of facades (A1).
2. Design openings and surfaces (A2).
3. Design tiling (A3).
4. Define element division and joints (A4).
5. Define properties of structural layers (A5).
6. Design edge shapes (A6).

Figure 2.8 provides the EXPRESS-G diagram for the data model that illustrates the main decomposition relationships of a facade.
Figure 2.7: Architectural design process for precast concrete facades (Karhu 1997)

Figure 2.8: Main decomposition relationship of a facade (Karhu 1997)
2.3.2 Process and Data Models for Generic Facilities Management

Bos’s work (1994) focused on developing an object-oriented facility management information system (FMIS) with the objective of improving the quality of the working environment and prolonging the building’s life cycle. The approach adopted in developing the model treated the structure of the building as the backbone of the model. Facility management functions such as space utilization, maintenance management, and cost monitoring were applied to the building model. Items such as furniture, appliances and machinery were modeled as objects, while facility services were modeled as functions of these objects. Figure 2.9 displays portion of the object model of FMIS.

![Object Model of FMIS](image)

Figure 2.9: Portion of the object model of FMIS (Bos 1995)

Underwood and Alshawi (1999) proposed scope-specific process and data models for an integrated application, “MAINCAST”, as part of an integrated construction environment prototype, “SPACE”, developed at the University of Salford. The objective behind developing MAINCAST is to forecast building element maintenance during the operational years of a project’s life cycle. In MAINCAST, determining a maintenance forecast requires information on the project’s individual building elements and components, which, when they deteriorate, develop defects that need to be rectified. A maintenance forecast valuation for a project is collated from annual maintenance forecast valuations over the project’s design life.

Figure 2.10 illustrates in IDEF0 notation that the process of generating a maintenance forecast valuation consists of three sequential processes as listed below:

3. Produce total maintenance forecast valuation (A3).
The process "Prepare Maintenance Forecast Valuation for Replacement Remedies" (A1) is broken down into three functions. These functions are illustrated in Figure 2.11, and listed below as follows:

1. Identify replacement remedies applicable to the building design (A11).
2. Determine years when each replacement remedy to be carried out (A12).

Figure 2.11: Node A1, prepare maintenance forecast valuation for replacement remedies (Underwood and Alshawi 1999)
The process "Prepare Maintenance Forecast Valuation for Repair Remedies" (A2) is broken down into three functions. The functions are illustrated in Figure 2.12, and listed below as follows:

1. Identify repair remedies applicable to the building design (A21).
2. Determine years when each repair remedy to be carried out (A22).

Figure 2.12: Node A2, prepare maintenance forecast valuation for repair remedies (Underwood and Alshawi 1999)

Figure 2.13 displays the EXPRESS-G model for MAINCAST. The figure illustrates the main entities in the model, the relationships between the entities and the cardinality specification.
Svensson (1998) proposed a FM process model, aiming at representing FM functions on a conceptual level from an organizational prospective. These functions are initiation/identification, planning/design, execution/implementation, and control/appraisal. The process model consists of four processes as illustrated in Figure 2.14 and described below:
1. Core process (A1): this process is divided into three decision and management levels. These levels are strategic (handling visions and goals), tactical (handling the strategy and planning of the business), and operational (producing the results). This process is illustrated in Figure 2.15.

![Figure 2.15: Node A1, core processes (Svensson 1998)](image)

2. FM configuration process (A2): this process is divided into three functions. It includes the planning stage of the configuration process (concerned with FM issues on the corporate level), the execution stage (brings the policy of the organization into a workable FM solution), and the control stages (evaluates the solution against the FM strategy). This process is illustrated in Figure 2.16.

![Figure 2.16: Node A2, FM configuration process (Svensson 1998)](image)

3. FM execution process (A3): this process contains four functions as illustrated in Figure 2.17. It includes initiation of FM task (to define the objective of the specific task), planning of FM task (preparing activities concerning the specific task), execution of FM task, and controlling of FM task (to ensure that the work corresponds to its specification in the plan). Controlling of FM task is further detailed into three functions as illustrated in Figure 2.18.

![Figure 2.17: Node A3, FM execution process](image)
4. FM Control process (A4): this process includes surveying time and cost (obtaining cost and time feedback from the execution process of the process model), analyzing performance and report, and predicting time and cost (for budget revision purposes). This process is illustrated in Figure 2.19.

While the process model by Svensson (1998) focused on representing FM functions, it did not address operational strategies or implementation procedures as how would these functions be carried out.
Svensson (1998) then proposed a Knowledge Based System (KBS) model for facilities management. The KBS model was structured in five parts as follows and as illustrated in Figure 2.20:

1. **KBS Core**: this part has association with the three catalogue parts (2, 3 and 4). It manages information on systems, spaces, and construction parts. Some of the entities in the core part include Building, System, Technical System, Spatial System and Construction Part.
2. **Catalogue of technical systems**: this part catalogues different types of technical systems in a building. Some of the entities in this part include Earthworks, HVAC and Sanitation, Electrical and Transport.
3. **Catalogue of spatial systems**: this part catalogues different types of spaces in a building. Some of the entities in this part include Section, Storey space, Block, Room, Part of room and Area.
4. **Catalogue of construction parts**: this part catalogues different types of construction parts in a building. Some of the entities in this part include Frame element, Plate, Lining and covering, and Opening and connection.
5. **Work section sub model**: this part manages information on the physical parts of a building and the result of a work-section process where resources are put together. Some of the entities in the part include Soil reinforcement, Excavation or filling, Pile foundation, Trench, Pavement, Method, and labor.

![Figure 2.20: The schematic structure of the KBS model (Svensson 1998)](image)

Yu et al. (2000) proposed the development of Facilities Management Classes (FMC) as a similar effort in advance of, and in extension to, the IFCs for FM domain. The intended FMCs would have similar goal to the IFCs in terms of allowing multiple applications to share and exchange project information. Some of Yu’s reasoning behind introducing FMCs can be summarized as follows:

1. While the IFCs aim at representing all the common information requirements of all industry processes across all AEC/FM domains, such a broad scope has not, so far, provided enough facilities management (FM) application models to support Computer Integrated Facilities Management (CIFM).
2. The IFCs may not address as much specific detail of various FM functionality as is needed to adequately support CIFM, due to the emphasis on inter-process interoperability.

3. From an implementation perspective of FM application, some of the important FM specific information may not be modeled in the IFCs directly and explicitly, since the IFCs are not developed exclusively for FM.

4. The IFC objects contain much detail that is not needed by FM applications.

5. The IFCs represent all AEC/FM information combined together in a set of models of a much larger scope of what FM applications actually needs.

The FM Domain Committee within the United Kingdom (UK) Chapter is currently active on developing a set of IFC object models that support a number of maintenance processes (Wix et al. 1999).

2.4 The International Alliance for Interoperability

The IAI is a global, non-profit, industry-based consortium for the AEC/FM industry. The goal of the IAI is to enable interoperability among industry processes of all different professional domains in AEC/FM projects by allowing the computer application used by all project participants to share and exchange project information (Froese et al. 1998). The scope of the IAI is the entire life cycle of the building project, covering strategic planning, design and engineering, construction, and building operation. The IAI was established with the objective of defining, publishing and promoting a specification (the IFCs) which describes building project objects in a neutral computer language that represents information requirements common to all industry processes.

Since the establishment of the IAI in 1996, four IFC releases have been published: IFC release 1.0 in January 1997, IFC release 1.5 in November 1997, IFC release 1.5.1 in August 1998, IFC release 2.0 in March 1999; Release 2X in October 2000. The IAI exhibits great professional diversity among its members, who come from AEC/FM industry firms, software vendors, research institutes and universities, professional organization, and government agencies. The IAI is organized into nine regional chapters worldwide, overseen by an International Co-ordination Council. Current IAI chapters include North America, German-speaking countries, United Kingdom, France, Singapore, Nordic countries, Japan, Korea, and Austral-Asian, with more than 600 members from over 20 countries. A series of domain committees, which can work on a number of projects, are established within each of the chapters, based on the interests and initiatives of the members. Each domain committee represents a specialized industry sector, which will then provide information requirements for the IFCs. Current classification of domain committees to date include architecture, building services, civil engineering, construction management, codes and standards, estimating, facilities management, project management, structural engineering, and cross domain projects.

The author participated in recent IAI/FM domain activities, including presenting the framework for a maintenance management model of critical built-assets, as addressed later in the thesis. Supporting that activity, the IAI has developed some of this framework’s processes within the current IFCs. Other processes that have not yet been
developed—though their potential has been recognized within the IAI—include the modeling of performance requirements, recording of asset condition, and life cycle costing as a decision making criteria for maintenance, repair and renewal of critical assets.

2.4.1 IFC Model Development Methodology

The maintenance management data model in this dissertation was developed following a methodology similar to that used within the IAI (IAI 1999). The methodology began with developing initial process models and usage scenarios (discussed in the following sections), which led to defining data model extensions to the existing IFCs.

2.4.1.1 Process Models

A process model describes the activities that exist within a business process. It defines those tasks that need to be undertaken, and illustrates how and what information needs to be communicated between tasks (Federal 1993). The IDEF0 notation (Integration Definition for Function Modeling) is the IAI-preferred notation for the creation of graphical process models for IFC specification projects; the IDEF0 notation was selected within the IAI because of its formal documentation support and extensive software support (IAI 1999). In this representation, boxes represent tasks while arrows from the left, right, top and bottom represent inputs, outputs, controls, and mechanisms, respectively. Appendix A provides a description of the process modeling methodology.

2.4.1.2 Usage Scenarios

Usage scenarios, as defined by the IAI, are textual descriptions of situations that show the use of IFCs to carry out the selected process, i.e. developing a roofing maintenance management data model. They relate together the scope definition, the process model and the object classification. In usage scenarios, a set of assertions is made. The objective of developing assertions is to help identify: classes, relationships, cardinality of relationships, and attributes (IAI 1999). These assertions then can be modeled and implemented.

2.4.1.3 Object Models

An object model, according to the IAI, is a representation of the information content and structure that will be exchanged or shared (IAI 1999). Object models are used to analyze the use and design of IFCs. The IAI encourages the establishment of new modeling projects among domain committees, as can be seen from the various releases of IFCs to reflect the expansion in scope (IAI 2000). EXPRESS-G is the preferred graphical notation of object modeling. In this notation, boxes represent classes, corresponding to real world items of interest, solid lines represent relationship between classes, labels on
solid lines represent role names, and heavy lines represent subtype inheritance relationships.

2.4.2 IFC Model Architecture

2.4.2.1 Model Requirements

The IFC model architecture is developed based on the following requirements (IAI 1999):

1. Provide a modular structure to the model.
2. Provide a framework for sharing information between disciplines within the AEC/FM industry.
3. Ease the continued maintenance and development of the model.
4. Enable information modelers to reuse model components.
5. Enable software authors to reuse software components.
6. Facilitate the provision of upward computability between model versions.

2.4.2.2 IFC Model Architecture Decomposition

The architecture of the IFC model operates on four layers, following a "ladder principle", i.e. at any layer, a class may reference a class at the same or lower layers but may not reference a class from a higher layer (IAI 1999). Figure 2.21 illustrates the architecture of the IFC model.

![IFC Model layers](image-url)
1. Resource Layer: provides Resource classes used by classes in the higher levels. Resources can be characterized as general purpose or low-level concepts or objects that do not rely on any other classes in the model for their existence. Resource classes may only reference or use other resources.

2. Core Layer: provides a Core project model. This Core contains the Kernel and several Core Extensions. Core classes may reference other Core classes (subject to limitations listed in 3), and may reference classes within the Resource layer without limitations. Core Classes may not reference or use classes within the Interoperability or Domain/Application Layer. Within the Core layer the "Ladder Principle" also applies. Kernel classes can be referenced or used by classes in the Core Extensions but the reverse is not allowed.

3. Interoperability Layer: provides a set of modules defining concepts or objects common across multiple application types. Interoperability layer classes can reference classes in the Core or Resource layers, but not in the Domain/Application layer.

4. Domain/Application Layer: provides modules tailored for specific AEC industry domains or application types. Domain/Application layer classes may reference any class in the Interoperability, Core, and Resource layer.

2.5 Total Project Systems

The maintenance project models developed in this dissertation provide a direction for the implementation of IFCs in a distributed model-based maintenance management system. The prototype application has been implemented within a new, integrated system named "Total Project Systems" for AEC/FM (Froese et al. 1997, Froese and Rankin 1998, Froese 1999). Total Project Systems (TOPS) have the objective of integrating construction and facilities management applications through implementing the developed industry standards, the IFCs, hence sharing and exchanging information among applications and/or software. Within Total Project Systems, each application has its own data models, thus capturing related information to the specific application.

The term "total-project systems" was adopted to mean a class of construction and facilities management computer systems that are defined by the following characteristics:

1. Comprehensiveness: they include a suite of applications that support a broad range of construction and facilities management functions.
2. Integration: all applications contribute to and draw from a shared pool of project information.
3. Flexibility: they operate in a highly modular, open, flexible, and distributed framework rather than in a restrictive and prescriptive manner.

Through implementing these data models in an object-oriented system and delivering the content of information over the Internet and/or intranets using Extensible Markup Language (XML) datasets (aecXML 2000), information from each application becomes accessible to other applications within Total Project Systems. Tagging XML data allows a more defined and accurate way to search for particular data. XML enabled documents use semantic markup that identifies data elements according to what they are, rather than
how they should appear. As a result, many applications can make use of the information in XML documents.

2.6 Maintenance Management Software Review

This section objectively reviews the general capabilities of four commercially available applications that are known within the FM industry. These four applications while encompassing wide selection of capabilities, represent a typical selection of information technology tools and techniques used in strategic asset management. The objective of this review is to investigate their development in the context of an overall framework, study the operational characteristics, functionalities and to assess the capability of software interoperability of a representative sample of information technology tools known within the FM industry. The review excludes comments on the user interface through which the user interacts with the application. All products evaluated in this section represent the most recent major implementation of the software at the time of the evaluation.

2.6.1 MicroROOFER

MicroROOFER, Version 1.3, from the US Army Construction Engineering Research Laboratories, is an application developed to provide capabilities for inventory collection, condition assessment information collection and maintenance/repair/renewal analysis of single-ply and built-up roofing systems. The inventory capability within the software allows storage and retrieval of general information on the building, and detailed information on the assembly of components that makes up a roof section within the building.

The condition assessment capability within the software allows storage and retrieval of information, as obtained from visual inspections, on the type, severity and quantity of distresses found in the membrane, flashing and insulation materials of a roof section. The fields where these data are recorded contain pre-defined choices from which the user may make the appropriate selection. Based on the collected condition assessment information from visual inspections, MicroROOFER uses three separate condition indexes to provide a quantitative measure of the component’s ability to perform its function. These indexes are the Membrane Condition Index (MCI), Flashing Condition Index (FCI) and Insulation Condition Index (ICI). Index values are determined through deducting values from a scale of 100, taking into account multiple types of distresses and severity levels found in a roof section. These deduct values are determined from a series of performance charts collated from empirical data. The MCI, FCI, and ICI are combined to determine the Roofing Condition Index (RCI), which serves as a measure of the condition of a roof section, and to indicate the level of maintenance/repair/renewal needed. The RCI is based on a numerical value from 0 to 100, with 100 representing excellent condition.

The maintenance/repair/renewal module of MicroROOFER provides a basic estimate for costs of repair and replacement of roof sections. It also provides deterministic predictions as to the specific year that a roof section would have to be replaced if no repairs were
carried out, and the number of additional service life years if repairs were to take place. MicroROOFER is a standalone application (relational database shell). It has no links with other applications. Figure 2.22 is a screen capture from MicroROOFER application illustrating its inventory capability.

![MicroROOFER Application Screenshot](image)

**Figure 2.22: A screen capture from MicroROOFER application**

### 2.6.2 BUILDER

BUILDER, Version 1.1, also developed by the US Army Construction Engineering Research Laboratories, provides capabilities for inventory collection, condition assessment information collection on buildings and maintenance/repair analysis.

The inventory capability within the software allows storage and retrieval of general information on buildings, where each building is divided into twelve systems, including: roofing, site, specialties, structural, fire suppression, HVAC, interior construction, exterior construction, plumbing, conveying, electrical and exterior closure. Each of the twelve systems is divided into a number of components, and each component is further divided into an appropriate number of sections.

Based on the collected condition assessment information, BUILDER uses the Building Condition Index (BCI), which is derived from the System Condition Index (SCI), which is in turn derived from the Building Component Condition Index (BCCI), to provide a
qualitative measure of the building, system and component’s ability to perform its function. It involves inspecting each component of the above-listed systems visually, evaluating it against a set of pre-defined rating criteria, and selecting an appropriate rating. The condition rating procedure is a less accurate than that of MicroROOFER, but it presents a faster method for performing a condition survey.

The rating approach is based on employing three broad rating categories “Red”, “Amber” and “Green”, where “Red” implies serious problems, and that major maintenance and/or repair is need; “Amber” serves to caution that while things are generally adequate, maintenance and/or repair would make economic sense; “Green” implies that things are fine, although minor maintenance and/or repair may be needed. While the inspector determines which rating category to classify a component in, each rating category is further subdivided into three sub categories, donated as high (+), low (-), and middle for further refinement. The Condition Survey Distress Manual for Assessing Building Condition of BUILDER (Uzarski 1999) has indicated that such an imprecise approach to condition rating may lead to more uncertainty when using this information for maintenance, repair, planning and budgeting modules of the software. BUILDER is a standalone application (relational database shell). It has no link with other applications. Figure 2.23 is a screen capture from BUILDER application illustrating its inventory capability.

![Figure 2.23: A screen capture from BUILDER application](image-url)
2.6.3 MAXIMO Enterprise

MAXIMO Enterprise (MAXIMO 2001), version 4.03, developed by MRO Software, Inc., provides capabilities for inventory collection, condition monitoring, maintenance planning and scheduling, and procurement of machinery and components in plant facilities.

The inventory capability within the software allows storage and retrieval of general information on equipment assets. MAXIMO allows the specification of inspection techniques to be followed such as visual inspection or destructive testing through its Job Plan function. Its ability to use failure and problem codes that can be tied directly to work orders allows the recording of defects found in spaces or elements within buildings. Inspection results can be recorded on work orders. There is limited capability in the condition-monitoring module to determine the existing condition of an asset relative to pre-set performance requirements. However, it allows the user to generate a preventative maintenance plan and record failures and problems with components in order to generate a work order to rectify the condition of the asset.

The job plan functions of MAXIMO allow the estimating of the cost of labor, equipment and material needed to perform a maintenance/repair/renewal action. There does not appear to be capability of MAXIMO to specify the probability of asset failure, the consequences of asset failure, or the remaining service life if no maintenance/repair/renewal action was carried out.

The reporting capabilities within MAXIMO allow the set up and printing of maintenance work requests, including the identification of maintenance/repair/renewal backlog awaiting completion. While the reporting capability of MAXIMO is flexible, its usefulness is dependent on work order coding and use of status flags. Using MAXIMO’s job plans, the user can specify various attributes related to the work being requested. These attributes include specifying work location, labor needed, material needed, equipment needed, and desired completion date. MAXIMO is a client-server application (relational database shell). It allows the capability to link CAD and text files to inventory items. Figure 2.24 is a screen capture from MAXIMO application illustrating its work order tracking capability.
2.6.4 RECAPP

RECAPP, Version 2001.0.0, developed by Physical Planning Technologies Inc., is an application developed to provide capabilities for inventory collection, maintenance work order reporting and budgeting. The user can install any one or many of the software modules, including the Report Manager for allowing the user to chart graphics for various data; Administration Module for controlling the overall look and feel of the dataset, the objects, and the data fields; Budget Manager for displaying financial projections out to 25 years in the future; Event Manager for reporting outstanding maintenance actions; or Scenario Builder for creating “what if’s” scenarios regarding future budgets.

RECAPP permits the user to add any number and type of assets to its database. An object-oriented approach is used to display the asset base at a number of levels of abstractions, namely: portfolio, buildings, sectors, technical components and events. The user can customize a number of fields to express the attributes of the various levels of asset entities; for example, construction year, gross area, replacement value and building type for the building asset; replacement schedules for the technical components; and the estimated cost and time schedule, estimated cost and difficulty factor for the maintenance events. The data structure allows the user to enter a maintenance event for any level of...
abstraction, that is, repair strategies can be associated with portfolio, buildings, sectors
and technical components.

The software does not claim the capability to record performance requirements through
specific sets of metrics. Events can be created to schedule inspections, record their
attributes, and hence reach a condition statement on the components being inspected. The
user can also prioritize maintenance projects based on condition and budgetary
constraints. The application allows the user to select which maintenance project will be
approved for the appropriate implementation. RECAPP is a standalone application
(relational database with object-oriented classes). It allows the capability to link CAD
and text files to inventory items at any level of abstraction (i.e. portfolio, buildings,
sectors and technical components). Figure 2.25 is a screen capture from RECAPP
application illustrating its budgeting capability.

![Figure 2.25: A screen capture from RECAPP application](image-url)
2.7 Roofing Systems as an Application Domain

The BELCAM project has chosen roofing systems as a building system that is representative of built-asset maintenance domain. The project is investigating service life prediction and maintenance management of building envelope components. The BELCAM project, headed by the NRCC and PWGSC, identifies maintenance management as one of the essential "enabling" technologies to permit asset managers to operate assets efficiently. Some of the reasons justifying the choice of roofing systems as an application domain, were stated in the initial project proposal (Vanier and Lacasse 1996) as follows:

1. Roofing repairs are expensive and form a large portion of maintenance budgets.
2. There is a considerable literature dealing with roofing durability.
3. Roofing systems constitute a well-defined domain with some well-known links to other subsystems.

A roofing system as an assembly of interacting roof components is designed to weatherproof and, normally, to insulate a building's top surface. Various roofing systems are available, fulfilling a wide range of functions including energy conservation, acoustical and thermal insulation and water, fire and wind resistance. The thesis will limit the scope to low-slope conventional roofing assemblies. A conventional membrane roofing system has the following different, yet integrated components (Smith 1994):

1. Roof deck and the supporting structure.
2. Vapor and/or air barrier.
3. Thermal insulation.
4. Roofing membrane.
5. Roof flashing.
6. Top cover (ballast, though not always included).

In conventional roofing assemblies, the insulation material is set below the roofing membrane, thus protecting the insulation against wetting and mechanical damage; reducing the dead load that must be supported by the structural system due to dispensing of ballast material; and allowing easy inspection of the roofing membrane. The roofing system, as any other component, is expected to undertake a maintenance action to ensure its continual operation. Figure 2.26 illustrates the basic components of the conventional membrane system in contrast to the protected membrane system.
2.8 Discussion

The chapter provided a description and analysis of knowledge areas in the FM and IT domains that are relevant to the objectives of the research. The following statements constitute points-of-departure for carrying out the research.

The research focuses on the intersection of two topic areas: IT, which is providing significant improvement to business practices throughout many industries, and FM, which is an area dealing primarily with sustainability of capital investments.

Operational knowledge for the practice of FM was found to exist within literature, current software, and current practice. This body of knowledge appears to be less well developed than areas such as project and construction management (e.g., as observed by the relative number and range of books and scholarly literature in these areas). Some efforts were found to formalize FM knowledge and practice in models, but these were found to be few and sparse and partial in their coverage of the breadth and depth of FM concerns. One of the research challenges addressed in this dissertation was to synthesize the available knowledge sources into a formal model for the practice of maintenance management. The primary motivation for this was a step in the development of IT tools for FM, but the resulting process model also offers other benefits in structuring the organization and management FM knowledge and FM operations.

Following the investigation of facilities maintenance management functions, and the development of process models, the methodology for designing integrated IT tools requires the development of data models. As discussed in this chapter, the IFC models have already been developed as an industry-wide solution for data interoperability throughout the lifecycle of AEC projects. There has been some, but relatively little FM
models for FM applications. A second major research challenge addressed in this
dissertation is the development of data model requirements to support FM and an
assessment of the IFC models to server this purpose. The research explores the
integration capabilities offered by the application of standard data models, and recognizes
the significance of data standards and product modeling as enabling technologies for
representing data and knowledge, and using these to deliver software interoperability.

It has also been shown in this chapter that software to support the FM industry does exist,
but that it has generally been developed to provide standalone solutions (that do not
conform to an overall framework) to a wide range of tasks throughout FM practice.
Examples include MicroROOFER and BUILDER providing capabilities for inventory
collection and condition assessment information collection for specific building systems;
MAXIMO providing capabilities for maintenance planning, scheduling and procurement
of machinery and components in plant facilities; and RECAP providing capabilities for
maintenance work order reporting and budgeting. This class of software has led to large
volumes of independent, loosely structured data with poor interoperability. No software
solutions were found that tightly integrated data throughout the full life cycle of AEC
projects (this is not surprising since it requires a standard like the recent IFCs to achieve
this). A final research challenge addressed in this dissertation is to explore what benefits
might be attained by creating FM software that is capable of tight integration with a wide
range of AEC data and applications.
Chapter 3
Maintenance Management Process Model

Chapter Abstract: This chapter provides a description of the proposed asset maintenance management framework. The main challenge faced in the development of the framework is the formalization and representation of FM data from a large and unstructured body of knowledge. Presentation of the framework in this chapter is in the form of a process model. IDEF0 notation was used to visually represent the selected management processes, their supporting functions with their logical sequence, and information requirements. A discussion as to the advantages of using the process modeling technology to describe the proposed framework is included at the end of the chapter.

3.1 Proposed Generic Maintenance Process Model

This chapter presents the development of an integrated framework for maintenance management of infrastructure built-assets (Hassanain et al. 1999; Hassanain et al. 2000; Hassanain et al. 2001a; Hassanain et al. 2001b). This framework was first conceptualized by the author (along with the research supervisor Professor Thomas Froese, Construction Engineering and Management Group, and Kevin Yu, a graduate student from the University of British Columbia) while attending the FM domain experts group at an IAI meeting in Seattle, Washington, 16-17 September 1997. The framework was subsequently developed by the author as part of the research for this dissertation.

The framework was motivated by the desire to develop IT solutions for the FM industry. However, the framework is useful beyond its role in supporting IT. The framework, presented as a process model, is generic, meaning that the activities involved can be applied to non-specific assets, rather than to a specific asset type (for example, the framework can be applied to systems such as roofing systems, cladding systems, heating, ventilation and air-conditioning (HVAC) systems, etc.). Further, the framework can be applied at both the level of individual assets or on a network of projects. The framework can also be used to analyze current maintenance management practices in any facilities management organization engaged in managing several assets, regardless of whether the tasks involved are implemented by in-house staff or professional maintenance contractors.

The proposed framework model is unique in the sense that it describes a unification or collection of diverse knowledge areas that have been analyzed and introduced to the FM domain in a formalized and standardized view as described in this chapter. It consists of five sequential processes. For each of the processes, the author has defined a number of supporting activities, with their logical sequence and information requirements. A detailed description of the processes and the functions within is provided below. It should be noted that, while every maintenance project is likely to be unique, some of the
identified functions within each process can be omitted depending on the characteristics of the asset being examined. The five processes forming the proposed framework are as follows:

1. Identify Assets (referred to as node “A”)
2. Identify Performance Requirements (referred to as node “R”)
3. Assess Performance (referred to as node “P”)
4. Plan Maintenance (referred to as node “M”)
5. Manage Maintenance Operations (referred to as node “O”)

The development methodology for the framework follows a process-oriented methodology set by the IAI to define data standards, in the form of IFCs. The methodology involves developing usage scenarios, which are descriptions of situations that show the use of IFCs to carry out a selected process, detailed definition of the processes, and a visual representation of the supporting functions with their logical sequence and information requirements.

The generic framework is described schematically as an IDEF0 process model diagram, as shown in figure 3.1. As indicated earlier, the IDEF0 notation (Integration Definition for Function Modeling) was selected within the IAI because of the formal documentation support and the extensive software support (IAI 1999). A process model describes the activities that exist within a business process. It defines the tasks that need to be undertaken within each process, and illustrates how and what information needs to be communicated between tasks (Federal 1993). A series of interrelated diagrams illustrating information flow from one activity to another, at different levels of detail, are presented throughout this chapter. Appendix A provides a description of the process modeling methodology.

![Figure 3.1: General processes involved in maintenance management model](image-url)

39
3.2 The "Identify Assets" Model

3.2.1 Process Definition

The "Identify Assets" process (node “A” in Figure 3.1) involves carrying out an inventory activity to identify the assets that may require maintenance operations within their service life. An asset may be defined as a uniquely identifiable element or group of elements which has a financial value and against which maintenance actions are recorded (IAI 1999). Service life may be defined as the actual period of time during which the asset, or any of its components, performs without unforeseen costs of disruption for maintenance and repair (CSA-S478 1995). Asset data are obtained from asset registers. An asset register may be defined as a table which records the assets of an organization, within which the value of all assets can be determined (IAI 1999). The inputs necessary to carry out the "Identify Asset" process are: an already existing facility and a set of resources, as displayed by the input arrows in node “A” in Figure 3.2. The output is a list of assets requiring maintenance. This process is broken down into three functions as shown in Figure 3.2. The following paragraphs provide a description of the functions involved.

![Figure 3.2: Node A, identify assets](image)

3.2.2 Process Activities

Identify Assets for Evaluation (A.1): serves to identify the assets (against which maintenance, repaired or renewal activities are carried out) that contribute to the wealth of an organization. Data identified in the asset register may include asset name, identifier, location, expected life, original value, current value, depreciated value, total replacement value, incorporation date, commissioning date, and warranty duration from manufacturer (IAI 2000).
Select Building Products to Treat as Assets (A.2): serves to identify the specific building products that may undergo maintenance work, hence treated as assets for the purpose of evaluating their need for maintenance, repair or renewal activities, in an asset management information system.

Compile Inventory of Assets (A.3): serves to compile a record of the identified assets in an asset management information system for the purpose of identifying their performance requirements, assessing their current condition against a pre-determined set of performance requirements, identifying the specific maintenance, repair or renewal activities that needs to be carried out, and developing schedules for performing these activities. These assets are not yet evaluated at the time of compiling the inventory list.

3.3 The “Identify Performance Requirements” Model

3.3.1 Process Definition

The “Identify Performance Requirements” process (node “R” in Figure 3.1) includes functions required to identify categories of performance requirements of an asset as a unified entity (e.g. a building), as well as the components that make-up the assembly of the asset (e.g. technical building systems). Performance may be defined as the behavior of a product related to use (ISO-6241 1984). The scope in this process also extends to identifying performance indicators and their means of expression within each category of performance requirements. The input to this process is a list of assets (e.g. buildings and/or technical building systems and system components) requiring maintenance, which are obtained from asset registers. The outputs are statements of the performance requirements as well as a range of acceptable performance values. This process is broken down into five functions as shown in Figure 3.3. The following paragraphs provide a description of the functions involved.
3.3.2 Process Activities

**Verify Current Use of Asset (R.1):** serves to verify and/or analyze the current use of the asset and the occupancy conditions against those specified in specification documents, at the beginning of the commissioning phase. In this function, performance agents which may impact the behavior of the asset may be identified according to their origin and nature (ISO-6241 1984).

**Identify Performance Requirements of Asset (R.2):** serves to identify the performance requirements that the asset, as a unified entity, has to meet. A performance requirement may be defined as user requirement expressed in terms of the performance of the product (ISO-6241 1984). In this function, performance requirements are defined without imposing constraints on the form or materials of the solutions proposed to fulfill these requirements. Compiled below are the most representative categories of performance requirements in agreement with a number of references on the concept of total building performance (ISO-6241 1984), (ISO-6242-2 1992), (Hartkopf et al. 1986), and (Blachere 1993):

1. Durability requirements,
2. Fire safety requirements,
3. Air quality requirements,
4. Acoustical quality requirements,
5. Thermal quality requirements, and
6. Lighting requirements.

**Identify Performance Requirements of Asset Components (R.3):** This function runs in parallel with function R.2. It should be considered when the performance of a specific
asset component is in question. It serves to identify the performance requirements that the asset components (e.g. technical systems) have to meet. The reason is that technical systems are designed to their component performance requirements, resulting in the inability of two systems (component-to-component interfaces) to satisfy all performance requirements specified.

Groups of performance requirements for low-slope conventional roofing systems were highlighted by Lounis et al. (1998a) as:

1. Water tightness: prevention of water leakage into the building, a requirement ensured by the waterproofing membrane and flashings,
2. Energy control: prevention or minimization of heat (or cooling) exchange between the interior and exterior, a requirement ensured by the thermal insulation,
3. Condensation control: prevention of water vapor condensation within the roofing system using the vapor barrier,
4. Air leakage control: minimization of air leakage through the roof system using the air barrier,
5. Load accommodation: ability to sustain dead and live loads by the structural deck,

**Identify Performance Indicators (R.4):** serves to identify the parameters adequate to measure all aspects of performance in performance categories. Review of literature indicated that the diversity of performance requirement categories of assets and/or components defies the definition of a single parameter which is adequate to measure all aspects of performance. Hence, to judge performance effectively, each category of performance is considered separately. For example, while some of the indicators of the durability requirement category include the existence of deflections, cracks and corrosion, some of that of the fire safety requirement category include duration of evacuation time, survival time, provisions of smoke detectors and exist signs.

**Identify Performance Values (R.5):** serves to state the upper and lower limits of acceptable performance values, hence providing a range of acceptable solutions to fulfill the performance requirements. While international standards may specify performance categories for particular assets and components, specification of performance values is the task of building designers (ISO-7361 1986). Each performance requirement has a "comfort zone" that establishes the limits of acceptability for the type of occupancy in a building. These limits in turn are translated into regional standards and codes. Such limits are established by the physiological, psychological, sociological and economic requirements of the occupancy (Hartkopf et al. 1986).

### 3.4 The “Assess Performance” Model

#### 3.4.1 Process Definition

The “Assess Performance” process (node “P” in Figure 3.1) includes functions required to assess the condition of an asset, and determine the deviation in the performance, which
occurred through its service life. It involves identifying the performance assessment method(s) and their pre-set frequencies, depending on the configuration of the asset being examined. The objective of this process is to catalog assets and/or components that cease to meet the performance requirements specified in process R and, hence, require a maintenance, repair, renewal, or “do nothing” action. Maintenance includes general activities such as cleaning drains, removing obstructions in roofs, and replenishing depleted protection fluids in mechanical equipment. Repair includes unplanned intervention activities, performed to rectify situations of distresses found. Renewal includes activities to install a new asset and/or component to replace the one in-place, due to economic, obsolescent, modernization or compatibility reasons (Vanier 2000). One example would be installing a new assembly of roofing system either above the existing system, or after disposing of the old roofing system. “Do nothing” includes postponing or ignoring maintenance, repair or renewal. The inputs to this function are statements of acceptable performance values from process R. The outputs are statements of the asset condition and a range of management options that objectively specify a set of actions when a specific set of conditions occurs. This process is broken down into four functions as shown in Figure 3.4. The following paragraphs provide a description of the functions involved.

3.4.2 Process Activities

**Identify Condition Assessment Technique (P.1):** this function identifies the condition assessment technique “inspection” that would be followed to assess the performance of an asset and/or its components, and indicate the necessity for required corrections. Condition Assessment Surveys (CAS) may vary from being simple, visual walk-through, to thorough analysis that may include in-depth review of background documentation, in-situ and laboratory testing and disassembly of selected components (Cole and Waltz 1995).
Assess Asset Condition (P.2): This function is the core function of the “Assess Performance” process. All functions within this process exist to support this function.

Identify Distress (Anomaly) (P.3): serves to identify the distress or anomaly found in the asset through the CAS. Identifying distresses is achieved through carrying particular functions, depending on the type of asset being assessed. This function is broken down into 4 sub-functions as shown in Figure 3.5. The following paragraphs provide a description of the functions involved.

Identify Distress (Anomaly) Type (P.31): serves to describe the type of the distress found.

Identify Distress Severity Level (P.32): serves to describe the severity level of the distress found. Severity levels might range from low, medium to high.

Measure Quantity of Distress (P.33): quantities are measured as number of units, or combined length, or areas of distress, depending on the type of the distress found.

Document Distress Cause(s) (P.34): causes might range from an aggressive environment, inadequate design, poor workmanship to lack of maintenance.

Figure 3.5: Node P.3, identify distress (anomaly)

Identify Management Options (P.4): serves to describe the range of management options available when specific sets of conditions occur or are imminent to occur. Management options include carrying out one or a combination of the following: Maintenance, Repair, Renewal (MRR) or doing nothing. An input to this function is a statement on the condition of the asset being examined. A review of literature indicates that asset conditions may be expressed either quantitatively, as a numerical rating (i.e. condition index), or qualitatively as a categorical rating.

To exemplify the quantitative approach, Table 3.1 illustrates a range of MRR actions for roofing systems. Implementing a particular action is dependent on the value of the
Roofing Condition Index (RCI), as obtained from MicroROOFER application (Bailey et al. 1990). This approach may also be viewed as being a qualitative approach based on a numerical scale.

Table 3.1: MRR options corresponding to values of roofing condition index

<table>
<thead>
<tr>
<th>RCI</th>
<th>MRR Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>86-100</td>
<td>Routine Maintenance</td>
</tr>
<tr>
<td>71-85</td>
<td>Minor Repairs Needed</td>
</tr>
<tr>
<td>56-70</td>
<td>Moderate Repairs Needed</td>
</tr>
<tr>
<td>41-55</td>
<td>Major Repairs Needed</td>
</tr>
<tr>
<td>26-40</td>
<td>Replacement Probable</td>
</tr>
<tr>
<td>11-25</td>
<td>Replacement Needed</td>
</tr>
<tr>
<td>00-10</td>
<td>Replacement Critical</td>
</tr>
</tbody>
</table>

To exemplify the qualitative, and alternative, approach in determining a particular management option to proceed with, Table 3.2, as reported by Smith (1988), illustrates a qualitative or subjective method of assigning condition ratings, and hence implementing a particular maintenance, repair and renewal option. These categories of physical conditions were developed through a conditional appraisal program developed for the National Health Service in the United Kingdom to define categories of physical conditions of the estate.

Table 3.2: Subjective listing of condition rating categories

<table>
<thead>
<tr>
<th>Condition</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>The element is as new and can be expected to perform adequately for its full normal life.</td>
</tr>
<tr>
<td>B</td>
<td>The element is sound, operationally safe and exhibits only minor deterioration, which can be corrected by routine maintenance.</td>
</tr>
<tr>
<td>C</td>
<td>The element is operational but major repair or replacement will be needed soon. Usually a time frame is established.</td>
</tr>
<tr>
<td>D</td>
<td>This category would cover those elements where there is a serious risk or imminent breakdown or of their being unacceptable on health and safety grounds.</td>
</tr>
</tbody>
</table>

3.5 The “Plan Maintenance” Model

3.5.1 Process Definition

The “Plan Maintenance” process (node “M” in Figure 3.1) includes functions that are required to determine maintenance priorities based on three identified conflicting management objectives as one of the methods involved in planning maintenance. While these objectives were identified in the context of a specific asset, roofing systems (Lounis et al. 1998b), the author believes that analysis of the same set of objectives is valid for non-specific assets. These objectives are:
1. Minimizing maintenance cost: this objective is treated through performing a life cycle costing analysis to predict initial and future expenditures associated with a maintenance, repair or renewal operation over the life cycle of an asset.

2. Maximizing asset performance: this objective is treated through predicting the performance of an asset for each of the different maintenance options. One method of performance prediction of assets is based on the principles of Markov chain, which determines the deterioration in condition through a series of algorithms using Markovian probability matrices and condition states (Lounis et al. 1998b).

3. Minimizing risk of failure: this objective is treated through concurrently considering the probability of failure and the consequences of failure. One method to calculate the probability of failure is obtained through the Markovian model. A consequence of failure is a statement of cost figures associated with loss of productive time and damage to surroundings (Lounis et al. 1998b) or damage to other systems that could in turn exacerbate damage.

The inputs to this process are statements of the asset condition and its components, as well as a set of management options to be implemented when a specific set of conditions occur or are about to occur. The output is an optimal decision or a strategy based on the result of the analyses carried out within this process. This optimal decision is translated into identifying maintenance workload to proceed, and as a result, a maintenance work order is issued so that maintenance jobs would be implemented. Another output of this process is a list of deferred maintenance jobs, which are of secondary priority and are awaiting completion. This process is broken down into five functions as shown in Figure 3.6. The following paragraphs provide a description of the functions involved.

Figure 3.6: Node M, plan maintenance
3.5.2 Process Activities

Predict Remaining Service Life (M.1): serves to perform an analysis to predict the performance and the service life of an asset.

Estimate Cost of Maintenance (M.2): serves to estimate the resources required to carry out the maintenance work requested. This function is broken down into 4 sub-functions as shown in Figure 3.7. The following paragraphs provide a description of the functions involved.

Estimate Number and Trade of Workers (M.21): serves to estimate the number and the trade of workers needed to perform the maintenance work requested. Some maintenance jobs require a crew of a single trade. Some jobs require multiple crews of multiple trades.

Estimate Number and Type of Equipment (M.22): serves to estimate the number and type of equipment needed to execute the maintenance work. The number of hours the equipment is going to be used can be estimated. The hourly rate for using the equipment can be determined.

Estimate Quantity and Type of Materials (M.23): serves to estimate the quantities and the type of material needed to perform a requested maintenance work. Some maintenance jobs are simple and require only one type of material. Some jobs are complex and require the combination of several materials.

Determine Total Cost of Maintenance Activity (M.24): serves to estimate the total cost of carrying out the requested maintenance job. The estimated cost would be the summation of the following cost items: man-hours, equipment/tools, and materials needed to perform the work.

Figure 3.7: Node M.2, estimate cost of maintenance
Perform Multi-objective Decision Analysis (M.3): serves to perform a risk-based multi-objective decision analysis to recommend a decision taking into consideration conflicting management objectives. These objectives are: minimization of maintenance and repair costs, maximization of the building system performance and minimization of risk of failure.

Identify Maintenance Workload to Proceed (M.4): serves to identify first class priority maintenance jobs to be carried out based on the results obtained from the above mentioned analyses.

Identify Deferred Maintenance Workload (M.5): serves to identify the remaining maintenance, repair or renewal jobs, which have been deferred due to lack of funds in annual budget cycles, and for being of less priority, that should be carried out after the completion of the first class priority jobs.

3.6 The “Manage Maintenance Operations” Model

3.6.1 Process Definition

The “Manage Maintenance Operations” process (node “O” in Figure 3.1) includes functions that are required to support the execution of maintenance operations and the implementation of maintenance, repair or renewal activities. The inputs necessary to carry out this process are a list of maintenance workload awaiting completion and a set of resources. The output is an operational facility. This process is broken down into five functions as shown in Figure 3.8. The following paragraphs provide a description of the functions involved.

![Figure 3.8: Node O, manage maintenance operations](image-url)
3.6.2 Process Activities

File Maintenance Work Order (O.1): serves to communicate the need for carrying out maintenance work to the operational staff in a facilities management organization. The communication takes the form of a maintenance work order. Work orders can be received in both oral and written forms. A work order can be the result of either a planned maintenance work, or unplanned maintenance work, or both. Planned maintenance work may be defined as work organized and carried out with forethought, control and the use of records to a pre-determined plan (Then 1982). Unplanned maintenance work may be defined as work carried out outside planned maintenance and for which there is no regular pattern of inspections (Then 1982). Filing work orders provide the basis for planning, scheduling and effectively tracking maintenance workload. This function is broken down into 6 sub-functions as shown in Figure 3.9. The following paragraphs provide a description of the typical tasks associated with filing and processing a maintenance work order. Each of these tasks is defined below.

Define Exact Problem (O.11): serves to describe the failure or defect in an asset (e.g. technical building system) that calls for a maintenance action to restore it to its original condition.

Define Location of Problem (O.12): serves to specify the location where the problem exists.

Identify Contact Person (O.13): serves to define the information of the person requesting the maintenance action to be carried out.

Note Request Date (O.14): serves to note the date for which the call for a maintenance action is requested.

Define Resource Type (O.15): serves to describe the type of resource needed to perform the maintenance work. Resources can be manpower, equipment and special tools required to perform the work.

Determine Quantity of Work (O.16): serves to determine the quantity of maintenance work, based on the extent of the defect described in the work order.

Plan Maintenance Activities (O.2): includes functions required to plan maintenance activities, and the method followed to achieve them. In essence, it involves setting up a work plan, for which questions like what, who, where, when and how the operational staff in a facility management organization will respond to a filed work order. It should be noted that there exist similarities between the functions involved in planning and scheduling a maintenance operation and those of a construction project. The input to this function is a list of maintenance jobs awaiting completion. The output is a maintenance work plan. It should be noted that every maintenance project is likely to be unique, meaning that some of the functions within this process can be overlooked depending on the characteristics of the asset undergoing maintenance. This function is broken down
into 5 sub-functions as shown in Figure 3.10. The following paragraphs provide a description of the functions involved.

**Choose Maintenance Work Method (O.21):** serves to identify the method to be followed in performing the maintenance work. The choice of a specific maintenance work method over others will directly influence the cost and the duration of the maintenance work carried out.

**Define Maintenance Activities (O.22):** involves defining a series of maintenance activities through the description of the given problem in the work order and the chosen work method to perform the work. This function is broken down into 3 sub-functions as shown in Figure 3.11. The following paragraphs provide a description of the functions involved.

Figure 3.9: Node O.1, request maintenance work order
Define Repetitive Activities (O.221): serves to list/outline the activities that would be carried out more than once as a result of processing a single work order. One example to illustrate this function would be having to replace the window glazing in several floors in one building.

Define Unique Activities (O.222): serves to list/outline the activities that would be carried out only once as a result of processing a single work order. One example to illustrate this function would be having to paint the walls of one room in a building.

Define Lower-Level Activities (O.223): serves to define implicit activities when performing a maintenance activity. For example, painting work would be the general description in a requested work order (either repetitive or unique). To be able to perform this task, certain sub-tasks have to be carried out in process (lower-level activities), such
as removing old paint, sanding wall surfaces, applying a sealer, a first coat of paint and then a second coat of paint.

**Identify Precedence Relationship (O.23):** serves to determine the logical sequence of the steps involved to carry out a requested maintenance work. This function is broken down into 3 sub-functions as shown in Figure 3.12. The following paragraphs provide a description of the functions involved.

**Define Starting Activity (O.231):** serves to identify the first activity to start with in the process of carrying out a maintenance work.

**Define Successor Activity (O.232):** serves to identify the successor activities to the starting activities, and their sequencing logic.

**Determine Lag between Activities (O.233):** this function involves defining the time between the completion of one activity and the commencement of the successor activities.

**Estimate Activity Duration (O.24):** serves to estimate the time a maintenance activity takes to be completed. The inputs to this function are information on estimated resources, productivity rates, sequencing logic, quantity of work, and location of work. Estimates of productivity rates for various facilities maintenance and repair tasks can be obtained from published cost data such as RS Means Facilities Cost Data (1997). The output of this task is a maintenance plan.

**Schedule Maintenance Activities (O.3):** includes functions required to schedule maintenance activities. The inputs to this function are a list of maintenance activities, activity duration, estimated resources and sequencing logic. The output is a maintenance schedule. This function is broken down into 3 sub-functions as shown in Figure 3.13. The following paragraphs provide a description of the functions involved.
Figure 3.13: Node O.3, schedule maintenance activities

**Determine Activity to Proceed (O.31):** serves to denote the activity to proceed with to accomplish the requested maintenance work.

**Determine Work Date (O.32):** serves to indicate the date for the commencement of the activity chosen to proceed with.

**Determine Activity Location (O.33):** serves to denote the location where the requested maintenance work will be performed.

**Accomplish Maintenance Workload (O.4):** includes functions involved in accomplishing the maintenance workload. The inputs to this function are sequencing logic, activity location, the activity to proceed with, and activity duration. The output is a completed maintenance workload. This function is broken down into 4 sub-functions as shown in Figure 3.14. The following paragraphs provide a description of the functions involved.
Figure 3.14: Node O.4, accomplish maintenance workload

**Set Up Work Area (O.41):** serves to establish and organize the work-space, depending on the complexity of the maintenance work requested.

**Prepare Resources (O.42):** serves to mobilize and coordinate the resources (manpower and equipment) needed to perform the work.

**Perform Work (O.43):** this function is the core function of “Accomplish Maintenance Workload” model. All functions within this process exist to support this function.

**Clean Up Work Area (O.44):** serves to denote the task of separating the waste products and removing them from the work space after the maintenance work is carried out. Partially consumed resources would also be salvaged for future use if needed.

**Record Maintenance (O.5):** includes functions required in recording accomplished maintenance work. These functions are usually noted by the crew that performed the work. The input to this function is a completed work unit. The output is an operational facility. This function is broken down into 4 sub-functions as shown in Figure 3.15. The following paragraphs provide a description of the functions involved.

**Report Completed Work (O.51):** serves to report the completion of the work requested. In some circumstances, with the start of maintenance operation, the crew might discover much larger problems necessitating an increase to both man-hours and material quantity than originally estimated for. Such case of change of work scope prompts revising man-hour and materials estimate to reflect actual crew productivity and actual consumption of resources. Reporting completed work also serves to describe how well the maintenance work has been performed for quality control purposes.
**Report Consumed Resources (O.52):** serves to report the actual amount of resources consumed to accomplish a maintenance work order. This monitoring effort helps determine if the work has been accomplished at the lowest cost through examining manpower utilization, material usage and costs. Analysis of manpower utilization can be achieved through collecting data from all maintenance work orders on the number of man-hours and the quantity of material that each trade used to complete the work. Comparisons can then be made to determine the relative efficiency of current operations (Magee 1988). This function is broken down into 3 sub-functions as shown in Figure 3.16. The following paragraphs provide a description of the functions involved.

**Report Number and Trade of Workers (O.521):** serves to define the actual number and trade of crews who performed the work. This function also serves to compare the actual man-hours spent against the estimated man-hours.

**Report Type and Number of Equipment Used (O.522):** serves to report the type and actual number of equipment needed to perform the work. This function also serves to compare the actual equipment-hours needed against the estimated equipment-hours.
Report Type and Quantity of Material(s) Used (O.523): serves to report the type and actual quantity of material(s) needed to perform the work. This function also serves to compare the actual quantity of materials consumed against the estimated quantity.

Update As-built Drawings (O.53): serves to instruct facilities management staff to update as-built drawings to reflect the changes, if any, to the configuration of the asset.

Report Actual Activity Duration (O.54): serves to report the actual time taken to perform the work. This function also serves to compare the actual duration of an activity against the estimated duration. This monitoring effort verifies that the assigned work crew is appropriate and the productivity rate is acceptable.

3.7 Knowledge Feedback within the Proposed Model

The IDEF₀ model illustrated in Figure 3.1 shows the generic framework of maintenance management. It shows facility management tasks that are essentially sequential on a "per issue" basis. At any point in time, new information and thinking can cause the facility manager to go back to reconsider earlier work carried out in a preceding process. This creates opportunities for knowledge feedback with the maintenance management model. As a result, outputs from a succeeding management process become controls to the proceeding process. Controls are entities which influences or determine the process of converting inputs to outputs (Federal 1993).

It should be also noted that when considering the overall FM process, these tasks are being carried out over and over again, on a multitude of issues, at varying levels of detail; and any or all of these tasks may be going on at any given time. From this perspective, there can be a great deal of interdependency between the tasks. In particular, most of the information inputs to any of the tasks are based in some degree upon historical information, which comes about from the culmination of the output of previous cycles of the FM tasks. Although not displayed in Figure 3.1, it is implied that a feedback loop exists between any two succeeding and proceeding processes, respectively.
3.8 Discussion

The research proposed a generic framework for asset maintenance management. It describes a collection of knowledge areas within the domain of asset maintenance management. Although the knowledge areas described in the framework have previously existed in practice and documented in the literature, they have not yet been introduced to the FM domain in a formalized and standardized view as presented through the development of the process model in this dissertation. For example, although the development of formal process models is considered to be part of the IAI model development methodology, no models similar in scope to the framework presented here have been developed within IAI activities. This framework comprises one of the contributions of this dissertation to the FM domain. It also constitutes a planned deliverable to the BELCAM project, in which the University of British Columbia is a member in its consortium.

The framework can act as a generic policy guidelines for the conduct of maintenance in any organization engaged in managing diverse collection of assets. It presented a solution to bridge the gaps in the practice of maintenance of built-assets by asset managers, such as implementing procedures for identifying performance requirements of assets and strategies for considering conflicting management objectives in decision-making. The framework served to standardize process descriptions, the activities that need to be undertaken, and the methodology of how and what information needs to be communicated between activities.

The framework is schematically described as an IDEF0 process model. Advantages gained from using IDEF0 process models can be seen in the legibility exhibited in defining boundaries and responsibilities of functions within management processes, as well as the potential to improving the level of communication between the project’s participants. Illustrating the framework in the form of IDEF0 notation emphasizes the interaction and dependencies among knowledge areas. Outputs from one management process become inputs to another in a subsequent hierarchy. The structure of the framework becomes flexible and robust. Updates in knowledge can be accommodated through incorporating new management processes and/or activities, as well as establishing new sequencing logic for these processes and/or activities.

The listing below provides elaboration on specific aspects in the framework:

1. Regardless of the project delivery mode, it is envisaged that the execution of the "Identify Assets" process would be optimal in Design-Build-Operate projects, where information on physical assets may be captured and recorded during both design and construction phases. Design-Build-Operate projects are growing trend in the construction industry in the United Kingdom (Wix 1999). Computer-Aided Design and Drafting (CADD) tools can provide a significant role in executing the process of "Identify Assets". This is mainly due to the ease of transferring data (library data associated with building elements in floor plans and three dimensional models) obtained from the design and construction stage of a facility to asset registers in the Asset Management Tool. A list of building products that
can be treated as assets can then be decomposed to evaluate their need for maintenance, repair or renewal activities.

2. Within the “Identify Performance Requirements” process, performance requirements can be defined for whole buildings and technical systems within buildings. However, treating performance requirements on the level of whole buildings, rather than the level of individual technical systems, satisfies the concept of total building performance. Literature (Hartkopf et al. 1986) revealed that although a building system may provide adequate performance in one dimension, it might fail in other area due to specification or context. Therefore, care should be taken not to identify the various measurements with individual technical systems and assemblies (in isolation), such as roofs, or walls, since it is usually the interactions of these systems or assemblies that fail.

3. In the “Assess Performance” process, a condition assessment survey of an asset relies on counting visible defects. While this method may be appropriate for accessible and easy to observe systems such roofing systems, other systems such as structural members, plumbing piping, electrical wiring may be difficult to access and observe. This, in turn, necessitates looking for clues such as uneven floors, out of plumb walls, water stains, musky odors and unusual noises to ascertain condition (Uzarski 1999).

4. Within the “Plan Maintenance” process, it should be noted that exceptions to the view of considering conflicting management objectives in planning maintenance is possible, and that other subjective methods may exist.

Overall, the asset maintenance management framework served to identify research and practice avenues throughout the practice of asset management, where integration of data and knowledge through the development and implantation of object-oriented models holds great potential for improving operations, reducing dependency on paper-based views of project information.
Chapter 4
Maintenance Management Data Model

Chapter Abstract: This chapter presents the development of an object model for maintenance management of roofing systems as a case study to demonstrate the applicability of the proposed generic framework presented in chapter 3, to a type of a building product that has been treated as an asset. A challenge faced in the development of the maintenance management data model is the fact that there has been some, but relatively little, FM input to the IFC standards to date, and virtually no implementation or assessment of these standards for FM applications. The model was developed to be consistent with IAI standards. The model builds upon the existing IFCs in Release 2X of the IFC model to define object requirements and relationships for the exchange and sharing of maintenance information between applications. The chapter proposes several extensions to the IFCs including the representation of functional requirements, condition assessment of objects, inspection and maintenance tasks, and libraries of non-project-specific information.

4.1 Overview of the Data Model

The purpose of developing a process model is to define requirements for information sharing amongst the various management areas. Within the AEC domain, a large-scale effort is underway to create a general purpose data model, the IFCs, for information exchange throughout the project lifecycle. Some FM domain input has been incorporated into the current release of the IFCs, but to date, there has been virtually no assessment and validation of how well the IFCs can support maintenance management. The chapter presents a data model that the author developed by assessing the information requirements detailed in the preceding process model and mapping these requirements to the IFC model. It is shown that some data requirements can be met by the FM-specific classes already contained within the IFCs; some data requirements can be met by using generic classes that have been included in the IFCs, even through they were not developed specifically with FM in mind; and other data requirements require that new classes be added into the IFCs. Figure 4.1 presents this data model in the form of a class diagram for the roofing maintenance management. Roofing systems, specifically low-slope conventional assemblies, were chosen in the BELCAM project as a building system that is representative of the built-asset maintenance domain (Vanier and Lacasse 1996).

Figure 4.1 shows a collection of project-specific model elements (i.e. classes and relationships), and project-independent, or "library", elements, which represent information that can apply across a wide range of projects. The model, while making use of existing IFCs within Release 2X, proposes a set of IFCs that should be considered within Release 3.0 or later. In this chapter, throughout Figures 4.1 to 4.6, existing IFCs
within Release 2X are shown in plain text, while proposed IFCs to be included in Release 3.0 or later versions are shown as underlined text. Appendix B lists the newly proposed set of IFCs with their properties and corresponding data types.

The model in Figure 4.1 illustrates that a roofing system (IfcRoof), a subtype of IfcProduct, can be associated with various information (in this model, the author actually associates much of this information with a related IfcAsset object, which is explained later in the chapter, but have been omitted here for simplification). The roof can be associated with a series of defined functional requirements (IfcFunctionalRequirement). The roof can be assessed against these functional requirements through inspections or "condition assessment surveys" (CAS) (IfcInspectionTask). The CAS determines the condition (IfcCondition) of the roofing system, which may, in turn, lead to a range of options to maintain, repair, or renew the existing system (IfcMRRTask). Assessment of the risks (IfcRiskSchedule) and costs (IfcCost) associated with a certain management option would be considered, as would the resources (IfcResource) required to execute a maintenance, repair, or renewal (MRR) task.

This approach uses "type" objects to represent "library" information. That is, project-specific objects in a project data set can reference objects that represent types of that object. For example, a specific resource, a roofing maintenance crew, can be associated with a resource type object that describes attributes typical of all roofing maintenance crews (typical crew makeup, productivity rates, unit costs, etc.). The specific crew object can inherit these values from the type object, or can define overriding values. The concept of type objects has many uses within the IFCs (product component types, actor types, resource types, work task types, etc.). Although the concept of type objects did not exist in previous releases of the IFCs, they have been added in the IFC 2X model. Also, the concept of type objects is closely related to the concept of linking objects within a project object model to additional information referenced from an external library. A project is currently underway within the IAI to investigate external library issues for the IFCs (IAI 2000).
4.2 Development Methodology

The maintenance management data model was developed following a methodology similar to that used within the IAI (IAI 1999). The methodology began with developing initial process models (as shown in Figures 3.1-3.16) and usage scenarios (presented in the following sections) and led to defining data model extensions to the existing IFC classes. These data models are shown in EXPRESS-G notation (ISO 10303-11 1994) in Figures 4.2 through 4.6 (boxes represent classes, corresponding to real world items of interest; solid lines represent relationship between classes and heavy lines represent subtype inheritance relationships).
4.3 IFC Data Models

4.3.1 The “Identify Assets” Model

4.3.1.1 Usage Scenario

For the example of the roofing asset, the facility management team, through its administrative staff, carries out an inventory activity, which identifies all roofing sections to be managed. Managing roofs on the roof-section level provides a more precise means of evaluating condition and determining maintenance, repair, and renewal requirements (Bailey et al. 1990). For each roof section, the inventory contains basic information on occupancy (the building use function occurring under the roof section), section area, type of structural frame, thermal insulation, waterproofing membrane, and flashing.

As indicated earlier in Chapter 2, sets of assertions are complementary parts of usage scenarios (IAI 1999). They are drawn from the definitions of processes (i.e. the five sequential processes in the maintenance management framework). The objective of developing assertions is to help identify classes, relationships, cardinality of relationships and attributes. These assertions then can be modeled and implemented. The following set of assertions is made:

- A **Product** (such as **Roof**) can play the role of an **Asset**
- A **Roof** contains one or more **RoofSection**
- A **RoofSection** has an **Insulation**
- A **RoofSection** has a **Slab**
- A **RoofSection** has a **Flashing**
- A **RoofSection** has a **Membrane**

4.3.1.2 IFC Model

An IfcAsset represents the fact that an object is being treated as an asset for FM economic and maintenance management applications. IfcAsset is a subclass of IfcGroup, and is associated with IfcInventory, also a subclass of IfcGroup, for which an inventory of assets may be carried out and defined through the attribute “DefinedType”. IfcRoof object is treated as a type of asset. The location characteristic of the roof is determined through the location attribute of IfcBuildingElement of which IfcRoof is a subtype. The assignment relationship of an inventory to an actor is modeled by IfcRelAssignsToActor as an objectified entity carrying information on the role of the actor played within the context of the assignment to the asset(s).

While the central focus of the IFCs has been on representing physical project data, the scope of the IFCs extends to representing non-physical data (Froese and Yu 1999). The IFC model in Figure 4.2 provides a representation of the physical things (components of
the roofing system, i.e. IfcRoofSection, IfcInsulation, IfcFlashing, IfcSlab, and IfcMembrane) and non-physical things (IfcInventory and IfcActor).

Figure 4.2: IFC Model for identifying roofing system components

4.3.2 The “Identify Performance Requirements” Model

4.3.2.1 Usage Scenario

The facility management team, through its engineering staff, identifies the categories of performance requirements of roofing systems, as well as the performance indicators and their means of expression within each category. These requirements can be either arbitrarily defined by the engineering staff, or selected from pre-defined list of requirements. Performance requirements of roofing systems have been identified as water tightness, energy control, condensation control, air leakage control, load accommodation, and maintainability (Lounis et al. 1998a).

The following set of assertions is made:

- An Asset has one or more FunctionalRequirement
- A FunctionalRequirement has a FunctionalRequirementType

4.3.2.2 IFC Model

Figure 4.3 shows the data model for representing the performance requirements of an asset. If an IfcAsset object is associated with roof insulation, for example, an IfcFunctionalRequirementType object may be used to represent a requirement to maintain a thermal resistance value (R-value), and an IfcFunctionalRequirement object
may be used to represent the specific outdoor environment. The general IFC classification system mechanism can be used to define a catalog or library of functional requirements.

To tie into the current release of the IFC model (Release 2X), IfcFunctionalRequirements can be associated with IfcProperty, which is an abstract generalization for all types of properties that can be associated with IFC objects through the property set mechanism. Similarly, IfcFunctionalRequirementType can be associated with IfcPropertyDefinition, which defines the generalization of all characteristics (i.e. grouping of individual properties) that may be assigned to objects. Figure 4.4 illustrates the existing property classes in the current release of the IFC model.

![Diagram of IFC Model for identifying performance requirements of roofing systems](image)

**Figure 4.3: IFC Model for identifying performance requirements of roofing systems**

![Diagram of Existing property classes in IFC release 2X model](image)

**Figure 4.4: Existing property classes in IFC release 2X model**

### 4.3.3 The "Assess Performance" Model

#### 4.3.3.1 Usage Scenario

The facility management team, through its specialist members (roofing trade), carries out one or both of the following scenarios: (a) a preventative maintenance action, represented as a scheduled condition assessment survey for predefined components of the roofing assembly, at predefined frequencies, to inspect the condition of the roof visually, assess its condition, and record anomaly severity level and quantity, (b) a corrective maintenance action, which is generated from a request by a facility user to rectify
anomalies found on the roof. A recommendation for MRR of the existing roofing system is made based on the observations recorded and condition rating of the roofing assembly.

The following set of assertions is made:

- An **InspectionOrder** orders one or more **InspectionTask**
- An **InspectionTask** has a **Frequency**
- An **InspectionTask** requires one or more **InspectionTest**
- An **InspectionTest** has an **InspectionResult**
- An **InspectionResult** determines a **Condition**
- A **Condition** dictates a **MRRTask**
- A **RoofSection** requires one or more **MRRTask**

4.3.3.2 IFC Model

As shown in Figure 4.5, the point at which an inspection is initiated is modeled as an IfcInspectionOrder, a subtype of IfcControl. An inspection activity (IfcInspectionTask), a subtype of IfcProcess, then takes place and a particular inspection method (IfcInspectionTest) is conducted, thus resulting in an inspection result (IfcInspectionResult). IfcProcess represents any general actions taking place in completing a work of design, construction, or facilities management (Froese et al. 1999). IfcInspectionTask is intended to describe the work processes that make up performance assessment of assets. Information such as schedule dates and duration is defined in IfcInspectionTaskType. The current condition (IfcCondition) can be then inferred from the inspection result. Determining the condition of a roof section dictates the need for a maintenance, repair, or renewal activity (IfcMRRTask).

![Figure 4.5: IFC Model for performance assessment](image-url)
4.3.4 The "Plan maintenance" Model

4.3.4.1 Usage Scenario

The facility management team, through its feasibility analysts, evaluates the recommendation for carrying out MRR, taking into consideration a range of conflicting management objectives. These objectives are minimizing cost, minimizing the risk of roof failure, and maximizing the performance of the roof based on the remaining service life (Lounis et al. 1998b).

The following set of assertions is made:

- A RoofSection requires one or more MRRTask
- A MRRTask has one or more CostValue
- A CostValue has one or more CostSchedule
- A RoofSection has one or more RiskSchedule

4.3.4.2 IFC Model

The IfcRiskSchedule (See Figure 4.6) represents a collector class for the risks associated with a roofing section. These risks are evaluated through concurrently considering the probability of roofing system failure and the consequences of failure. In this process, an evaluation of costs is also considered. Cost information (IfcCost) is represented as being related to the object being costed (IfcRoofSection) and to a cost schedule document (IfcCostSchedule) that describes the context of a list of cost values. The entity IfcCostSchedule can be used to represent any form of cost list, such as an estimate, a budget, or a unit price table (Froese et al. 1999).

![Figure 4.6: IFC Model for maintenance planning of roofing systems](image-url)}
4.3.5 The “Manage Maintenance Operations” Model

4.3.5.1 Usage Scenario

The facility management team, through its engineering staff, allocates resources to carry out a MRR task. Resources include subcontractor, equipment, material, and labor. The staff also determines MRR work methods as well as a schedule date and duration.

The following set of assertions is made:

- A RoofSection requires one or more MRRTask
- A MRRTask has a MRRTaskType
- A MRRTask uses one or more Resource
- A Resource has a ResourceType

4.3.5.2 IFC Model

In modeling maintenance processes, the association between processes and the products upon which they operate is illustrated through using an objectified relationship entity named IfcRelAssignsToProcess. As shown in Figure 4.7, a roof section (IfcRoofSection) is being assigned to undergo a maintenance, repair, or renewal task (IfcMRRTask). IfcResource (and its subclasses such as IfcMaterialResource, IfcEquipmentResource, IfcLaborResource, etc.) represent the resources used to complete a maintenance, repair, or renewal task (IfcMRRTask). Since processes use resources, the resource-use relationship is represented through the objectified entity IfcRelUsesResources, which stores information on resource use, duration, quantity, waste factor and costs (Froese and Yu 1999).
4.4 Roofing Maintenance Management Example

This example identifies typical maintenance management tasks and provides an example project. In this example, Unified Modeling Language (UML) (Oestereich 1999) was selected since EXPRESS-G, the notation used for class diagrams within the IAI, contains no notation for representing object (i.e. instance) diagrams (Froese et al. 1999). While classes represent real world items of interest through collections of attributes and relationships, instances represent specific, individual sets of values for the attributes and relationships. In particular, while the data models in figures 4.2-4.7 define the classes used in developing the roofing maintenance data models, the project scenarios in figures 4.8-4.12 define instances of these classes to represent the data used for the scenarios. Some intermediate objects have been omitted in these figures, and a few other minor simplifications have been made for clarity.

4.4.1 Identifying Roofing System Components Scenario

Figure 4.8 illustrates a UML object diagram that identifies roofing system components. In this scenario, a roof is treated as an asset, which is listed in an inventory. The roof is comprised of several roof sections. For one of the roof sections, the associated insulation, membrane, flashing, and slab objects are shown.
4.4.2 Identifying Performance Requirements scenario

Figure 4.9 illustrates a scenario for identifying performance requirements of roofing systems. The figure illustrates that one or more roof assets may have (IfcRelDefinedByType) a Roofing Condition Index (IfcPropertySet), whose value is derived from the combination of the Membrane Condition Index (IfcPropertyBoundedValue), the Flashing Condition Index (IfcPropertyBoundedValue) and the Insulation Condition Index (IfcPropertyBoundedValue). The Roofing Condition Index reflects specific roof condition requirement set “water tightness” (IfcTypeObject). Since the two classes IfcPropertySet and IfcPropertyBoundedValue do not have unique association with particular values, expressing condition indexes as values for these classes represents the author’s view in assigning values to these classes in a roofing maintenance management application.
Figure 4.9: UML diagram for identifying performance requirement of roofing systems

4.4.3 Performance Assessment Scenario

Figure 4.10 illustrates a UML object diagram for a roofing system performance assessment scenario. In this scenario, the condition of the roof section is dependent on the collective condition of the roofing system components, i.e. the condition of the roofing membrane and the condition of the roofing insulation.
Figure 4.10: UML diagram for performance assessment of roofing systems

4.4.4 Maintenance Planning Scenario

Figure 4.11 illustrates a UML object diagram for a roofing maintenance planning scenario. The scenario illustrates that a roof section may have one or more risks associated with it.
4.4.5 Maintenance Operations Management Scenario

Figure 4.12 illustrates a roofing maintenance operations scenario. The scenario illustrates that a minor repair task (IfcMRRTask) may need one or more resources for its completion.
4.5 Discussion

The efficient practice of asset maintenance management requires sharing of technical and administrative information among various computer applications, in addition to sharing information among individuals. This data sharing, in turn, demands common data standards that can enable information to be exchanged from one application to another. The research findings in this chapter focus on data representation and standards for facilities maintenance management.

The research developed a set of object models for maintenance management of built-assets, with an example application to roofing systems, to demonstrate the applicability of the proposed generic framework presented in Chapter 3. The development of the set of object models generated a robust set of data standards, considering existing standards and data representations, which can be used by all participants in a facilities maintenance management project. It should be noted that the IFC model has no data manipulation role (i.e., it does not attempt to model behavior), but rather it facilitates data interoperability between systems. The set of data model schemas were shown in EXPRESS-G graphical notation.

The framework allows incremental development of industry processes, leading to development of additional data standards. Thus allowing users to develop their own
information systems, while maintaining interoperability with each other. It is envisaged that while the developed data standards supports information exchange within the FM domain, they are capable of being re-used, hence supporting information exchange between different domain areas.

The proposed set of extensions are recommended for inclusion in Release 3.0 or later versions of the IFC model. These extensions constitute a deliverable to the FM Domain within the IAI, in which the University of British Columbia is an active member in its consortium.

The author supports the view that while the IFC model aims at representing all common information requirements of industry processes across all AEC/FM domains, such a broad scope has not, so far, provided enough FM application models to support computer integrated facilities management. There exist specific details of various FM functionalities that the IFC model has not addressed such as inspection orders, inspection tasks, inspection results, condition, and maintenance, repair, renewal tasks.

The developed set of instance diagrams within the roofing maintenance management example provides representation of the data used for the defined classes in the data models. Instance diagrams were kept in a simplistic form to enable readability. Some intermediate objects have been omitted, and data values for only three or four attributes in each object were illustrated.

A significant intellectual contribution of this work was the development of the set of extensions to the IFC model. Development of the generic IFC model for maintenance management, with specific application to roofing systems (as a type of building product that can be treated as an asset) constitutes a scope of work that is considered to be representative of maintenance management practice. In arriving at the particular set of data structures listed in Appendix B, a complete set of data and information requirements was sourced from the literature and the current state of FM practice. Comparing these sets of data and information requirements to the current IFC model-Release 2X, the author contributed the following analytical review:

1. The objective of the current IFC model is to represent all common information requirements of industry processes across all AEC/FM domains. Such a broad objective has not provided a set of classes that model the scope of a maintenance management project as described in the usage scenarios provided in Chapter 4.
2. Some FM-specific classes currently exist with the IFCs, and these were found to be generally appropriate to meet certain requirements identified in this chapter. Examples of these classes include IfcAsset and IfcInventory.
3. Some IFC classes were found to be highly generic in purpose; and can be used to model items that are relevant to FM projects, even though they had not been developed with any specific FM requirements in mind. Examples of these classes include IfcProperty and IfcPropertyDefinition to model IfcFunctionalRequirement and IfcFunctionalRequirementType, respectively.
4. Additional FM-specific functionality defined within this chapter were not found to exist with the current IFCs. This in turn has called for the development of a set
of extensions to the current IFC model to fully address all representative issues in a maintenance management project.

Therefore, additional IFCs representing physical project data (such as IfcRoofSection, IfcMembrane, IfcFlashing) and non-physical project data (such as IfcInspectionOrder, IfcInspectionTask, IfcMRRTask, etc.) were developed to cover the full scope of a maintenance management project as described within this dissertation. Other physical project data would be added if the scope of work were expended to other types of building products that can be treated as assets. It should also be noted that the IFCs do not necessitate that all required classes and attributes be modeled explicitly in the IFCs. Much of the low-level detail would be represented using dynamically configurable property set mechanisms and product libraries.

While there is no expectation that the proposed model is the only—or even the best—model possible to satisfy the requirements, the use of appropriate development methodologies is the first step in assuring that the model offers a good solution. The prototype implementation described in the following chapters provides additional validation that the proposed model serves its purpose adequately. If the proposed model were to be adopted by the IAI as extensions to the IFCs, the model would receive additional vetting and validation by committees of domain experts, through implementation by commercial software developers, by end users of software systems, and then the entire cycle would be repeated in future release cycles. These validation mechanisms, beyond the initial prototype, are beyond the scope of this dissertation.

The next chapter explores the implementation of these models through the development of an Asset Management Tool prototype integrated application, as a proof-of-concept for implementing the view of generic asset management in a distributed, model-based, integrated systems architectures environment, named the Jigsaw Distributed System.
Chapter 5
Distributed Systems and Architecture

Chapter Abstract: This chapter explores the implementation of the developed maintenance management models through the development of the prototype Asset Management Tool application, a distributed, model-based integrated system. It describes the set of inter-connecting components forming a typical or reference system architecture for integrated distributed systems. It describes the Jigsaw Distributed System version 0.6, as the implementation environment of the reference architecture, which facilitates a wide range of data exchanges and software interoperability.

The objective of the prototype application is to provide proof of concept for implementing a generic asset management in the context of distributed, model-based, integrated AEC/FM system architecture. To achieve this, an Asset Management application was developed as one component of a larger ongoing research project to develop a suite of integrated AEC tools along with the necessary IT infrastructure to deliver distributed, model-based integrated systems for the AEC/FM industry. The overall distributed system, named the Jigsaw Distributed System (JDS, version 0.6) is described in this chapter. Although the Asset Management Tool has contributed to the distributed systems research, this is not the focus of this dissertation, and the work is described here as context for the Asset Management tool rather than as a major contribution of this dissertation. The Asset Management Tool is then described in Chapter 6.

5.1 The Jigsaw Distributed System

The Jigsaw Distributed System, developed through an ongoing research project led by Professor Thomas Froese, Construction Engineering and Management Group at the University of British Columbia (JDS 2001), aims at developing tools to support both project model-based data integration and transaction-based application interoperability. The Jigsaw Distributed System implements the ideas of the general reference architecture (described later in section 5.2 within this chapter) by creating a system in which a wide variety of data clients can communicate with a wide variety of data servers through a standard Application Programming Interface, so that individual data clients need not know the details of the individual data servers and vice-versa.

5.1.1 Data Clients

Data client applications interact and initiate data exchanges with data server applications through a number of data import/export operations, typically initiated from within the User Interface (UI) of that of the data client application. Data client applications
typically can be custom-built applications, or current, desktop applications, referred to as "legacy applications" that work with their own application data. Data exchanges within the Jigsaw Distributed System are facilitated through abstract components called "adaptors". Adaptors map the schema of the data client application to the Jigsaw data interface. Current custom-built data client applications within the Jigsaw Distributed System include the following:

1. Asset Management Tool, referred to as "JsAMT": focusing on generic asset management and maintenance planning,
2. CAD Tool, referred to as JsCAD: focusing on providing CAD based access to product and process information,
3. Materials Management Tool, referred to as JsMMT: focusing on procurement and handling of materials,
4. PECAD: focusing on cost estimating through Timberline Estimating software,
5. Information Browser, referred to as JsBrowser: focusing on providing general browsing capability for Jigsaw data and linking to external information,
6. Data Transporter, referred to as JsDataTransporter: focusing on copying data between any two data servers.

5.1.2 Jigsaw Interface

The standardized Jigsaw interface uses many "industry standard" elements, namely, Extensible Markup Language (XML) and the XML Document Object Model (DOM) for the "data content" Application Programming Interface, and the IFCs for much of the data content schema. Jigsaw adds some custom elements for the data access and control Application Programming Interface, and some extensions to the IFC schemas. The Jigsaw data server Application Programming Interface is called JsServerDOM. The JsServerDOM is an "abstract" component: it defines the interface for all Jigsaw data server components.

As Figure 5.1 illustrates in a typical Jigsaw situation, a data client may work with its own client data, but it uses the JsServerDOM interface for all interaction with other applications or data sources. These data requests may be for full models or for specific pieces of information. The Jigsaw data server component provides a "wrapper" around some other form of data source, adapting it to the JsServerDOM Application Programming Interface.

![Figure 5.1: Jigsaw Distributed System client-server interaction](image-url)
5.1.3 Data Servers

Current types of data servers as illustrated schematically in Figure 5.2, for which the standard Jigsaw data interface can respond to information requests (data exchanges) for either full data models or specific pieces of information, include: XML files, Building Lifecycle Interoperable Software (BLIS) IFC data files, other legacy applications such as Microsoft Project and MicroROOFER, relational databases, and remote data sources accessed over the World Wide Web.

![Diagram of data servers in the Jigsaw Distributed System](image)

Figure 5.2: Current types of data servers in the Jigsaw Distributed System

5.2 A Tiered Reference Architecture

Figure 5.3 illustrates a typical set of software components for distributed, model-based, integrated systems for AEC/FM (Froese et al. 2000). These components describe a typical, or reference architecture. A range of variation is feasible. One possible configuration of where these logical tiers may reside consists of two or more physical computer systems, a user’s workstation and one or more central servers. The listing below describes how the architecture is organized into three logical tiers. It also describes the system components within each tier as:

1. The application or presentation tier: contains application programs and related user-centric components. This dissertation is focusing on implementation work within the application tier, where the prototype Asset Management Tool represents a custom-built application that has the capability to import and export data files from other legacy applications. The Asset Management Tool was developed using Microsoft Visual Basic 6.0.
In the application or presentation tier, both custom-built and legacy system applications maintain their own data sets in addition to the information shared through the integrated system. Adaptors, as illustrated in Figure 5.3, are pieces of code that serve to map application schema to the common data schema. This mapping is carried out by an application-specific adaptor software component.

2. The business objects or middle tier: brings the data and services of the distributed system to the local applications and implements business logic processes. In this configuration, local model proxies are software components on users' workstations. Applications and adaptors access a local version of the shared data that acts as a proxy for remote data sources. These components expose the distributed information services to client applications, and handle the communication of the local data within the distributed servers.

3. The data tier: handles the persistence of project model data. While there can be various centralized or distributed data repository alternatives, a typical configuration involves a central database and data server component that interacts with model proxy components across the Internet/Intranet connection.

Figure 5.3: A reference architecture for a distributed, model-based, integrated system (Froese et al. 2000)
5.3 Asset Management Tool, A Prototype Integrated Application

The Asset Management Tool is a prototype application developed to provide proof-of-concept for implementing the view of generic asset management in a context of distributed, model-based, integrated AEC/FM systems environments. The Asset Management Tool is a custom-built application implemented as a Jigsaw client, whose implementation is situated within the application or presentation tier of the reference-system architecture described in section 5.2. Figure 5.4 illustrates the implementation model for the Asset Management Tool prototype application in UML notation. The figure illustrates the interaction of data client software components (Jigsaw Asset Management Tool and Jigsaw Application Objects) with data server software components (Jigsaw XML file, Jigsaw BLIS data source, Jigsaw MicroROOFER data source and Jigsaw Microsoft Project data source) through a standard Jigsaw data interface. The listing in the following subsections provides a description of the data client and data server software components involved in the implementation of the Asset Management Tool prototype application:

![Implementation Model Diagram](image)

Figure 5.4: Asset Management Tool’s implementation model

5.3.1 Data Client Software Components

The two main data client components of the prototype application include the user interface components of the Jigsaw Asset Management Tool and the set of Application Objects for the implemented schema.
5.3.1.1 Jigsaw Asset Management Tool

The Jigsaw Asset Management Tool is a custom-built application consisting of Visual Basic user interface forms and code through which the user interacts with project data and initiates a wide range of data exchanges. Chapter 6 provides a detailed description of the user interface components and functional requirements for the Asset Management Tool prototype application.

5.3.1.2 Jigsaw Application Objects

The Jigsaw Application Objects “JsAppObjs06” component is a dynamic, customized data model component, specific to all data clients implemented within the Jigsaw Distributed System. The Jigsaw Application Objects consist of Microsoft Visual Basic classes that implement a collective schema encompassing all objects necessary for all data clients in the Jigsaw Distributed System. These Microsoft Visual Basic classes are generated by the Jigsaw Modeling Tool (JMT) where the set of specific IFCs supporting the asset maintenance management data model, among other IFCs supporting other data models intended for implementation within the Jigsaw Distributed System, was entered and edited through its user interface. The Jigsaw Modeling Tool was developed by Professor Thomas Froese, Construction Engineering and Management Group at the University of British Columbia. It is a utility for entering and editing data models (schemas) that represent a business object component within the reference architecture described in section 5.2. Since the Jigsaw Modeling Tool can import and export various forms of models (i.e., models with different meta-models), it acts as a "meta-meta model". Figure 5.5 illustrates a snapshot of the collective IFC schema for all data client components, defined and edited using the Jigsaw Modeling Tool’s user interface.
In Figure 5.5, the schema is displayed through an explorer-style interface that provides the user with the visual ability to browse through individual classes within the IFC schema. The user can view the properties of a particular class in the explorer-style interface through either expanding the node of the class, or double-clicking on that class. The user can also view and/or edit the entity name, description, type, etc. on the tabs view to the right of the explorer-style interface. Figure 5.6 illustrates the operation of creating Jigsaw application objects (Microsoft Visual Basic code) for all objects defined within the data model.
5.3.2 Data Server Software Components

In addition to the existing data server components (including XML files, BLIS data sources, MicroROOFER data sources and Microsoft Project data sources), the Asset Management Tool prototype application is capable of exchanging data with any other application that works with a similar information structure through the standard Jigsaw interface.

5.3.2.1 Jigsaw Server Document Object Model

The Jigsaw Server Document Object Model “JsServerDOM” is an “abstract” component that defines the interface for all Jigsaw data server components. The interface can be divided into four layers:

1. **Transport:** the data content of the transaction is contained in XML format. Since this is a program interface rather than a file format, the form of the XML is the XML interface, the XML Document Object Model.
2. **Encoding:** a custom approach is used for encoding object based data in XML, similar to BLIS-XML and IFC-XML mappings. IFC-based Object Data is mapped into XML as follows:
   a. The root element is a “JsCollection” element containing a list of all objects.
   b. Each object is represented as an element named after the object class.
   c. Each attribute is represented as a child element of the object element, named after the attribute name.
   d. If an attribute is not a collection or a reference to another object, its value is simply the text of the attribute element.
   e. If an attribute is a collection, then the value of the attribute element is a child element of type “JsCollection”.
   f. If an attribute is an object reference, then the value is a child element of type “JsObjRef”.

Figure 5.6: Creating Jigsaw application objects
g. At present, collections can only contain ObjRef data types.
h. Each object includes an attribute called “ObjRef” which contains a “JsObjRef” element. This is the ID for the object.

3. **Schema:** the schema used is IFC Release 2X in addition to the extensions developed within this dissertation.

4. **Data Management:** this layer includes a series of customized methods such as:
   a. Establishing connection information from a data client to a data server.
   b. Opening and closing connection from a data client to a data source.
   c. Loading specified objects from the data source into the Document Object Model.
   d. Beginning a transaction, which occurs within an open connection.
   e. Aborting a transaction by restoring the Document Object Model to the state that it was at the beginning of the transaction.
   f. Committing a transaction by saving the Document Object Model to the data source.

5.3.2.2 Jigsaw XML File

The JsXMLFile0.6 data server component can read or write data as XML files. In importing XML files, information that is supported through the application’s schema is imported, while unsupported information is simply ignored.

5.3.2.3 Jigsaw BLIS Data Source

The JsBLISDS0.6 data server component can read and write from IFC 2.0 data files as used by software applications from the BLIS project. Building Lifecycle Interoperable Software (BLIS) is a group of software developers who are members in the consortium of the IAI, with the mission to develop interoperable prototype software to validate the use of the IFCs (BLIS 2001). At present IFC 2.0 data sources undergo a partial mapping to the IFC 2X schema.

5.3.2.4 Jigsaw MicroROOFER Data Source

The JsRooferDS0.6 data server allows the user to import and export roof building product objects from the MicroROOFER application. This data source component has been developed specifically for facilitating data import and export with the prototype Asset Management Tool application.

5.3.2.5 Jigsaw Microsoft Project Data Source

The JsMSProjectDS0.6 data server component allows the user to import and export tasks information, such as task name, duration, start date, finish date, predecessor and successor tasks from Microsoft project files. This data source component has been
developed specifically for facilitating data import and export with the prototype Asset Management Tool application.

5.4 Discussion

This chapter presented the distributed system architecture underlying the prototype Asset Management Tool application through describing its sets of data clients, data servers and the standard data interface between them. It described the set of inter-connecting components forming a reference architecture for integrated distributed systems, upon which the Jigsaw Distributed System is based. The Jigsaw Distributed System is an ongoing research project aimed at developing tools to support both project model-based data integration and transaction-based application interoperability. The Jigsaw system is presented here not as a contribution of this dissertation, but rather to provide the background material for the implementation of a FM process in the context of integrated systems.

The next chapter describes the components of the Asset Management Tool prototype application itself. It addresses the set of functional requirements and the implementation environment of the user interface. It also presents an evaluation and testing scenario for the prototype application that demonstrates software interoperability through a wide range of data exchanges between applications.
Chapter 6
Implementation and Evaluation

Chapter Abstract: This chapter describes the development of a generic Asset Management Tool prototype data client application that can initiate data exchanges with a number of data servers (including MicroROOFER, Microsoft Project, and files that can be used by several other applications), thus demonstrating software interoperability in the FM domain. The Asset Management Tool prototype application served as a validation for the structure of the developed process and data models. The chapter starts by describing the components of the user interface and the specific set of actions through which the user interacts with the prototype application. The chapter then presents the set of functional requirements and the implementation environment of the user interface. Finally, the chapter presents an evaluation and testing scenario for the prototype application.

The contribution made in this chapter involves the implementation of a generic Asset Management Tool prototype data client application that can initiate data exchanges with a number of data servers in a FM application domain. Such an approach to software implementation offers application users and domain specialists the benefit of relating multiple kinds and sources of information in a project file, thus facilitating software interoperability. The Asset Management Tool prototype application is one of the data client components in the Jigsaw Distributed System. The prototype development is an embodiment of the distributed, model-based systems architecture concepts that were discussed in Chapter 5. The generic view in the application stems from designing it with the capability to process data on any building product that can be treated as an asset that has financial value, and against which MRR tasks are applied. The programming Visual Basic code is written specifically for the Asset Management Tool with the assistance of the research supervisor Professor Thomas Froese.

While the original intention of the research was to implement an application that embodies all management processes identified through the maintenance management framework presented in Chapter 3, a decision was made to limit the scope of the implementation because of the amount of programming effort involved. The prototype application can interoperate with a number of specific applications, allowing data exchanges and facilitating the development of an inventory list of assets and the creation of MRR tasks. The prototype itself is not intended to provide a greater functionality than existing FM software. Rather, it is intended to demonstrate the nature of software interoperability that is relevant and useful for maintenance management, as well as to provide some testing and validation of the process models and data models described in this dissertation. A detailed description of the prototype capabilities is presented in the evaluation and testing scenario section of this chapter.
6.1 Components of User interface

An integral part of the developed Asset Management Tool prototype application is the user interface, through which the user interacts with the project data. Two main user interface components have been used to provide the required functionality for the user to browse, input and edit project information. These components are an explorer “Tree View” and Multiple-Document Interface forms.

6.1.1 Tree View

The explorer-style interface “Tree view” is a user interface component that provides the application user with the visual ability to locate a project’s data records in potentially large data collections. The user can view multiple copies of the explorer at the same instance of the Asset Management Tool application. This allows the user flexibility in coping and/or moving sections of the explorer (branches of the tree) through a Drag and Drop method in the same explorer or from one explorer to another as displayed in Figure 6.1. This type of user interface lends itself well to navigating or browsing large numbers of data records in a project. Figure 6.1 is a screen image of the Asset Management Tool’s user interface, illustrating the explorer-style interface “tree view” that provides a visual display of the hierarchy of a single project in the database and the association between various different objects in the hierarchy. In essence, the explorer “Asset Management Tree View” brings up a view of the implemented data model for view by the user, thus providing the user with a multi-dimensional view of the data model and the instantiated objects in the database.
The two icons and are used in the project hierarchy to denote "Objects" and "Collections" respectively. Working with extensive project information through a hierarchical tree view interface, a new project has the following object collections:

1. Products: every product object in the "Products" collection may have association with the following two object collections, as illustrated in Figure 6.2:
   a. Assets: a collection of asset objects that can be treated as building products through the objectified relationship "IfcRelAssignsToGroup".
   b. Tasks: a collection of task objects that can be associated with products through the objectified relationship "IfcRelAssignsToProcess".

Figure 6.1: Asset Management Tool’s interface
2. Assets: every asset object in the “Assets” collection may have association with the following two object collections, as illustrated in Figure 6.3:
   a. Products: a collection of product objects that can be treated as assets through the objectified relationship “IfcRelAssignsToGroup”.
   b. Tasks: a collection of task objects that can be associated with assets through the objectified relationship “IfcRelAssignsToProcess”.

3. Asset Types: this object collection is not currently implemented in the Asset Management Tool.
4. Inspection Orders: this object collection is not currently implemented in the Asset Management Tool.
5. Work Orders: this object collection is not currently implemented in the Asset Management Tool.

6. Tasks: every task object in the "Tasks" collection may have association with the following three object collections, as illustrated in Figure 6.4:
   a. Objects operated on: a collection of objects that a particular task operates on through the objectified relationship "IfcRelAssignsToProcess".
   b. Predecessors: a collection of predecessor task objects obtained through the objectified relationship "IfcRelSequence".
   c. Successors: a collection of successor task objects obtained through the objectified relationship "IfcRelSequence".

Data values for each object are input in the database through choosing the corresponding menu item from the "Insert" pull-down menu in the main menu as in Figure 6.5. The main menu displays several menu items, corresponding to the processes of the asset maintenance management framework, described in Chapter 3, and illustrated in Figure 3.1 of this dissertation. The Asset Management Tool application is created using Microsoft Visual Basic 6.0.
6.1.2 Multiple-Document Interface Forms

Data for each object in the hierarchy may be edited in the database through calling the appropriate Multiple-Document Interface form by double clicking on the appropriate object within its collection in the explorer. The Multiple-Document Interface forms-style interface allows the user to display multiple forms at the same time; each form is displayed in its own window.

6.2 Functional Requirements of User Interface

This section describes the specific set of actions through which the user interacts with the prototype application. These actions are summarized as browsing, creating and/or editing object data, and importing/exporting data from and to other data sources.

6.2.1 Browsing

Browsing refers to the user’s ability to view structured project information through a hierarchy, view details of project information, and view the relationship between objects. An implemented explorer-style interface “tree view” seems very effective in organizing records of project data into collections of objects data, as illustrated in Figures 6.1 through 6.3. Having an explorer-style interface “tree view” permits the user to visually confirm the actions of adding and editing data records (object nodes) in the project file. The explorer can be collapsed or expanded for the purpose of listing data records, accessing details, or both simultaneously.

6.2.2 Creating and Editing Object Data

Creating and editing object data refers to the user’s ability to easily input, add, and modify data records belonging to the appropriate collection in the project hierarchy. Users can create new data record by making the appropriate selection from the “Insert” pull down menu to input data values of the new object in an Multiple-Document Interface form. The new object simultaneously appears under the appropriate collection in the explorer. A double click on an object node in the explorer brings up the appropriate Multiple-Document Interface data entry form for that object to allow the user to view and modify its attributes.

6.2.2.1 Creating/Editing a Project Object

In the event of creating a new project and populate the property of the class IfcProject, the user needs to choose the “Project” item from the “Insert” pull down menu. A “Project” dialog box then appears on the screen, prompting the user to enter the name of the project file as in Figure 6.6. A double click on the project’s object node in the explorer brings up the appropriate Multiple-Document Interface data entry form for that object to modify its name if desired by the user.
After the user has named the newly created project file, the project name is updated in the explorer “Asset Management Tree view”, illustrated in Figure 6.1, which also appears instantly when the user chooses the “Project” item from the “Insert” menu.

6.2.2.2 Creating/Editing a Product Object

To create a new product, the user chooses the “Product” item from the “Insert” pull-down menu shown in Figure 6.5. A “Select Product Type” dialog box then appears on the screen, prompting the user to select the type of the product from a pre-defined list of product types, as illustrated Figure 6.7.

After selecting a product type, a display of the new product appears within the “Products” collection in the “Asset Management Tree View”. A double click on the product’s object node in the explorer opens the appropriate Multiple-Document Interface data entry form for that product allowing the user to modify its name if desired, as illustrated in Figure 6.8.
6.2.2.3 Creating/Editing an Asset Object

To create a new asset and populate the properties of the class IfcAsset, the user chooses the “Asset” item from the “Insert” menu shown in Figure 6.5. A data entry form then appears on the screen, prompting the user to record various information related to the new asset as in Figure 6.9. Upon executing the data entry action for the new asset and pressing the “OK” button, a display of the new asset appears within the “Assets” collection in the “Asset Management Tree View”. A double click on the asset’s object node in the explorer brings up the appropriate Multiple-Document Interface data entry form for the asset object to modify its data values if desired by the user.

6.2.2.4 Creating/Editing a Task Object

To create a new task and populate the properties of the class IfcTask, the user chooses the “Task” item from the “Insert” menu. A form then appears on the screen, prompting the
user to record information related to the new task as in Figure 6.10. Upon executing the data entry action for the new task, a display of the new task appears within the “Tasks” collection in the “Asset Management Tree View”. A double click on the task’s object node in the explorer brings up the appropriate Multiple-Document Interface data entry form for the task object to modify its data values if desired by the user.

![Task Data Entry Form](image)

Figure 6.10: Task data entry form

Creating new “Asset Type”, “Inspection Order” and “Work order”, and displaying them under the appropriate collection in the “Asset Management Tree View” are not currently implemented within the Asset Management Tool.

### 6.2.3 Importing/Exporting Data from and to Data Servers

Importing/exporting refers to the user’s ability to bring in and send off data records from and to data servers. The Asset Management Tool can currently import and export data records from and to four types of data sources within the Jigsaw Distributed System. These four types of data sources are as follows:

1. XML file data server, referred to as JsXMLFile06.
2. Ifc2.0 file data server, referred to as JsBLISDS06.
3. MicroROOFER database data server, referred to as JsRooferDS06.
4. Microsoft Project file data server, referred to JsMSProjDS06.

Not only can Asset Management Tool interact with the applications for which data server adaptors have been created (MicroROOFER and Microsoft Project), but it can also interact with any other applications that can work with any of these data sources, such as programs from the BLIS group that can read and write Ifc2.0 files (BLIS 2001).

In the event of importing data from any of the currently implemented data sources within the Jigsaw system, the user needs to select the data source type by choosing “Connection Type” from the “File” menu. A “Connection Information” dialog box then appears on the
screen, prompting the user to select the data source type from the above pre-defined list of data sources, as illustrated in Figure 6.11.

![Connection Information Dialog](image)

Figure 6.11: Connection type dialog box

After setting up the connection to a particular data source type and selecting “Open” from the “File” menu, the user chooses to either replace the currently open project by choosing “Yes” or add contents to the currently open project by choosing “No”.

The user then is prompted to browse a list of data files of the same data type to select a data source (e.g., a file) to import into the currently open project in the Asset Management Tool application, as illustrated in the “Set Connection” dialog box in Figure 6.12.

![Set Connection](image)

Figure 6.12: Set connection dialog box

Figure 6.13 prompts the user to select an XML file from a list of XML files to import its contents into the currently open project in the Asset Management Tool. Similarly, the user can save an Asset Management Tool project file as an XML file for read/write purposes.
Figure 6.13: Dialog box for opening XML data files for read/write

Figure 6.14 illustrates a log report generated from importing a Microsoft project data file. The log confirms completion of the import/export operation. It displays a list of execution general-purpose messages, warnings and errors, if any. The user can also modify the log options for what to log, how to log and where to save the log file, through choosing the "Option" button on the log form and making the appropriate selections in the "Log Options" form, as illustrated in Figure 6.15.

Figure 6.14: Import/Export data log form
6.3 Evaluations and Testing Scenario

Project-information examples were used to serve dual purposes. The first was testing the user interface browsing and editing capabilities. The second was validating the structure of the developed asset management data model. The scenario below presents a progression of steps that describes how the Asset Management Tool can be used to demonstrate software interoperability. As mentioned earlier, the focus of the research was not to develop software with features already available in commercially available software. The focus was rather to implement a distributed model-based, integrated prototype application that combines building product and process information and is capable of data exchange, based on using the internationally recognized IFC data standards to represent project information.

6.3.1 Starting from Scratch

Starting with an empty project, the user starts with populating the “Products” collection in the Asset Management Tool application with roof objects as appropriate, through choosing the “Insert” menu and selecting the “Product” menu item.

The user may also choose to import roof product objects into the “Products” collection of the Asset Management Tool application from one or a combination of data sources.
Moreover, the user may also import roof product data records in an XML file format from a currently developed CAD application, acting as another data client component within the Jigsaw system. The Jigsaw Distributed System project includes a CAD application referred to as JsCAD. Figure 6.16 illustrates schematically the data client and data server components involved in populating the Asset Management Tool application with roof product objects from scratch.

![Diagram](image)

Figure 6.16: Populating the Asset Management Tool application with roof product objects from scratch

### 6.3.2 Creating Association Relationships between Objects

The user may then populate the “Assets” collection in the Asset Management Tool application with asset objects by selecting “Asset” from the “Insert” menu. The intent here is to define generic data values, entered in the “Asset” data entry form, for the “Product” object that is being treated as an asset. An association relationship between the two objects (a roof object and an asset object) can then be established within the project hierarchy. This is facilitated through dragging the roof product object from its “Products” collection and dropping it on the appropriate asset object in the “Assets” collection. Figure 6.17 illustrates associating “Roof 14” product object within the “Products” collection to “Asset 1101 – Roof” asset object in the “Assets” collection.
The user may also create an asset object from a particular roof product object. This is facilitated through choosing the “Assets” pull-down menu and selecting “Create Assets from Selected Products” menu item after placing a check mark in the adjacent box to the appropriate roof object. Figure 6.18 illustrates creating “Roof 25 Asset” asset object in the “Assets” collection from “Roof 25” product object within the “Products” collection.
6.3.3 Exporting Data to MicroROOFER Application

The user may then export the three roof product objects, “Roof 14”, “Roof 25” and “Roof 32” from the Asset Management Tool as a data client component to MicroROOFER as a data server component within the Jigsaw system. The three exported roof product objects are treated as three distinct roof sections in the MicroROOFER application. Mapping a roof product object in the Asset Management Tool application to a roof section in MicroROOFER application is facilitated by an adapter component, developed within the Jigsaw system, to map the schema used in the Asset Management Tool application to that of MicroROOFER.

The process of exporting the three roof product objects from the Asset Management Tool application to the MicroROOFER application starts with setting the data source type to “JsRooferDS06”. The user is then prompted to save the project file by either replacing a file for an existing project, or add the project file to an existing project file. At the time of composing this evaluation and testing scenario, exporting roof product objects to the MicroROOFER application has the limitation of adding the newly exported objects as new roof sections to an existing building project database.

A selection form then appears, prompting the user to choose the building project for which the three roof product objects are appended to its roof sections. In this scenario, the three roof product objects were added to “M-16-Graphics & Illustrative Services” building project as illustrated in Figure 6.19.

Figure 6.19: Project selection form
Exporting the three roof product objects to the MicroROOFER application and treating them as distinct roof sections permits the user to record condition assessment data within MicroROOFER. Figure 6.20 illustrates the addition of the three roof objects, each treated as a distinct roof section to the already existing two roof sections in building “M-16 Graphics & Illustration Services” within the MicroROOFER application.

Figure 6.20: Addition of three new roof objects to existing roof section in MicroROOFER application

6.3.4 Importing / Exporting Data to Microsoft Project Application

After the condition assessment data has been input for “Roof 25” product object by MicroROOFER, and for the purpose of composing this scenario, it is found that the roof condition index of “Roof 25” product object is 24 (out of a possible 100). The MRR option that corresponds with this condition according to MicroROOFER’s set of MRR options is “Replace Roof” as illustrated in Table 3.1: MRR options corresponding to values of roofing condition index. The user of the Asset Management Tool application may then specify the set of tasks that need to be performed under the “Tasks” collection within the project hierarchy. This is facilitated through either:
1. The user selecting “Task” from the “Insert” menu in the Asset Management Tool application. The user can then export this set of tasks to Microsoft Project to create a schedule, or

2. The user specifying the set of tasks within Microsoft Project and then importing them into the Asset Management Tool application.

Importing and exporting from and to Microsoft Project is facilitated by an adapter component developed within the Jigsaw system to map the schema used in the Asset Management Tool application to that of Microsoft Project.

Figure 6.21 illustrates the export data log form for the set of tasks as they were imported from Microsoft Project into the Asset Management Tool application.

After importing tasks from Microsoft Project, the user may examine each task to browse a list of product and asset objects that a task operates on, a list of predecessor tasks, and list of successor tasks. Figure 6.22 illustrates an overview of the Asset Management Tool’s user interface populated with data for a sample project file “project 327”. The figure illustrates that the task object “Remove old thermal insulation board” operates on the “Roof 14” product object, which has been treated as an asset object with the name “Asset 1101-Roof”. The task object “Remove old roofing membrane felt” is the predecessor task, and the task “Remove old vapor barrier” is the successor task. The figure also illustrates the class names for the instantiated objects within the project hierarchy in the explorer-style interface “Asset Management Tree View”, which can be turned on and off through selecting “Show Class Names” menu item from “View” menu.
6.4 Discussion

The Asset Management Tool prototype application was designed and implemented as a generic prototype application for the purpose of demonstrating software interoperability through the use of the internationally recognized IFC data standards to represent project information. The application is not limited to roof products only. Similar scenarios can be generated for demonstrating data exchange for diverse sets of building products. The prototype application developed addressed the user interface’s functional requirements of the integrated Asset Management Tool. Application development, with proof of concept in mind, was facilitated using Microsoft Visual Basic, a common software development tool.

The evaluation and testing scenario demonstrates the operations that the Asset Management Tool supports. In summary, the user may:
2. Also import roof product objects into the “Products” collection of the Asset Management Tool application from one or combination of data server components within the Jigsaw Distributed System.

3. Populate the “Assets” collection with asset objects from within the Asset Management application.

4. Create association relationships between “Assets” and “Products” objects to identify the building products that can be treated as assets, for the purpose of compiling an inventory list of assets.

5. Export roof product objects from the Asset Management Tool application as a data client component to MicroROOFER as a data server component within the Jigsaw Distributed System. This export capability permits the user to record condition assessment data for the exported roof product objects within MicroROOFER and obtain a roof condition index value (out of a possible 100). MicroROOFER then recommends a MRR option that corresponds to the value of the roof condition index. On a separate note, the adopter component to MicroROOFER application, which has been developed within the Jigsaw Distributed System allows the user to import the values for the condition indexes into the Asset Management Tool application. However, the user interface components that allow the user to view the imported condition index values were not developed at the time of writing this dissertation.

6. Specify the set of tasks that need to be performed in accordance with the recommended MRR option under the “Tasks” collection within the Asset Management Tool application. The user then may export these tasks, with their durations and sequencing logic to Microsoft Project to develop a schedule for a MRR operation.

7. Equally import the set of tasks that need to be performed in accordance with the recommended MRR option under the “Tasks” collection from Microsoft Project as a data server component into the Asset Management Tool as a data client component. The imported tasks already have their durations and sequencing logic established after using Microsoft Project to develop a schedule for a MRR operation.

Execution of these operations demonstrates software interoperability by the prototype Asset Management Tool, and the successful implementation of the current state-of-the-art IFCs in the FM application domain.

Building up a project’s information from scratch evaluated the developed asset maintenance management data model in capturing project information through populating a new project’s object hierarchy, and tested the ability of the user interface to support the input process and representation of information. The user interface adequately supports exporting parts of the project information to other legacy applications such as MicroROOFER and Microsoft Project. The user interface also adequately supports modifying the contents of an existing project within the Asset Management Tool application through merging existing data in a project with imported data from multiple data servers. The content of the newly modified project information is delivered using XML datasets. This in turn makes the new project information file accessible to:
1. Other applications (within the Jigsaw Distributed System), such as the CAD tool, the Information Browser and the Data Transporter; that use IFC data standards-compliant schemas to represent project information.

2. Other research prototype applications developed within the BLIS group of the IAI (BLIS 2001).

3. Other legacy applications, through the development of multiple adapter components that map the schema of the Asset Management Tool to that of the appropriate legacy application.

Limiting the scope of the implementation, the "Asset Types", "Inspection Orders" and "Work Orders" collections within a project's object hierarchy were not implemented due to time constraints. Under more favorable time conditions, these object collections would not have been technically problematic to implement.

Finally, the developed prototype is an embodiment of the distributed, model-based, integrated systems architectures. The next chapter concludes this dissertation and highlights the contributions made by this research.
Chapter 7
Contributions and Conclusions

Chapter Abstract: This chapter concludes this dissertation. It summarizes the research findings and development areas resulting from the pursuit of the research objectives. The chapter lists the research contributions and the deliverables made to the appropriate societies and the FM community at large. Finally, the chapter discusses future research topics that are based on this dissertation and hold significant potential to improve the FM industry.

7.1 Summary

This dissertation discusses research in two themes. The first being data interoperability in the asset/facility management industry and the ability to inter-communicate project information among various users and diverse sets of computer applications. The second being the implementation of a tool developed as a proof of concept for the idea of data interoperability among users and applications. For each of these themes, this dissertation identifies and develops supporting areas of knowledge and technology.

The research contributed a description of relevant knowledge areas in the domain of asset management. The research also reviewed previous and current research directions on product modeling within the asset management domain, as an enabling technology, holding great potential for improving methodologies and computer tools. The research analyzed technologies and development tools currently in use within the asset management industry.

Working towards proving the first principle research hypothesis “FM knowledge can be formalized in a manner similar to that of project and construction management knowledge”, the research contributed a generic framework for asset maintenance management. It describes a unification or collection of diverse knowledge areas that have been analyzed, with particular specialization to the domain of asset maintenance management. The framework serves to standardize process descriptions, the activities that need to be undertaken, and the methodology of how and what information needs to be communicated between activities. The framework has been presented in the dissertation in the form of an IDEF0 process model. The process model served to illustrate the interaction and dependencies among diverse set of knowledge areas. Outputs from one management process become inputs to another in a subsequent hierarchy. The structure of the framework exhibited the characteristics of flexibility and robustness. Updates in knowledge can be accommodated through incorporating new management processes and/or activities, as well as establishing new sequencing logic for these processes and/or activities.
Proving the second principle research hypothesis “The development of data standards facilitates the integration of data and knowledge of FM projects”, the research contributed the development of an asset maintenance management project domain model. The model constitutes a combination of product and process views of the product being maintained. The model represents a schema or data structures for representing both physical AEC/FM project data such as representing the components of the product being modeled, and non-physical AEC/FM project management data, such as representing performance requirements, inspection orders, and planning and scheduling data. The research has chosen to adopt a similar methodology to that of the IAI to develop the data standards in the form of IFCs. The model, while making use of the existing IFCs within Releases 2.0 and 2X of the IFC model, develops and proposes a set of IFCs that is recommended for inclusion within Release 3.0 or later versions.

The third principle research hypothesis “Interoperable software that integrates data and knowledge can be developed for FM applications” was proved through contributing the implementation of the Asset Management Tool prototype application. The application was developed as an embodiment of the distributed, model-based systems architecture concepts. The Asset Management Tool represents a data client component that can interact and exchange data with a number of possible data server components (legacy applications, including Microsoft Project and MicroROOFER) through a shared, model-based project database. The project database is structured according to a schema based on IFC Release 2X. The Asset Management Tool delivers the capability to import and export (read and write object data) IFC information from, and to, other legacy applications, thus facilitating data exchange and achieving software interoperability. Such capability holds the promise of significantly impacting AEC/FM industries through communicating the critical relationships between project information and greatly reducing the dependency on paper-based views of project information.

Overall, the research investigated and proved the series of principle hypotheses listed in Section 1.3, upon which the research objectives have been formulated.

7.2 Research Contributions and Deliverables

The following sub-sections summarize the primary contributions of this research:

7.2.1 Contribution to Facility Management Practice

The research contributed a framework that aims at being a systematic and generic reference description to the practice of asset maintenance management, as described in Chapter 3. The framework served to identify avenues throughout the practice of asset management where effective strategies and procedures (integration of data and knowledge through conceptual models) could be implemented to improve operations and sustain the capital investment in an asset through its life cycle. Formulation and development of the proposed framework constitutes a deliverable to the consortium of the BELCAM project, in which the University of British Columbia is a member.
7.2.2 Contribution to Development of Domain Models

The research contributed a general object-oriented domain model or schema for asset maintenance management, as described in Chapter 4. The research proposed a set of extensions to the IFCs recommended for inclusion within Release 3.0 or later versions of the IFC model. While the developed schema for asset maintenance management supports information exchange within the FM domain, it is capable of being re-used in other domains, hence supporting information exchange and demonstrating software interoperability among different domain areas, as described in Chapters 5 and 6. The development of the asset management schema and the new set of extensions to the IFCs, constitutes a deliverable to the FM Domain within the IAI consortium, in which the University of British Columbia is a member.

7.2.3 Contribution to Development of Integrated Applications

The research contributed the development of the Asset Management Tool, a prototype integrated application that combines building product and process information; and is capable of importing and exporting IFC information. The prototype implements the developed process and data models in the context of distributed, model-based, integrated AEC/FM system architecture, as described in Chapters 5 and 6. The development of the Asset Management Tool constitutes a deliverable to the consortium of the collaborative Information Technologies for AEC/FM in Canada (ITAC) project, in which the University of British Columbia is a member. While the ITAC project has identified a number of collaborative subprojects within the project’s proposal, the specific subproject pertaining to this deliverable is the “Integrated system implementation in Facilities Management”.

The following item is also a contribution of a secondary nature:

7.2.4 Contribution to Domain Analysis and Description

The research contributed a succinct review of knowledge areas relevant to the domain of asset/facilities management, scattered across a range of publications. Mainly, it reviews the state-of-the-art in research on the development of conceptual object-oriented models for facilities management. It presents a critical review of some of the commercially available computerized maintenance management software. Overall, the review constituted a domain analysis and description, which sever to formulate the research’s points of departures.

7.3 Future Work

In light of the research completed, the following recommendations are made as to future research and development for improved methodologies and computer tools in the FM domain.
7.3.1 Facilities Management Domain

A list of research topics that may be developed beyond this dissertation could include investigating:

1. The effect of complexity on FM projects. Complexity may depend upon the consideration of the number and configuration of assets in a facility.
2. The effect of legislative constraints on the practice of FM. Legislative constraints could require tighter control of asset management due to environmental sensitivity.
3. The effect of project scale on the practice of FM. A large scale, large budget, high quality project might justify development of a more comprehensive maintenance management framework.

7.3.2 Development of Domain Models

It is recommended to continue development in product model standards, as they are enabling technologies for representing information and knowledge, hence communicating data between applications and end users within the domain of FM, and across other AEC domains.

7.3.3 Development of Integrated Applications

Following the implementation efforts reported upon in this dissertation, It is recommended to continue the development of integrated applications that can import and export data from/to other applications, thus offering an edge of software interoperability above existing software tools. More specifically, it is recommended to develop separate or combined integrated tools that treat the following areas:

1. Asset’s performance requirements.
2. Asset’s conditions assessment.
3. Asset’s lifecycle cost and risk analysis.

However, the development of such tools will, to a large extent, depend upon the availability of data standards to represent the knowledge and communicate the data within. Now that these knowledge areas have been modeled, and a set of IFCs have been developed as a product of the research objectives in this dissertation, I strongly advocate the adoption and inclusion of the new set of IFC extensions (presented in Chapter 4) to facilitate developing such integrated tools.

7.4 Conclusions

The research has achieved its objectives and confirmed that a model-based approach to system integration is feasible in the FM domain, and that the resulting application, employing a distributed systems architectures approach, exhibits flexibility and significant integration. I am optimistic that the FM industry at large will embrace the use
the proposed framework and the object-oriented data modeling standards in practice, resulting in powerful, highly integrated FM software systems. The framework can also be applied within any facilities management organization engaged in managing several assets, regardless of whether the tasks involved are implemented by in-house staff or professional maintenance contractors. Motivation for facilities managers to embrace the use of the proposed framework and the object-oriented data modeling standards in practice could include: (1) more efficient management operations that enable the facilities manager to provide an improved service, possibly more profitably; (2) minimizing errors and oversights that reduces the risks of damage and loss; and (3) making the overall project more profitable and returns to the building investor a better return on monetary investment.
References


Federal Information Processing Standards 183. (1993). Integration Definition for Function Modeling (IDEF0), United States National Institute of Standards and Technology (NIST), Computer Systems Laboratory, Gaithersburg, MD., USA.


Appendix A – IDEF₀ Process Modeling Notation Guide

This Appendix describes the notation for IDEF₀ process modeling language (semantics and syntax) followed to graphically present the proposed framework of maintenance management, presented in Chapter 3 of the thesis. The appendix is not meant to provide a full description of the IDEF₀ notation, as it is intended only to provide a description of the techniques used to develop the process models in Chapter 3. IDEF stands for Integration Definition for Function Modeling. Use of IDEF₀ modeling language permits the construction of models comprising system functions (activities, actions, processes, operations), functional relationships, and data (information or objects) that support systems integration (Federal 1993).

A.1 Background

The IDEF₀ notation was one of a series of techniques developed during the 1970s by the U.S. Air Force Program for Integrated Computer Aided Manufacturing (ICAM), to improve manufacturing productivity through systematic application of computer technology. These techniques are as follows:

1. IDEF₀, used to produce a "function model", which is a structured representation of functions, activities or processes within the modeled system or subject area.
2. IDEF₁, used to produce an "information model", which represents the structure and semantics of information within the modeled system or subject area.
3. IDEF₂, used to produce a "dynamics model", which represents the time-varying behavioral characteristics of the modeled system or subject area.

In 1993, the U.S. National Institute of Standards Technology (NIST) documented the IDEF₀ notation as Federal Information Processing Standard (FIPS) 183.

A.2 Structure of IDEF₀

A system is represented through IDEF₀ by means of a model composed of diagrams with supportive documentation that break a complex subject into its component parts. Each diagram consists of boxes and arrows to express the functional activities, data, and the function/data interfaces. Text accompanies each of the diagrams narrating the activities in that diagram (Federal 1993).

A.3 Schematic Presentation

Boxes in the diagram represents functions, corresponding to an activities, actions, processes, operations, or transformations. One or more inputs to the function are transformed into one or more outputs, using the mechanism provided. The transformation process for the data is controlled by one or more controls. The data entities are illustrated schematically in Figure A-1 and Table A-1 (Federal 1993).
Table A-1: Data entity descriptions

<table>
<thead>
<tr>
<th>Entity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function</td>
<td>An activity, action, process, operation, or transformation, which is described by an active verb. A function is shown as a box.</td>
</tr>
<tr>
<td>Input</td>
<td>An entity, which undergoes a process or operation, and is typically transformed. It enters the left of the box, and may be any information or material resource.</td>
</tr>
<tr>
<td>Output</td>
<td>An entity which results from a process or objects which are created by a function. An output is shown existing the right side of the box.</td>
</tr>
<tr>
<td>Control</td>
<td>An entity which influences or determines the process of converting inputs to outputs. A control is shown entering the top side of the box.</td>
</tr>
<tr>
<td>Mechanism</td>
<td>An entity such as a person or a machine, which perform a process or operations. A mechanism is shown entering the bottom side of the box.</td>
</tr>
<tr>
<td>Node</td>
<td>A unique identifier to every function, which is shown in the bottom right hand corner of the box.</td>
</tr>
</tbody>
</table>

A.4 Hierarchy of IDEF0 Diagrams

In IDEF0 the whole system is represented as a single box with arrow interfaces to the environment external to the system. The function in the box is decomposed into between three to six functions, each of which may be further decomposed into sub-functions. This top-down decomposition may be continued, resulting between three to six “child” or detailed diagrams for each function on any given level (Federal 1993). A schematic view of the hierarchy of the diagrams is illustrated in Figure A-2.

![Diagram of function box with interfaces for controls, inputs, outputs, and mechanisms](image)

Figure A-1: Schematic presentation of the function box (Federal 1993)

A.5 Decomposition

The number of functions within each diagram is limited to a minimum of three and maximum of six. These constraints limit the level of detail and complexity in any diagram, yet preventing the diagram from being trivial. Further, the level of detail is controlled by the position of the diagram in the hierarchy of diagrams. The amount of
detail increases through each level of decomposition. This results in gradual exposition of detail. In Chapter 3, each level of decomposition is illustrated as a separate process model, having different figure number. Figure A-2 illustrates the gradual exposition of detail in IDEF₀ diagrams.

A. 6 Modularity of IDEF₀ Diagrams

When a function in a box is decomposed, the scope of the function and its interface arrows create a bounded context for the sub-functions. The scope of the detail or “child” diagram fits completely inside its “parent” function. The interface arrows of the “parent” box match the external arrows of the detail or “child” diagram. Thus, arrows entering and exiting the detail diagram must be the same arrows interacting with the parent diagram (Federal 1993). Figure A-2 shows the modularity of IDEF₀ diagrams.

A.7 Numbering the IDEF₀ Diagrams

Each major function in the system is assigned a node number, which acts as a unique identifier to the function in the process model. The node number is shown in the bottom right hand corner of the box. Decomposition of each box leads to a number of diagrams. Each diagram has a node number, which is formed by taking the node number of the parent diagram and appending to it the number of the box being decomposed (Svensson 1998) and (Federal 1993). The node numbering system allows the reader to trace back the steps of decomposition through the parent function of each diagram.

Figure A-2: The hierarchical structure of the IDEF₀ modeling methodology

(Svensson, 1998)
Appendix B - Proposed Extensions to IFC Model

While the developed integrated model of maintenance management made use of existing IFCs within Release 2.0 and 2X, the research proposes a set of 16 extension to IFCs that are recommended for inclusion within Release 3.0 or later. The listing presents the newly proposed set of extensions to the IFCs, along with some of their indicative properties and corresponding data types:

<table>
<thead>
<tr>
<th>Class</th>
<th>Properties</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>IfcRoofSection</strong></td>
<td>Subtype of IfcBuildingElement</td>
<td></td>
</tr>
<tr>
<td></td>
<td>AdjacentRoofSections</td>
<td>IfcLabel</td>
</tr>
<tr>
<td></td>
<td>ConsequencesOfFailure</td>
<td>IfcCostValue</td>
</tr>
<tr>
<td></td>
<td>DateConstructed</td>
<td>IfcCalendarDate</td>
</tr>
<tr>
<td></td>
<td>DateLastReplaced</td>
<td>IfcCalendarDate</td>
</tr>
<tr>
<td></td>
<td>Perimeter</td>
<td>IfcPositiveLengthMeasure</td>
</tr>
<tr>
<td></td>
<td>AdjacentWallLength</td>
<td>IfcPositiveLengthMeasure</td>
</tr>
<tr>
<td></td>
<td>ProbabilityOfFailure</td>
<td>IfcReal</td>
</tr>
<tr>
<td></td>
<td>RiskOfFailure</td>
<td>IfcLabel</td>
</tr>
<tr>
<td></td>
<td>Access</td>
<td>IfcLabel</td>
</tr>
<tr>
<td></td>
<td>Area</td>
<td>IfcAreaQuantity</td>
</tr>
<tr>
<td></td>
<td>Occupancy</td>
<td>IfcLabel</td>
</tr>
<tr>
<td><strong>IfcFlashing</strong></td>
<td>Subtype of IfcBuildingElement</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CounterFlashing</td>
<td>IfcLabel</td>
</tr>
<tr>
<td></td>
<td>Adhesive</td>
<td>IfcLabel</td>
</tr>
<tr>
<td></td>
<td>Base</td>
<td>IfcLabel</td>
</tr>
<tr>
<td></td>
<td>Perimeter</td>
<td>IfcPositiveLengthMeasure</td>
</tr>
<tr>
<td><strong>IfcMembrane</strong></td>
<td>Subtype of IfcBuildingElement</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Attachment</td>
<td>IfcLabel</td>
</tr>
<tr>
<td></td>
<td>Description</td>
<td>IfcLabel</td>
</tr>
<tr>
<td></td>
<td>Manufacturer</td>
<td>IfcLabel</td>
</tr>
<tr>
<td></td>
<td>Protection</td>
<td>IfcBoolean</td>
</tr>
<tr>
<td></td>
<td>Reinforcement</td>
<td>IfcLabel</td>
</tr>
<tr>
<td></td>
<td>Surfacing</td>
<td>IfcLabel</td>
</tr>
<tr>
<td></td>
<td>Type</td>
<td>IfcLabel</td>
</tr>
<tr>
<td></td>
<td>Warranty</td>
<td>IfcTimeMeasure</td>
</tr>
<tr>
<td></td>
<td>SpecificationNumber</td>
<td>IfcLabel</td>
</tr>
<tr>
<td></td>
<td>Walkways</td>
<td>IfcLabel</td>
</tr>
<tr>
<td></td>
<td>WarrantyExpiration</td>
<td>IfcCalendarDate</td>
</tr>
<tr>
<td><strong>IfcFunctionalRequirement</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Name</td>
<td>IfcLabel</td>
</tr>
<tr>
<td></td>
<td>Description</td>
<td>IfcLabel</td>
</tr>
<tr>
<td><strong>IfcFunctionalRequirement Type</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Description</td>
<td>IfcLabel</td>
</tr>
<tr>
<td></td>
<td>UpperBoundedValue</td>
<td>IfcReal</td>
</tr>
<tr>
<td></td>
<td>LowerBoundedValue</td>
<td>IfcReal</td>
</tr>
<tr>
<td></td>
<td>Unit</td>
<td>IfcLabel</td>
</tr>
<tr>
<td><strong>IfcInspectionOrder</strong></td>
<td>Subtype of IfcProjectOrder</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ProductDescription</td>
<td>IfcLabel</td>
</tr>
<tr>
<td></td>
<td>ShortJobDescription</td>
<td>IfcLabel</td>
</tr>
<tr>
<td></td>
<td>LongJobDescription</td>
<td>IfcLabel</td>
</tr>
<tr>
<td></td>
<td>InspectionTestRequested</td>
<td>IfcLabel</td>
</tr>
<tr>
<td></td>
<td>ContractualType</td>
<td>IfcLabel</td>
</tr>
<tr>
<td></td>
<td>IfNotAccomplished</td>
<td>IfcLabel</td>
</tr>
<tr>
<td></td>
<td>RequestedStartTime</td>
<td>IfcDateAndTime</td>
</tr>
<tr>
<td></td>
<td>RequestedFinishTime</td>
<td>IfcDateAndTime</td>
</tr>
<tr>
<td></td>
<td>ActualStartTime</td>
<td>IfcDateAndTime</td>
</tr>
<tr>
<td></td>
<td>ActualFinishTime</td>
<td>IfcDateAndTime</td>
</tr>
<tr>
<td></td>
<td>CostEstimate</td>
<td>IfcCostSchedule</td>
</tr>
<tr>
<td><strong>IfcInspectionTask</strong></td>
<td><strong>Subtype of IfcTask</strong></td>
<td></td>
</tr>
<tr>
<td>------------------------</td>
<td>------------------------</td>
<td></td>
</tr>
<tr>
<td>Status</td>
<td>IfcInspectionOrderStatusEnum</td>
<td></td>
</tr>
<tr>
<td>PerformedBy</td>
<td>IfcPerson</td>
<td></td>
</tr>
<tr>
<td>ActualCost</td>
<td>IfcCostValue</td>
<td></td>
</tr>
<tr>
<td><strong>InspectionTaskID</strong></td>
<td>IfcLabel</td>
<td></td>
</tr>
<tr>
<td><strong>InspectionTaskName</strong></td>
<td>IfcLabel</td>
<td></td>
</tr>
<tr>
<td>Status</td>
<td>IfcInspectionTaskStatusEnum</td>
<td></td>
</tr>
<tr>
<td>Milestones</td>
<td>IfcInspectionTaskMilestoneEnum</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>IfcInspectionTaskType</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Creators</td>
</tr>
<tr>
<td>Purpose</td>
</tr>
<tr>
<td><strong>StartTime</strong></td>
</tr>
<tr>
<td><strong>FinishTime</strong></td>
</tr>
<tr>
<td><strong>TotalFloat</strong></td>
</tr>
<tr>
<td><strong>Duration</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>IfcInspectionTest</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>InspectionTestDescription</strong></td>
</tr>
<tr>
<td><strong>EquipmentNeeded</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>IfcInspectionResult</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AnomalyDescription</strong></td>
</tr>
<tr>
<td><strong>ActionNeeded</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>IfcMRRTask</strong></th>
<th><strong>Subtype of IfcTask</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>MRRTaskID</td>
<td>IfcLabel</td>
</tr>
<tr>
<td><strong>MRRTaskName</strong></td>
<td>IfcLabel</td>
</tr>
<tr>
<td><strong>MRRStatus</strong></td>
<td>IfcMRRTaskStatusEnum</td>
</tr>
<tr>
<td><strong>MRRMilestones</strong></td>
<td>IfcMRRTaskMilestoneEnum</td>
</tr>
<tr>
<td><strong>MRRWorkMethod</strong></td>
<td>IfcLabel</td>
</tr>
<tr>
<td><strong>InplaceQuantity</strong></td>
<td>IfcMeasureWithUnit</td>
</tr>
<tr>
<td><strong>EstimatedQuantity</strong></td>
<td>IfcMeasureWithUnit</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>IfcMRRTaskType</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MRRMethod</strong></td>
</tr>
<tr>
<td><strong>StartTime</strong></td>
</tr>
<tr>
<td><strong>FinishTime</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>IfcCondition</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ConditionIndex</strong></td>
</tr>
<tr>
<td><strong>ConditionStatement</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>IfcConditionType</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td><strong>Description</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>IfcRiskSchedule</strong></th>
<th><strong>Subtype of IfcControl</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Title</strong></td>
<td>IfcLabel</td>
</tr>
<tr>
<td><strong>SubmittedBy</strong></td>
<td>IfcActorSelect</td>
</tr>
<tr>
<td><strong>ApprovedBy</strong></td>
<td>IfcActorSelect</td>
</tr>
<tr>
<td><strong>PreparedBy</strong></td>
<td>IfcPerson</td>
</tr>
<tr>
<td><strong>SubmittedOn</strong></td>
<td>IfcCalendarDate</td>
</tr>
<tr>
<td><strong>Status</strong></td>
<td>IfcLabel</td>
</tr>
<tr>
<td><strong>CostOfFailure</strong></td>
<td>IfcCostValue</td>
</tr>
<tr>
<td><strong>ConsequencesOfFailure</strong></td>
<td>IfcCostValue</td>
</tr>
<tr>
<td><strong>IntendedUse</strong></td>
<td>IfcText</td>
</tr>
<tr>
<td><strong>Comments</strong></td>
<td>IfcText</td>
</tr>
<tr>
<td><strong>TargetUsers</strong></td>
<td>IfcActorSelect</td>
</tr>
<tr>
<td><strong>ValidFromDate</strong></td>
<td>IfcCalendarDate</td>
</tr>
<tr>
<td><strong>ValidToDate</strong></td>
<td>IfcCalendarDate</td>
</tr>
<tr>
<td><strong>UpdateDate</strong></td>
<td>IfcCalendarDate</td>
</tr>
<tr>
<td><strong>ScheduleNumber</strong></td>
<td>IfcIdentifier</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>IfcResourceType</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td><strong>Description</strong></td>
</tr>
<tr>
<td><strong>ProductivityRate</strong></td>
</tr>
</tbody>
</table>