INDUSTRY FOUNDATION CLASSES FOR PROJECT MANAGEMENT - A TRIAL IMPLEMENTATION

SUBMITTED: March 1999
REVISED: November 1999
PUBLISHED: November 1999 at http://itcon.org/1999/2/
EDITOR: B-C. Bjoerk

Thomas Froese, Associate Professor
University of British Columbia, Canada
email:tfroese@civil.ubc.ca,
http://www.civil.ubc.ca/~tfroese/

Martin Fischer, Associate Professor
Stanford University, USA
email:fischer@ce.stanford.edu,
http://www.stanford.edu/~fischer

Francois Grobler, Civil Engineer/Principal Investigator
US Army Corps of Engineers Construction Engineering Research Laboratory
email:f-grobler@cecer.army.mil

John Ritzenthaler
Timberline Software Corporation, USA
email:johnritz@gte.net

Kevin Yu, Software Developer R&D,
Timberline Software Corporation, University of British Columbia, Canada
email:keviny@timberline.com

Stuart Sutherland
Bovis Construction Corporation, USA
email:stuart.sutherland@bovis.com

Sheryl Staub, Graduate Research Assistant
email:sstaub@leland.Stanford.edu

Burcu Akinci, Ph.D. Candidate
email:akinci@leland.Stanford.edu

Ragip Akbas, Graduate Research Assistant
email:rakbas@leland.Stanford.edu

Bonsang Koo, Graduate Research Assistant
email:bonsang@leland.Stanford.edu

Alex Barron, Graduate Research Assistant
email:abarron@leland.Stanford.edu

John Kunz, Senior Research Scientist
email:kunz@ce.stanford.edu
all: Stanford University, USA

SUMMARY*: An effort is underway to develop Industry Foundation Classes--industry-standard data structures for exchanging information about construction projects. In addition to physical information about buildings, these classes represent project management information such as estimating and scheduling data. Many core concepts relating to the project management portions of the Industry Foundation Classes have recently been added to these models, but these have received virtually no testing and implementation to date. A workshop was held to work through an implementation and evaluation exercise for these models. The results identified many areas for potential improvement, but also confirmed that the overall approach taken by the models worked well for representing and integrating product, work process, estimating, and scheduling information.*

KEYWORDS*: International Alliance for Interoperability, IAI, Industry Foundation Classes, IFC, Project Management, Estimating, Scheduling, Project Modelling, Data Standards, Integration.*

1. INTRODUCTION

For the past decade, the "Holy Grail" for the field of information technology in the architecture, engineering, construction and facilities management (AEC/FM) industry has been information integration throughout the suite of computer tools used to carry out engineering projects. Vendor-specific approaches have been developed (Staub et al. 1998), but these are limited in their coverage of their range of potential applications and users. An open, industry-wide solution is required, but this can only exist with standardised data models to support the exchange of information among disparate systems.

The Industry Alliance for Interoperability (IAI) is a global, industry-based consortium for the AEC/FM industry (IAI 1996, IAI 1998). Their mission is to enable interoperability among industry processes of all different professional domains in AEC/FM projects by allowing the computer applications used by all project participants to share and exchange project information. The IAI's scope is the entire lifecycle of building projects including strategic planning, design and engineering, construction, and building operation. The IAI's goals are to define, publish and promote a specification--called the Industry Foundation Classes (IFCs)--for sharing data throughout the project lifecycle, globally, across disciplines and technical applications (IAI 1998). The IFCs are used to assemble a project model in a neutral computer language that describes building project objects and represents information requirements common throughout all industry processes.

Much of the IFCs' focus has been on representing the facilities that are being designed and constructed, but the IFCs' scope also includes project management information such as costs, schedules, work tasks, resources, etc. The Project Management (PM) Domain Group of the IAI North American Chapter (IAI NA PM) has been developing portions of the IFCs to support estimating and scheduling processes. Much of these processes have now been incorporated into the IFCs. To date, however, there has been virtually no attempt to implement and test these portions of the IFCs. In February 1999, the IAI NA PM group held a charrette-style workshop to conduct trials of the project management portions of the IFCs. During the workshop, participants developed usage scenarios, drafted "paper models" of the scenario data based on the IFCs' data structures, and then implemented the scenarios in a computer system to exhibit some basic data sharing functionality. We wanted to test how well the IFCs represent the cost estimating and scheduling information necessary for everyday project management tasks. This paper describes this exercise, presents the scope of the IFCs related to PM and discusses the lessons learned from exposing these IFCs to an initial trial application--"first light" for the project management IFCs. It should be of interest to researchers and practitioners modelling construction process information.

2. THE TESTING PROCESS

From February 18 to 20, 1999, the IAI NA PM committee held a three-day workshop. The main objective of the workshop was to conduct a charrette-style exercise (Clayton et al. 1998) to evaluate the current IFC models as they relate to the project management tasks of estimating and scheduling. The purpose of the workshop was to determine how well IFCs serve real world representational needs of a project. The charrette-style approach was chosen so that we could bring the range of expertise and skills of the group members to bear on the problem. The tasks carried out were designed to work through as much implementation as possible within the three days available (some conclusions relating to the charrette process itself are presented at the end of this paper).

Although the IFCs have included some project management-related objects since their initial Release, a significant number of new and revised objects have been included in the IFC Release 2.0 as a result of the IAI NA PM group's and others' activities. To date, there has been very little testing and validation of the PM-related portions of the IFCs. A similar charrette workshop held in February 1999 focused on facilities management support within the IFCs (CIFE, 1999). A very few systems have been implemented with some application of IFC PM objects (Laitinen, 1999), but these were based on Releases 1.0 or 1.5, which were substantially different from the Release 2.0 PM models. Although many systems have been developed using IAI or ISO STEP models, these deal mainly with product model information rather than PM information, or they have used custom rather than standard models for PM data, since previously there has been only nominal support for project management information in IAI or STEP Building Construction.

The workshop was held at the Center for Integrated Facility Engineering (CIFE) at Stanford University and was attended by six committee members and a number of graduate research assistants from the Stanford Construction Engineering and Management Program. This group represented university and government researchers, industry practitioners, and software vendors, and most of the participants had expertise in at least two of the following disciplines: construction, data modelling, and commercial software development. The workshop began with a review of the current status and content of the IFC models. An overview of these models is provided in Section 4. Next, the CIFE group presented several project scenarios that they had developed in advance of the workshop. These scenarios identify typical project management tasks and provide actual project data with which to work. We then split into two groups corresponding to two of the basic scenarios, one focusing on estimating and the other on scheduling.

The first step in our attempt to apply the IFCs was for each group to prepare "paper models" of their scenario. That is, to try to structure the data from the scenario according to the IFC classes and to write these out on a whiteboard (the estimating group first entered the data into the commercial estimating application as an intermediate step in formalising the scenario information). After this task was completed, the resulting object diagrams were recorded in computers using UML notation. UML was selected since EXPRESS-G, the notation used for class diagrams within the IAI, contains no notation for representing object (i.e., instance) diagrams. Each group presented the results to the other and discussed issues such as the degree of support that the IFCs offered for representing the information, specific problems that could be identified, etc.

The next step was to implement the scenarios in a computer system using the IFCs. We did this by entering the IFC classes into PowerModel, an object-oriented modelling tool (Intellicorp 1999). We then defined instances of these classes to represent the data used for the scenarios. The result was that all of the data required for the scenarios were implemented in a computer system. This shows that the IFCs enable the data to be represented in a computer-interpretable form (although it does not yet provide any functionality). Next, the scheduling group went on to implement some rudimentary planning functionality in the PowerModel system. Again, the groups met to discuss the results, analyse the lessons learned, and record the outcome in this paper.

3. THE SCENARIOS

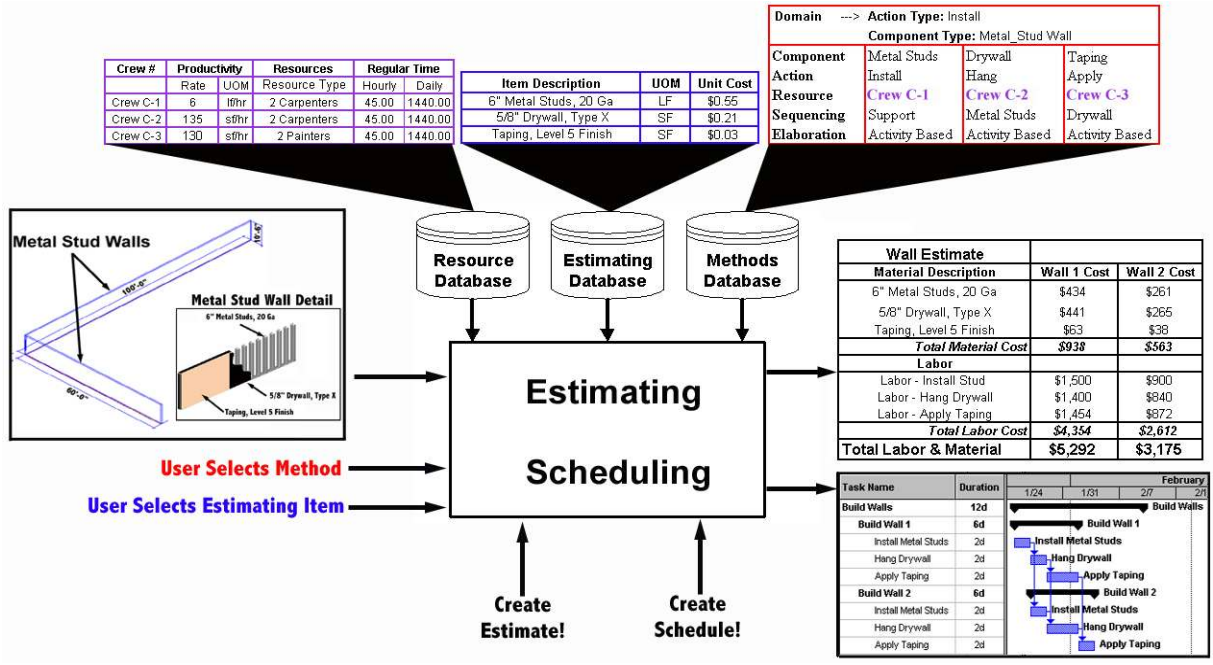

FIG. 1: Test Case Scenario.

To demonstrate the IFC models, we used a simple test case scenario drawn from actual data for a recent bio-technology facility project in California called Sequus Pharmaceuticals (Staub et al. 1998). The test case describes two walls, each one decomposed into three types of components: metal studs, drywall, and taping. The dimensions of these components are listed in Table 1.

TABLE 1: Dimensions of Components in the Sample Project

Object	Length	Height	Area
Wall 1	100 ft	10.5 ft	1050 Sq.Ft.
Wall 2	60 ft	24 ft	1440 Sq.Ft.

Fig. 1 describes this test case in a format based roughly on the $IDEF_0$ process model notation. The objectives of the test case were to create a schedule and a cost estimate for the walls at two levels of detail. On the left of the diagram are inputs to the process--in this case, the geometric description of the product model from CAD. On

top are constraints on the process--shown here as three databases with typical values used to create the estimate and schedule. On the bottom are the methods or reasoning processes that applications use to transform the data provided by the inputs and databases to create the outputs, which are shown on the right. One output is the schedule, shown here as a set of three activities for each wall: Install Metal Studs, Hang Drywall, and Apply Taping. The other output is the cost estimate consisting of material and labour breakdowns.

The creation of the sample implementations using specific values was a useful way of bringing out the strengths and weaknesses of the current version of the IFC models. In the process of specifying the attributes necessary for each object and the relationships that enabled integration of these data between objects, we were able to determine many of the modelling requirements for proper representation of the data needed to support the stated objectives for information sharing (as discussed in Section 6).

4. OVERVIEW OF IFC RELEASE 2.0 PROJECT MANAGEMENT CONCEPTS

Before we describe our application of the IFC models to the test scenario, we provide a brief overview of the IFCs and some of the core concepts used to model project management information. Closely related IFCs are group into schemas to provide modularity. This section provides a general introduction to IFC standards and then discusses the core concepts for the IFC Release 2.0 schemas that pertain to project management. For further information, see Froese and Yu (1999), which contains class diagrams for these models, Cole et al. (1998), Grobler et al. (1998).

4.1 Standards and the IFCs

The IFC effort is a standardisation effort. Some standardisation efforts have been highly successful. Most people around the world can send mail anywhere in the world. Most telephones in the world now interoperate globally. Internet users everywhere in the world can retrieve web pages from anywhere in the world. Buildings all over the US adhere to fundamentally the same high standards of occupancy safety.

Standards tend to be of two types: interface standards and content standards. An example of the former is a computer device connection standard such as RS232 or the TCP/IP protocol. Interface protocols are common for physical devices (e.g., train track width), electrical systems (e.g., RS232) and software (e.g., TCP/IP). Some interface standards emerge from market dominance of a product company (e.g., DXF format for CAD files), and some are developed by international standards bodies such as ISO. The Uniform Building Code is a successful example of a content standard. Medical practice is moving to content standards as official bodies specify "protocols" that prescribe in great detail the procedures of how to treat different medical situations. In comparison with interface standards, content standards are much larger and more complicated. They refer explicitly to a standard vocabulary of conceptual entities and defined processes. Governments often prescribe them (e.g., building codes, medical treatment protocols for cost reimbursement).

While an interface standard would be much easier to create and manage, the IFCs necessarily have the flavour of a content standard. Design models often get created initially in CAD software as graphic models of either a schematic or physical design. Designers add semantic attachments to describe, relate and elaborate initial graphics entities. AEC software applications generally do not now interoperate. An engineer manually reviews output from one application and formulates input to a related one. That manual translation is an engineering problem--and the motivation behind the IFCs and STEP (ISO Standard 10303, see ISO 1994). The complication for the IAI members is that the semantic description of design entities is enormously complicated, and there is generally no recognized explicit and formal representation of either the semantics of design models or of engineering analysis procedures. STEP and IFC are ongoing and dauntingly difficult efforts to represent some of the design and process model and semantics explicitly and formally in computer-interpretable form.

4.2 IfcProcessExtension Schema

The central concepts of products, processes, resources, actors, etc., are modelled in the IfcKernal schema. Other schemas then add additional detail to these core concepts. The IfcProcessExtension schema extends the basic IfcProcess entity defined in the IfcKernal Schema.

4.2.1 Construction Processes and Planning Documents

An important aspect of the project management portions of the IFCs is the explicit representation of work processes. While cost estimate items and schedule activities can be associated with work processes, the processes themselves are a distinct element that can be independently described and represented in project models. There are two major entities in the IFCs to support construction processes: IfcProcess and IfcWorkTask. The former represents any general actions taking place in completing a work of design, engineering, construction, or facilities management. IfcWorkTask, a subtype of IfcProcess, is intended to describe the work processes that make up construction operations. Information such as construction work methods, schedule dates and duration is defined at the IfcWorkTask level.

4.2.2 Work Plans

Any collection of IfcWorkTask instances makes up a plan, represented by the entity IfcWorkPlan. Tasks can be organised in different ways to make up different IfcWorkPlan instances. For example, tasks may be organised into a nested hierarchy according to the type of work required to define a plan that supports cost estimating. The same tasks can then be organised into another nested hierarchy according to work locations (possibly with additional work tasks added to expand certain details) to define a plan that supports work scheduling.

4.2.3 Work Schedules

Time scheduling information can be assigned to work plans and work tasks by associating them with related work scheduling objects. Specifically, an instance of the entities IfcWorkScheduleElement and IfcScheduleTimeControl, which hold time scheduling information such as start dates, end dates, float times, etc, can be associated with an IfcWorkTask instance to represent all the date and duration information for the work task (i.e, the combination of an IfcWorkTask and an associated IfcWorkScheduleElement and IfcScheduleTimeControl provide the equivalent of a traditional scheduling activity). All IfcWorkScheduleElement instances for a work schedule are grouped into an IfcWorkSchedule instance. One or more IfcWorkSchedule instances can be associated with any IfcWorkPlan. Thus, the related instances of IfcWorkPlan and IfcWorkSchedule comprise a subset of basic planning documents for a project.

4.2.4 Process Nesting

Work tasks can represent a process at any level of detail, from broad project phases to very detailed tasks. All levels have the same data structure rather than defining different entities for different levels of activities. For example, an overall project, its development phases, work packages, activities, work tasks, and operations can all be represented as instances of IfcWorkTask. A key requirement for the estimating and scheduling integration process is the nesting capability of work tasks. That is, a work task can be broken down into sub-tasks, but it still remains the same work task itself. In the IFCs, the entity IfcRelNestsProcesses is used to establish the nesting relationship between work tasks.

4.2.5 Process Sequence

The sequence of processes is defined as an objectified relationship between IfcProcess instances, as represented by the entity IfcRelSequence. This entity establishes the link between a successor and a predecessor process, providing a time lag and sequence type (e.g., start-to-start or start-to-finish sequence, etc.).

4.2.6 Linkages with Products

One of the central relationships in modelling construction processes is the association between processes and the products upon which they operate. The IFC approach to this is to use an objectified relationship entity named IfcRelProcessOperatesOn between IfcProduct and IfcObject indicating the operation type such as install, transfer, operation on, construct, remove, erect, and so on.

4.2.7 Resource and Process Relationship

Processes use resources. This resource-use relationship is modelled by IfcRelUsesResource as an objectified entity carrying information such as resource use duration, quantity, waste factor, and costs.

4.3 IfcProjectMgmtExtension Schema

The IfcProjectMgmtExtension schema extends the IfcKernal schema in PM-related areas such as costs and project orders.

4.3.1 Object Costs and Cost Context

In IFC R2.0, object costs are handled primarily in the project management extension schema which supports cost estimating for both construction and facilities management. For cost estimating, different types of costs are assigned to objects such as work tasks or products. This section explains how the IFCs currently handle costs.

Costs assigned to objects can only provide meaningful information if the context of the cost values is known. For example, information about whether the unit price of a work task includes material purchases, transportation, material use wastes, equipment uses (operational or rental costs), labour costs, taxes, general contractor mark-up, etc. must be provided along with the numerical cost value. The IFC entity IfcCostElement addresses this by providing cost information, relating to the object being costed, and relating to a cost schedule document (IfcCostSchedule) that describes the context of a list of cost elements. IfcCostSchedule can be used to represent any form of cost list, such as an estimate, a budget, or a unit price table. An IfcCostElement instance can be associated with the objects being costed (e.g., a product, a resource, a process, etc.), through the objectified relationship IfcRelCostsObjects. An IfcCostElement can also be used to group related sub-costs using an IfcRelNestsCostElements relationship.

4.3.2 Project Orders

In IFC R2.0, concepts such as work orders, purchase orders, and change orders created during a construction project or a facilities management process are captured in the project management extension schema.

These concepts are modelled as objects where the actual documents or computer files that hold the information about the objects can be referenced. Also, since they are modelled as objects, it is possible that these concepts can be associated with other related things or concepts. For example, a change order can be associated with a cost estimate and a work plan so that an application can retrieve not only the information about the change order itself but also the estimated cost and proposed work plan for doing the change. Similarly, a work order can be associated with the building elements such as a piece of equipment that receives maintenance referred by the work order. These associations allow the integration of different applications involved in creation and utilisation of the work orders for maintenance.

4.4 IfcConstructionMgmtDomain Schema

The IfcConstructionMgmtDomain schema models additional concepts required to support the construction management domain.

4.4.1 Construction Resources

Construction resources are things that are used to carry out construction processes. (Section 4.2.7 showed that resources are related to processes.) The entity IfcResource represents types of resources or individual occurrences of resources needed to aid in a construction process. The IFCs currently support five different resource types: subcontractor, construction equipment, construction material, labour, crew, and product resources. A crew is a collection of resources (typically a collection of labour resources with some associated equipment and materials). A product resource is used in the situation where a product that results from a work task is used as a resource in another process.

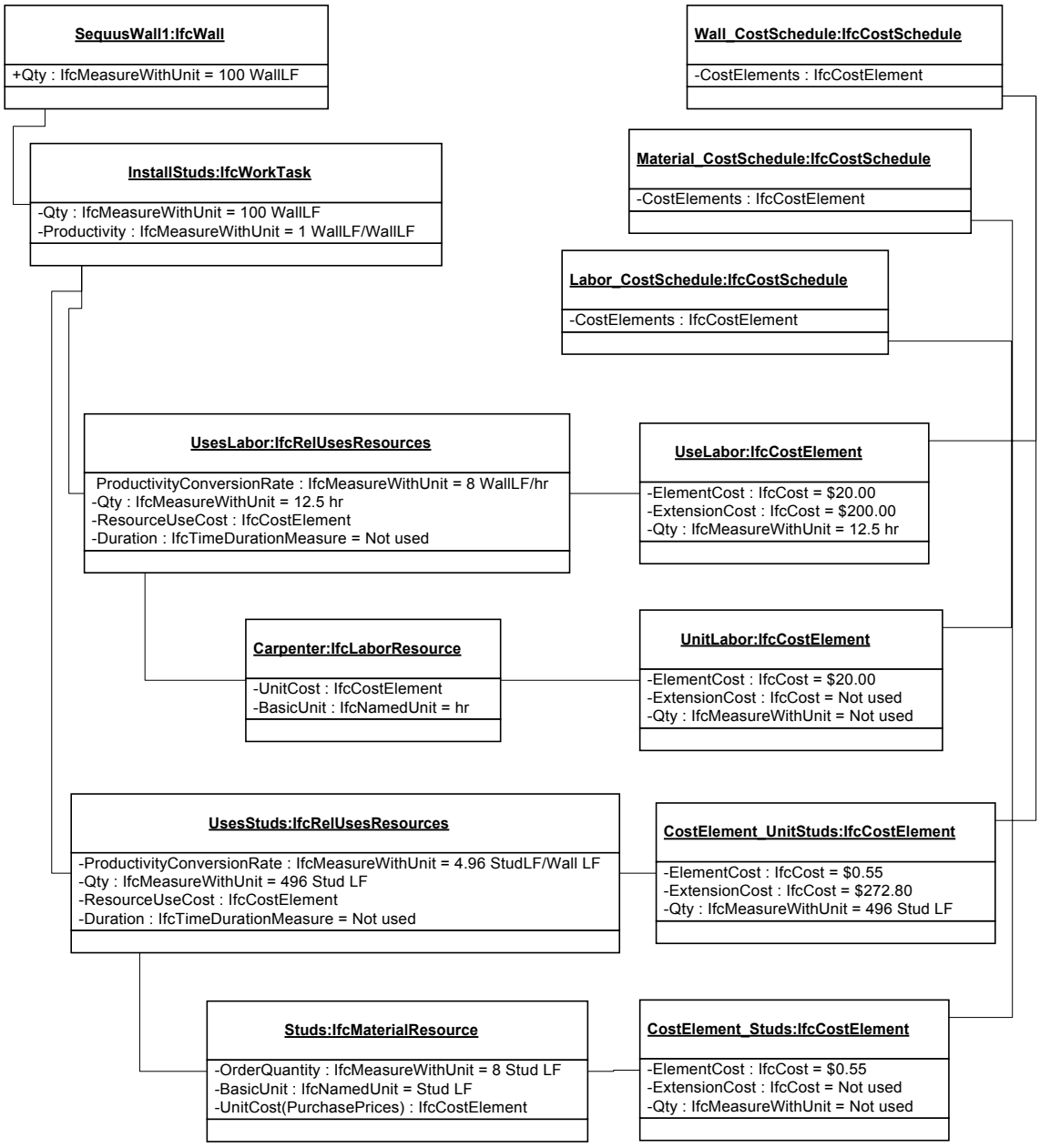

FIG. 2: UML Object Diagram of the Estimating Scenario

Although it is common to model things such as construction equipment, materials, and labor as resources, it is difficult to model them consistently because they play different roles on a project. Generally, things should be modeled as "what they are" rather than as "a role they play" (Sowa 1984). However, the characterization of things as resources, products, etc, can be very dependent upon the perspective of the user of the information. For example, a crane might be represented as a temporary constructed product, materials might be represented as design properties or as the basic components in a materials management application, and labor might be represented as part of the organizational information for a project. The concept of "resources" represents a role that certain things play on construction projects, and it is difficult to design representational structures that satisfy all these different perspectives. Thus, the NA PM group proposed the subtle but important change for the IFC Release 2.0 that IfcResource be interpreted as representing “the use of a thing in the role of a construction resource,” rather than representing the thing itself. If only basic resource information, such as the names, quantities, and prices, is of interest to users of a project model, then IfcResource objects alone are sufficient to

represent the information. However, if further information is required about the things that are being used as resources, then the IfcResource instances can be associated to other instances that represent those things (i.e., IfcProductRes and IfcMaterialRes can be associated to IfcProduct objects, IfcLaborRes can be associated with IfcActor instances, etc).

5. THE IMPLEMENTATION

5.1 Implementation of the Estimating Task

We implemented the estimating task by first creating an estimate using Timberline Precision Estimating software and then mapping that information into the IFC model. Fig. 2 illustrates this model in UML Object Diagram notation: rectangles represent instances of the IFC classes; the top line shows the instance name followed by the class name; the middle section shows attributes (attribute name, attribute type, and attribute value); and links between rectangles show relationships between instances.

5.1.1 Table-Level Mapping

The Precision estimate has a single table of estimate items (see Table 2). Although the IFCs have a comparable class to represent cost elements (discussed below), we first represented the information using the more fundamental IFC classes for work tasks and resources, from which we later derived cost element information (see Table 3). Using the fundamental IFC classes made the underlying cost information more explicit than a direct mapping between Precision estimate items and IFC cost elements. It was straightforward to map the information between the Precision tables and the IFC classes. However, the mapping between Precision estimate items and IFC classes would have been very difficult had we used some of the more complex organisations supported by the IFC model, such as nested work tasks and unlimited resources per work task, because the Precision estimate items do not provide these richer representations.

TABLE 2: Estimate Items Represented in the Estimating Software

Precision Estimate Items	
Item Description	**Cost Category**
Install 6” metal studs	Labour
	Material
Install Drywall	Labour
	Material
Tape Drywall	Labour
	Material

TABLE 3: Estimate Items Represented in the IFC Model

IFC Model	
Work Task	**Resource**
Install 6” metal studs	6” metal studs
	Labour
Install Drywall	6” metal studs
	Labour

5.1.2 Quantity and Unit of Measure Mapping

With the Precision cost item tables mapped to the IFC work task and resource objects, our next step was to map the quantities and units of measure. Precision supports a "Takeoff" quantity and Unit of Measure for each Item. Conversion/Productivity factors are available for each Cost Category. IFCs support a class called IfcMeasureWithUnit. This combines a numeric value with one or more units of measure (i.e. 100 ft/hr). Table 4 shows the mappings used.

TABLE 4: Mapping of Quantities and Units of Measure between Estimating Software and IFC Objects

Precision Field	IFC Class	IFC Attribute
Item Quantity	IfcProduct (IfcWall)	Quantity
Item Quantity	IfcWorkTask	Quantity
Labor Productivity Rate	IfcRelUsesResource	ProductivityConversionRate
Labor Productivity Multiply/Divide	IfcRelUsesResource	ConverterMultiplierNotDivider
Labor Hours	IfcCostElement	Quantity
Material Conversion Factor	IfcRelUsesResource	ProductivityConversionRate
Material Conversion Multiply/Divide	IfcRelUsesResource	ConverterMultiplierNotDivider
Material Quantity	IfcCostElement	Quantity

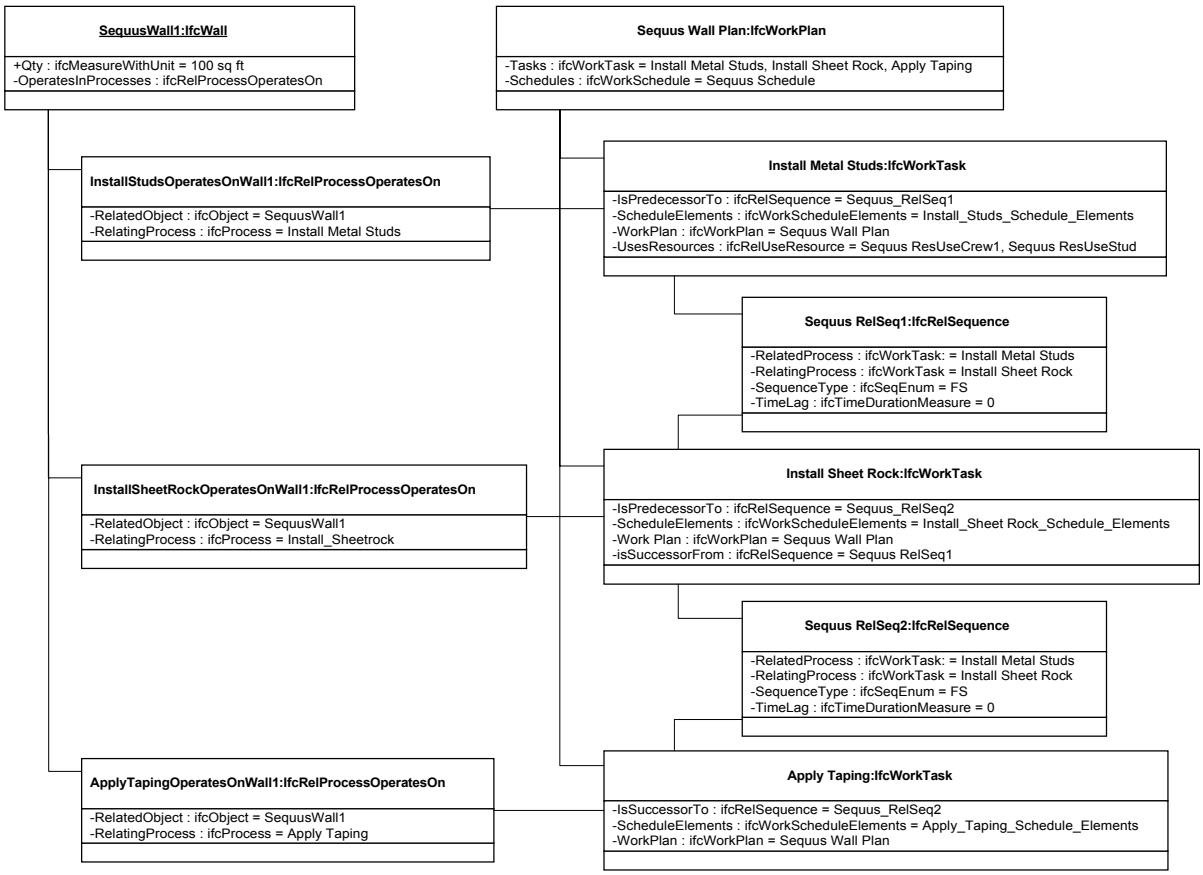

FIG. 3: UML Object Diagram of Scheduling Scenario, Work Plan Objects

When using conversion factors, Precision allows the factor to be used as either a multiplier or a divisor (e.g., you may specify units per hour or hours per unit). We added the attribute "ConverterMultiplierNotDivider" to IfcRelUsesResource to provide equivalent functionality. Initially, we were tempted to use the

IfcRelUsesResources' Duration attribute to specify labour hours, but we later decided that this attribute was for scheduling purposes.

5.1.3 Costs

Precision stores both unit costs and total costs by cost category. An internal flag determines if the total cost is calculated or is a lump sum value. In the IFCs, costs are stored in IfcCostElement objects. These objects include the attributes ElementCost, ExtensionCost, and Quantity. When an extended cost is needed, ExtensionCost is always used. After reviewing our use of IfcCostElement, we saw the need to specify whether the cost had originally been specified as a unit cost or a lump sum. Therefore, we added the attribute IfcCostUseEnum to IfcCostElement. The value of this attribute may be any one of ExtensionOnly, ElementOnly, ElementSetExtensionCalc, ExtensionSetElementCalc, UserDefined, or NotDefined.

5.1.4 Cost Schedules

Precision manages the summarisation of costs automatically. In the IFCs, costs are collected by relating them to IfcCostSchedule objects. An estimate is produced by summarising all the IfcCostElements in the IfcCostSchedule for the estimate. We also created other cost schedules. Each labour resource (IfcLaborResource) has a related IfcCostElement to store the unit labour costs. We defined a labour cost schedule by creating an IfcCostSchedule object and relating it to these IfcCostElements. We similarly defined a material cost schedule. Most participants initially found this use of multiple cost schedules to be confusing.

5.2 Implementation of the Scheduling Task

We conducted the trial implementation of the scheduling task by first working through the basic tasks that would be followed to complete the scheduling and then attempting to model the test case information required to perform these tasks on a white board using the IFC classes. The resulting object diagram was quite large. Fig. 3 illustrates the objects used to represent the work plan associated with the building product objects. Fig. 4 illustrates the work schedule and resource objects associated with one of the work task objects (similar work schedule and resource objects were defined for the other work task objects as well).

We planned and scheduled the object IfcWall:SequusWall (i.e., an object called SequusWall that is an instance of the class IfcWall) by first defining an ifcWorkplan and then creating an ifcWorkSchedule. IfcWorkPlan can point to multiple IfcWorkSchedules to allow alternative schedules or different types of schedules, i.e., a schedule for the owners' use, another for the sub-contractors, etc. We used the IFCs to represent the project information within the scheduling application because we did not use an existing scheduling software for the charrette.

The ifcWorkPlan features three instances of IfcWorkTask as defined in the test scenario for each of the two wall segments: Install Metal Studs, Hang Drywall and Apply Taping. The IfcWorkTasks are contained directly in the IfcWorkPlan as values of the Tasks attribute. A more flexible, objectified relationship was created between the IfcWall:SequusWall object and the IfcWorkTask objects using instances of the objectified relationship class IfcRelProcessOperatesOnObject. Sequence relationships between the IfcWorkTasks are represented by IfcRelSequence objects, which have attributes for time lag, sequence type, etc.

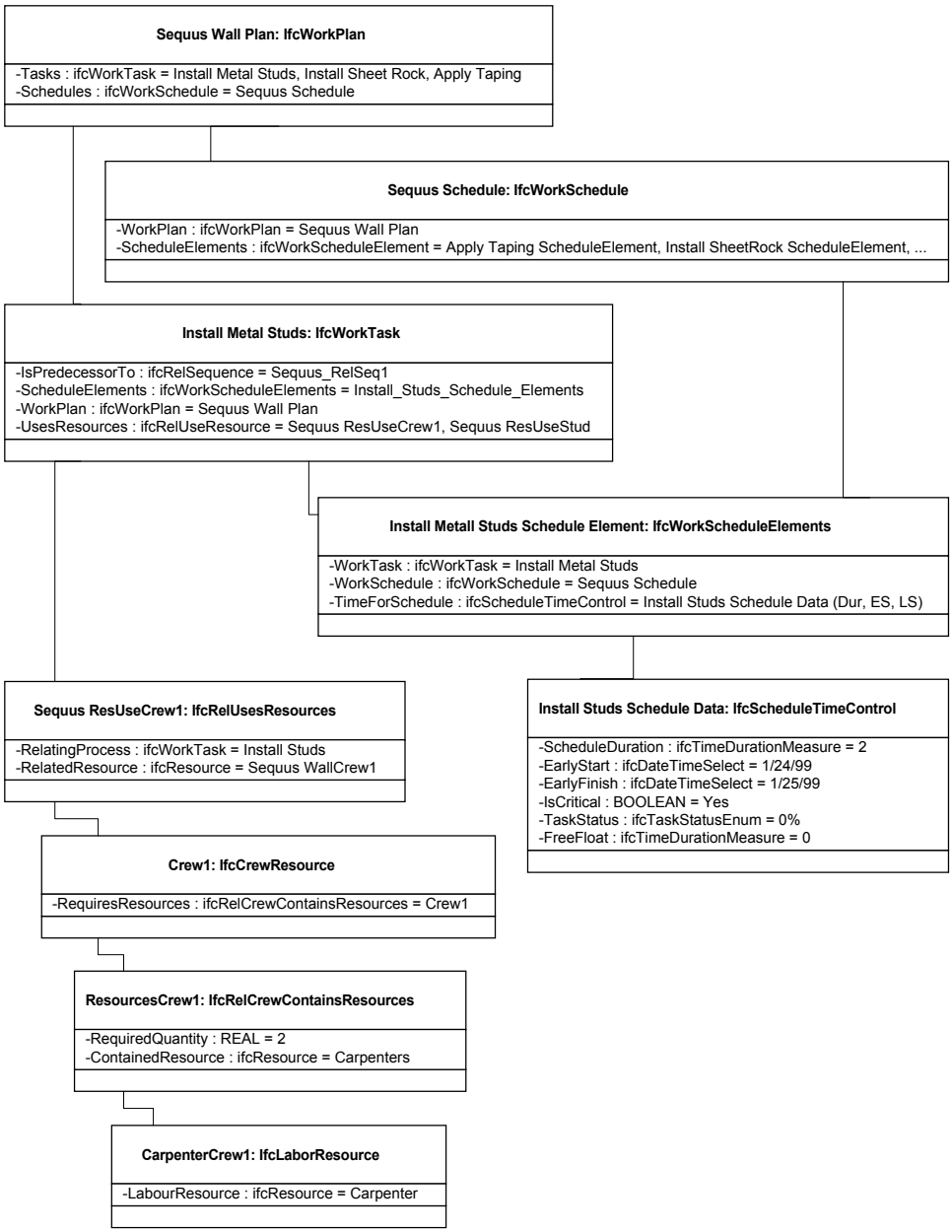

FIG. 4: UML Object Diagram of Scheduling Scenario, Sample Work Schedule and Resource Objects

We represented resource assignments using IfcRelUsesResource objects, which link one or more IfcCrews to an IfcWorkTask. Crew composition is defined by IfcRelCrewContainsResource objects, which specify the number and type of resources that make up a crew. A crew can also contain another crew, or resource, to allow nested resources.

IfcWorkSchedule acts as the entry point for defining and accessing scheduling information and it has attributes to describe the purpose, creator, creation date and other administrative schedule elements. Traditional schedule activity information is distributed between related IfcWorkTask, IfcScheduleElement, and IfcScheduleTimeControl objects. IfcWorkTask contains task information such a description and codes to represent a predefined work breakdown structure element; IfcScheduleElement acts as an objectified relationship between IfcWorkTask and IfcScheduleTimeControl; and IfcScheduleTimeControl has attributes to define various start and finish dates and float, both actual and scheduled, etc.

5.2.1 Implementation of a Planning Algorithm

As described previously, our methodology was to define objects first on a white board, then in a computer-based UML diagram, and then to implement the object model in the PowerModel software. Fig. 5 provides an example of a user interface used for displaying the object model in the computer system.

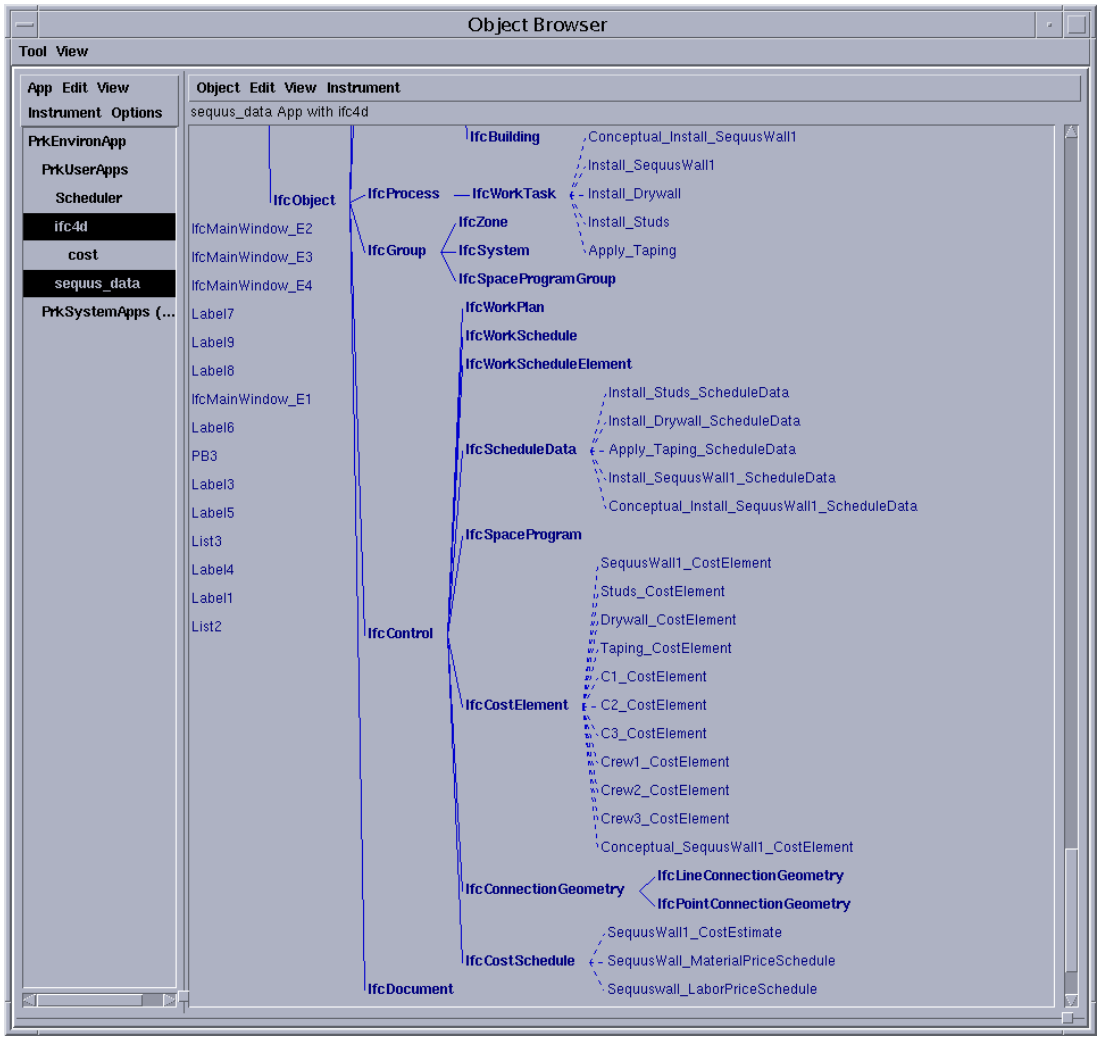

FIG. 5: Implementation Interface

After working through the basic scheduling process, we considered the process of computer-assisted creation of a cost estimate and a schedule related to the Sequus wall test case. Automatic generation of cost estimates and construction schedules of extracted building components requires the representation of building elements' product types, their related processes, and unit costs. Since the IFCs currently do not support product and process types, we created two additional objects to represent product types (CifeProductType) and their related process types (CifeProcessType) and defined a new relationship attribute (of_product_type) to relate instances of IfcProduct to instances of CifeProductType. We implemented the objects used for the scheduling process along with these additional objects and relationship attributes in the PowerModel system and added some simple planning reasoning (the reasoning algorithm is illustrated in Fig. 6). We were then able to automatically generate a construction schedule and a conceptual cost estimate for the Sequus Wall test case from the list of building elements.

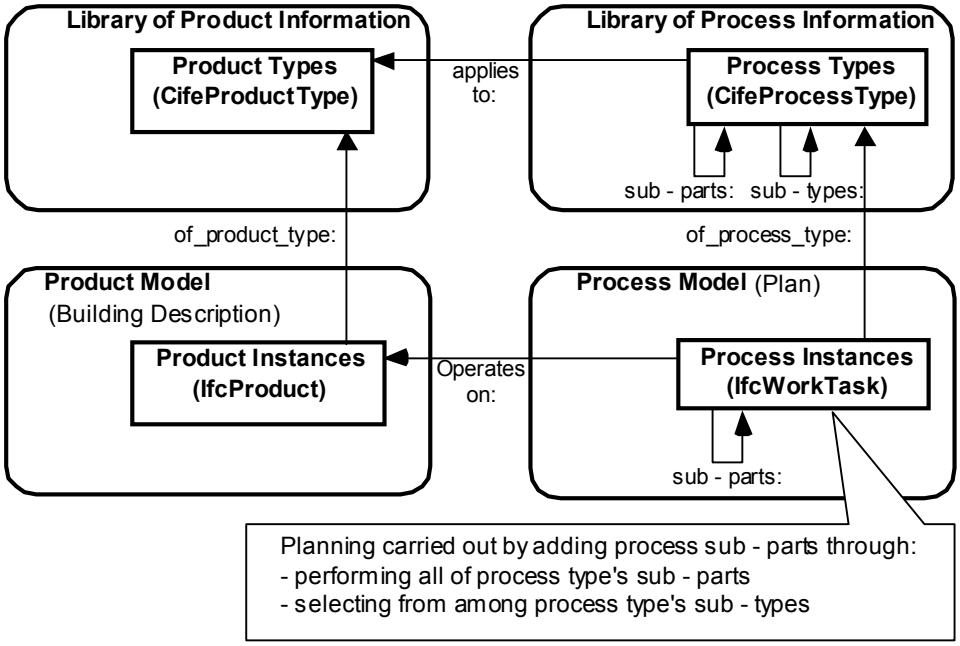

FIG. 6: Planning Algorithm

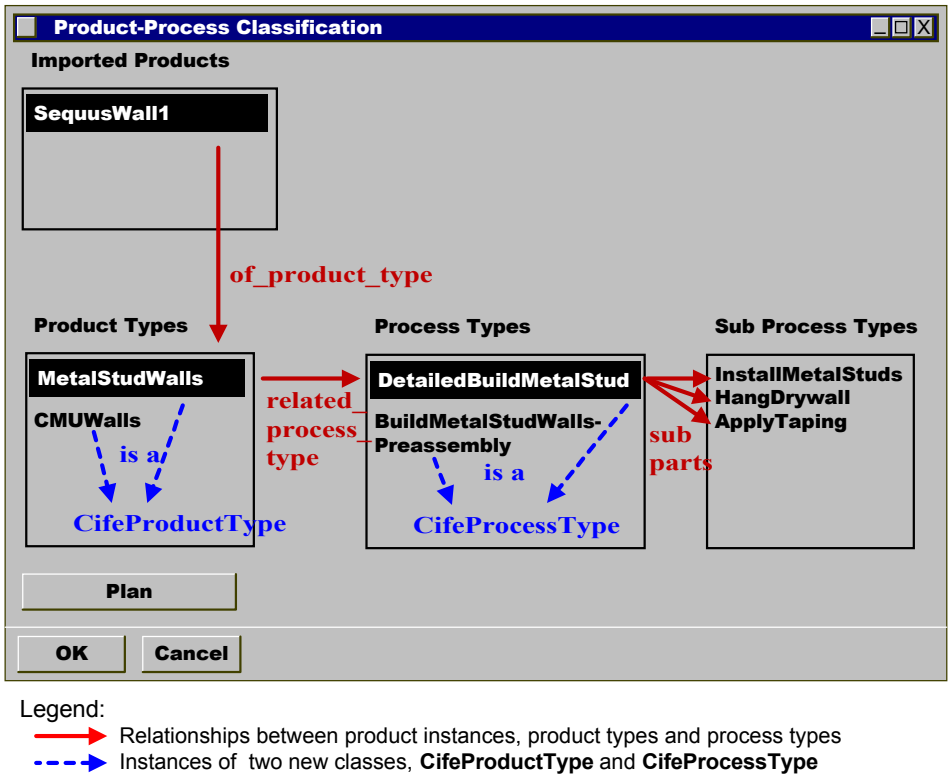

FIG. 7: Annotated User Interface

Fig. 7 shows the initial user interface. We annotated (in blue and red) the actual user interface to show the underlying relationship attributes between the imported building elements (imported products) and their product and process types. When the application is first loaded, only the imported building elements are displayed. When the user selects an imported product instances the system lists defined product types. The user then chooses the appropriate product type and is shown a list of defined process types for the selected product type

instance. Our small prototype system uses these relations to automatically generate construction activities and conceptual cost estimates for the imported building elements.

6. RESULTS OF THE IMPLEMENTATIONS

After completing the implementation efforts described in section 5, we collected all of the issues that arose as a result of difficulties in applying the IFCs to the tasks that we were trying to perform. This section lists these issues and discusses possible solutions from the perspective of the NA PM group.

6.1 Name changes of IFC objects between versions

We found that changes of class names between versions (from IFC Release 1.5.1 to IFC Release 2.0 Beta) were confusing and required redundant work when implementing our test case. Table 5 shows the specific examples of class names that had changed.

TABLE 5: Name Changes in IFC Project Management Classes

IFC R1.5.1	**IFC R2.0**
IfcScheduleData	IfcScheduleTimeControl
IfcRelProcessesProducts	IfcRelProcessOperatesOn
IfcCalendarDate	IfcDateTimeSelect

We initially started our implementation referring to IFC Release 1.5.1 manuals for class definitions and relationships. Due to the changes in IFC Release 2.0, we had to refer once again to the updated manuals to verify whether the definitions and relationships of these classes had been modified. We also had to change the names of classes that we had previously created in the PowerModel system.

We used IFC Release 2.0 documents that described newly defined classes and name changes of classes and relationships. To describe more types of objects in greater and more realistic detail, new class objects will need to be created as the IFCs develop. For example, the IFC Release 2.0 has defined objects such as IfcDoorPanel and IfcDoorLining, whereas the prior version only had IfcDoor. However, changes to the names of existing classes cause confusion and additional work, and should be avoided if at all possible. If changes are inevitable, the names that changed and the reasons for the change should be documented as well.

6.2 Decomposition of Product and Process models

The IFCs have defined four different objectified relationships to decompose product models:

1) IfcRelNests defines the general concept of elements being nested so that the nest is of the same type (or supertype) as the nested elements.

2) IfcRelContains defines a hierarchical containment relationship between elements, i.e., each element can only be contained by one instance of an element container (i.e. site, building, building storey or space).

3) IfcRelGroups handles the assignment of group members to group objects.

4) IfcRelAssemblesElements is an objectified relationship that assembles various Elements (or other Element Assemblies) into an Element Assembly. This relationship is defined at the IfcBuildingElement level.

Fig. 8 shows how two of these objectified decomposition relationships can be used to decompose the Sequus Wall test case.

For the process model decomposition, three relationships are inherited from the IfcObject level: 1) IfcRelNestsProcesses (a subclass of IfcRelNests), 2) IfcRelContains, and 3) IfcRelGroups. However, IFCs suggest the decomposition of processes through IfcRelNestsProcesses, which represents the relationships between a work task and its subtasks. This objectified relationship also provides information, such as nesting

purpose and nesting criteria, and hence makes the nesting process flexible. Fig. 8 shows how we used the IfcRelNestsProcesses relationship to decompose the installation of Sequus walls into lower level activities. As shown in the figure, IfcRelNestsProcesses is used to decompose work tasks according to both is_a relationships (e.g., BuildWall1 is_a BuildWalls) and part_of relationship (e.g., InstallMetalStuds for Wall1 is part_of BuildWalls).

The product and process model decompositions for the Sequus Wall case (Fig. 8) demonstrate that similar types of decompositions for the product and process models are handled with two different decomposition relationships in the product model and are handled as one decomposition relationship in the process model. We wondered whether product and process model decompositions should be represented similarly by defining additional decomposition relationships for IfcProcess or whether the process model decomposition should be kept flexible by defining one decomposition relationship and representing the purpose of the decomposition within each relationship instance. Categorising different decomposition relationships (as in the product model) will explicitly state the content of the decomposition. For example, aggregating the duration for BuildWalls will have a different value and meaning if it is aggregated through is_a relationships (e.g., BuildWall1 and BuildWall2 are BuildWalls), versus if it is aggregated through part_of relationships (e.g., InstallMetalStuds, HangDryWall, ApplyTaping are part of BuildWalls). During the workshop, we were inclined towards defining one objectified relationship, IfcRelNestsProcesses, for decomposition of process models.

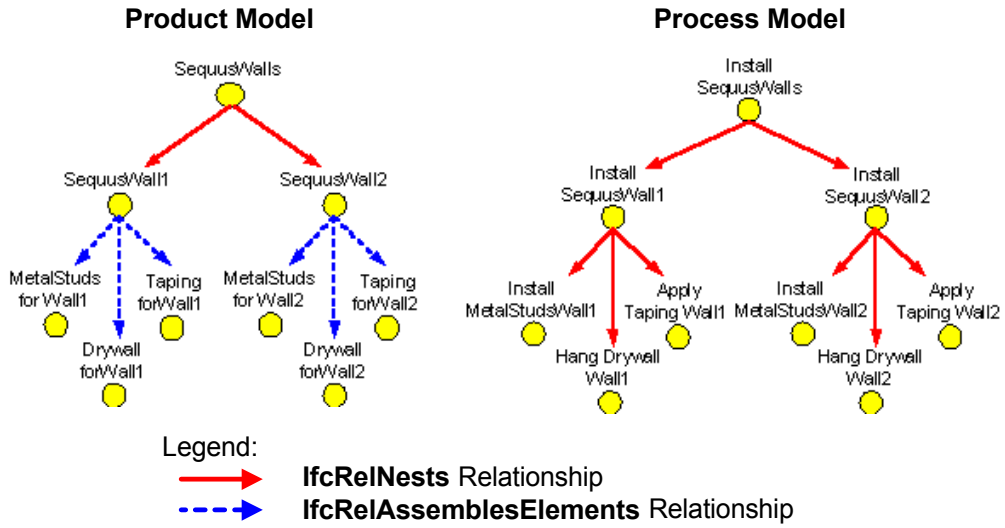

FIG. 8: Product Model Decomposition and Process Model Decomposition

6.3 Modelling of Product and Process Types

As discussed previously in the context of implementing a planning algorithm, the product type of a building element determines the unit cost used for conceptual cost estimating and the required construction activities and resources used for scheduling. For example, the activities and resources used for construction of a metal stud wall differ from the ones used for construction of a concrete masonry block wall. Similarly, the unit cost of a metal stud wall is different from the unit cost of a concrete masonry block wall. Consequently, the cost and duration of building a wall are different if it is a metal stud versus masonry block wall, even though the geometric features are the same. The IFCs do not currently have the capability of representing these "type" objects.

6.4 Assigning Quantity Information

It is unclear where material and resource quantities should reside. There are attributes for quantity information on the work task, the re-ified relationship (that is, a relationship between objects that is itself modelled as an object) between the product and the work task, the cost element, and the re-ified relationship between the work task and the resource, as shown in Fig. 9. This leads to the problem of creating inconsistencies as information is

shared. For example, one application, such as an estimating program, could consistently put all quantity information on the cost element whereas another application, such as a scheduling application, could put all quantity information on the work task. The main underlying problem is that there is no way to distinguish one quantity from another. For example, when looking at the work task and its related objects, the distinction between the different quantity attributes that are on IfcRelProcessOperatesOn, IfcWorkTask, and IfcCostElement is unclear (see Fig. 9). This same ambiguity exists for identifying quantity information for labour and material resources associated with each work task. The difference between the quantity attributes on IfcRelUsesResource and IfcCostElement is unclear. To solve these problems, we suggest that an additional attribute be added to each of the objects to define the context of each quantity.

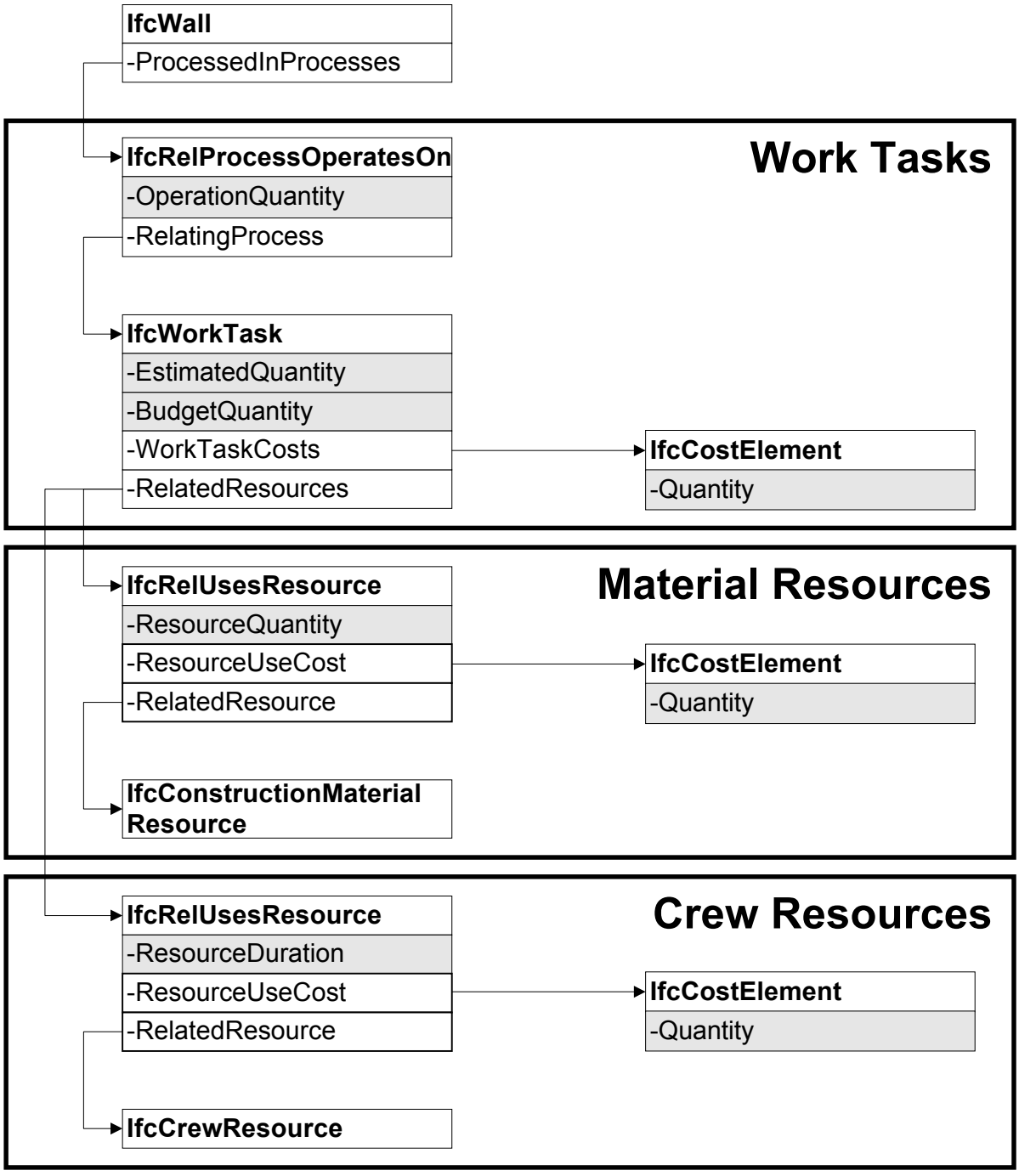

FIG. 9: Assigning Quantities

6.5 Inconsistent Re-ifying of Cost Relationship Objects

The re-ifying of cost relationship objects is not consistent. The relationship from a product object, such as IfcWall, to its corresponding cost is re-ified. On the other hand, the relationships from resource, work task, and resource use objects to their corresponding costs are not re-ified (see Fig. 10). In implementation, these inconsistencies require different methods for calculating costs on a product object and for calculating costs on work tasks, resources, and resource use. This issue could be resolved by re-fying all cost relationships using the IfcRelCostsObjects object.

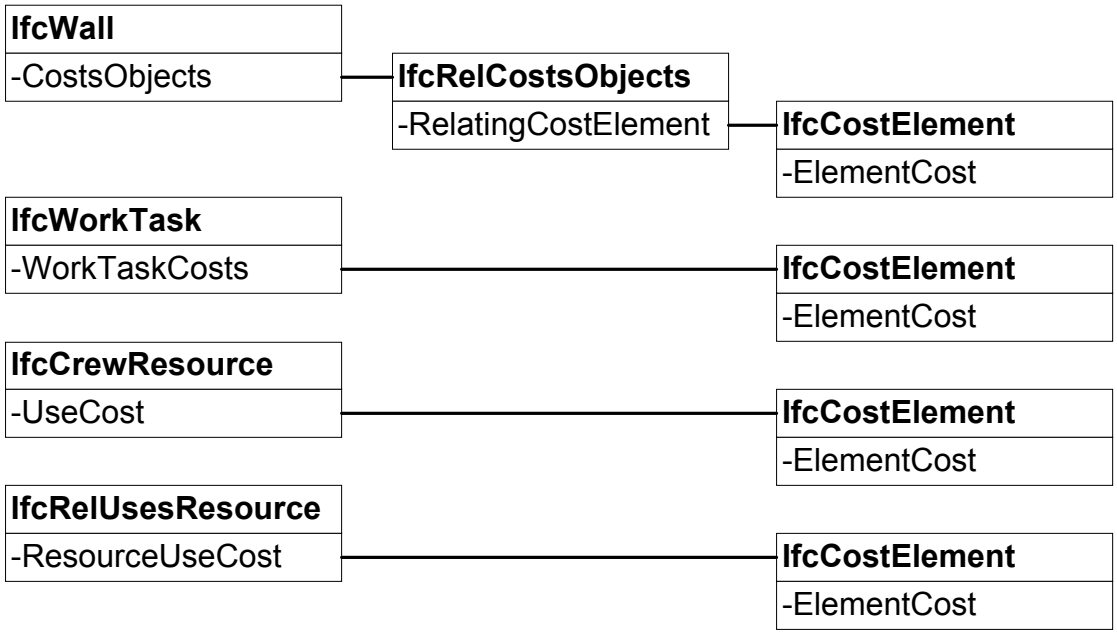

FIG. 10: Cost Relationships

6.6 Use of Multiple Cost Schedules

Cost schedules are used for grouping all types of cost information. For example, different cost schedules could exist for unit cost databases like metal studs or cost estimates for a specific project. Fig. 11 shows two different cost schedules that were used for modelling crew costs on the test case. The cost schedule related to IfcCrewResource contains the unit cost information for different crews. The cost schedule for IfcRelUsesResource represents the cost of this resource use in a process for a certain project, which would reside in the project estimate. The only way to distinguish one cost schedule from another is through the Title attribute, which is just a text string. We feel that this would create difficulties in managing different types of cost information. To solve this problem, we recommend one of two alternatives:

1) Create a new cost object, such as IfcCostEstimate, to distinguish project specific information from unit cost information. IfcCostSchedule would then become the object for grouping unit cost information. Modelling cost information in this way would also fit more closely with the way that the industry views this information. In practice, cost estimates and cost databases are viewed as different types of cost information and are maintained separately.

2) Add an attribute, such as CostType, on IfcCostSchedule to distinguish between unit cost information and project estimate information. The attribute value could then be an enumeration of different cost types.

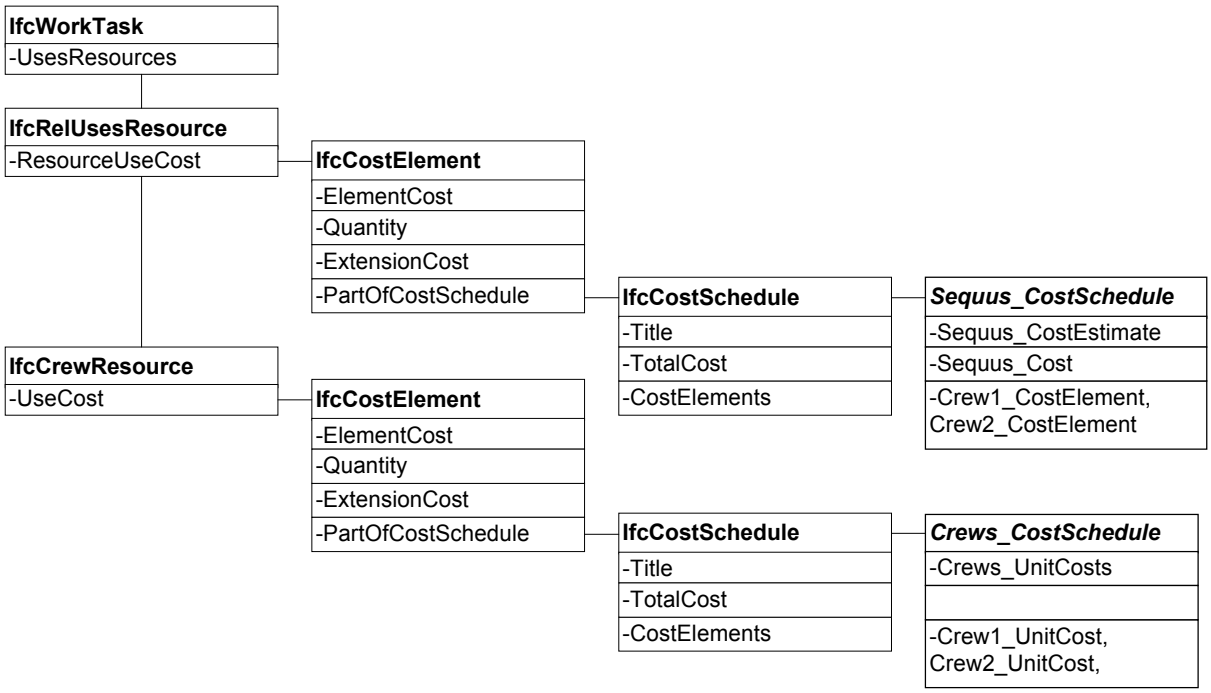

FIG. 11: Cost Schedule Objects

6.7 Modelling relationships between products

The IFC model did not allow us to model dependency relationships such as "supported by", or "enclosed in" between products. For example, there was no way to model the relationship that the metal studs are enclosed by the drywall for the wall.

We did not need to model these relationships for our test case. However, dependency relationships are essential for applications that use these relationships to infer precedence between instance objects. The IFCs reify relationships between objects by using the IfcRelationship class and its subclasses. However, there is no subclass that describes the dependency relationship. New subclasses of the IfcRelationship class should be developed that reify the dependency relationships between products.

6.8 Model decomposition criteria

There are no well-defined criteria in the IFCs describing the circumstances under which subclasses are created from building elements. For example, IFCWindowLining is a subclass of IFCBuildingElement, while various parts of walls are not explicitly defined as subclasses. The IFCs should provide guidance to make the implicit information about the decomposition criteria explicit. The criteria to decompose an object into its components should be stored in the IFCs. If available, it would eliminate the need to represent every detailed component and help the rapid modification of the geometry.

In the Sequus wall test case, the wall is decomposed into 5/8" drywall elements. We used IFCConstructionMaterialResource objects to represent each drywall element. Drywall may be a subclass of IFCBuildingElement in future versions of IFC. However, it is not practically feasible to create an instance for each drywall element, each with its own geometrical representation. An alternative would be the specification of the decomposition criteria in a separate class, which, in this case, specifies that each wall is decomposed into sheets of 5/8" drywall. Estimating, construction planning, and 4D-simulation software can then use this information to automatically generate relevant information about the drywall.

7. CHANGES MADE TO THE IFC MODELS

Prior to the workshop, there was no way to create instances of products when they were not defined as specific sub-classes of IfcBuildingElement. For example, in the case study presented here, there was no way to make

instances of a wall stud as a type of IfcProduct or IfcBuildingElement. This was a significant limitation because it meant that these components couldn't be represented explicitly in the building model and their quantities, geometry, etc. could not be expressed. As a result of this workshop, this problem has been solved by making IfcBuildingElement a non-abstract class.

8. CONCLUSIONS

The results of the trial implementation discussed in the previous section suggest potential improvements to the IFC models. This section presents our final, overall conclusions drawn from the exercise.

8.1 Recommendations for Use of Charrette Test Method

Exercises like the one described in this paper should be done earlier in the model development process to help guide the development of the use cases and the model. Detailed, specific test cases also help document the intent of classes and attributes. Since the class definitions and attribute names often change along the way and since many parties develop the standard, it is important to document the intent of each class and attribute in an easily understandable way. Charrettes could augment the model development process in the following ways:

1) The model development process could start with a charrette that brings together users and decision-makers to define test cases and use cases in parallel.

2) The model could be developed with frequent references to the use cases and test cases from the charrette.

3) The model could be tested in a second charrette that includes modellers and users.

As pointed out by the IAI facility management domain group (CIFE 1999), and by Clayton et al. (1998), a clear and appropriate scope definition for a charrette test case prior to running the charrette is essential for the charrette's success. All participants in the charrette presented here felt that the scope of the test case we used was about right. It is simple enough so that the participants could get their hands around it easily and quickly and keep track of what everyone else was doing. The number of objects--two walls that decomposed into three types of sub-components leading to a handful of estimating items and schedule activities--worked well. More objects would have required more modelling work, leading to longer test cycles and longer discussion periods without necessarily adding new evidence for the usefulness of the IFCs. For our case, it was important that the test case was developed at two levels of detail to give us some confidence that the IFCs work at more than one particular level of detail. Because the case was taken from an actual construction project, it brings out many of the aspects that are important for estimating and scheduling. It is also important that the case is worked out in complete detail, including specific and traceable values for all inputs and outputs.

8.2 Advantages of Open Data and the Role of the IFCs

Without IFCs, it is difficult to share information on a project between the owner, designers, general contractor, subcontractors, vendors, etc. because they all use discipline-specific software tools, and it is not likely that data translators between all these software packages will exist. For example, the estimating software market in North America is quite fragmented with a clear market leader and with many specialised estimating tools. The IFCs will make it more likely that an estimating tool used by a mechanical contractor will be able to share estimating information with the general contractor's estimating tool. This should lead to a faster generation of alternative scenarios and to a faster detection and resolution of conflicts. Hence disciplines will be able to take advantage of specialised tools and integrate with other participants.

8.3 The Use of IFCs to Support Project Management Tasks and Applications

The UML models, screenshots, and discussion show that the project management portions of IFCs Release 2.0 generally support the current practice of estimating and scheduling. Hence, developers of project management software should be able to write translators to the IFCs to allow project participants to share planning, scheduling, and estimating data.

In addition, the IFCs make associations between scope, estimate and schedule information explicit, enabling project participants to integrate estimating and schedule information better than is possible today. Estimates and schedules can be synchronised, and changes in the product model (design changes) or changes in resource assignments now propagate to estimates and schedules. The IFCs also document cost and schedule assumptions better, allowing several people to collaborate on estimates and schedules. This is difficult to do with today's estimating and scheduling tools. Links between information items often get lost in the estimates and schedules produced today. The IFCs allow software vendors and project participants to represent information about estimates and schedules explicitly and share it with other participants and other applications.

9. REFERENCES

CIFE (1999). Facilities Management: Move Management Charrette, [online] http://www.stanford.edu/group/CIFE/IFC/CIFE-CharretteWhitePaper.html. Accessed March 22, 1999.

Clayton, M.J., Fischer, M.A., and Kunz, J.C. (1998). CAD Prototype Testing: Worked Examples, Demonstrations, Trials, and Charrettes." Computing in Civil Engineering, Proceedings of International Computing Congress, Boston, October 18-21, Kelvin C.P. Wang (Ed.), ASCE, 106-116.

Cole, M., et al. (1998) "IFC Model Requirement Analysis, ES-1: Cost Estimating", IAI Project Document for IFC R2.0.

Froese, T.M. and Yu, K.Q. (1999). "Industry Foundation Class Modeling For Estimating And Scheduling" Durability Of Building Materials And Components 8. National Research Council of Canada: Ottawa. Vol. 4, pp. 2825-2835.

Grobler, F., et al. (1998). "IFC Model Requirement Analysis, PM-1: Scheduling", IAI Project Document for IFC R3.0.

IAI (1996). "End User Guide to Industry Foundation Classes, Enabling Interoperability in the AEC/FM Industry", International Alliance for Interoperability (IAI).

IAI (1998). "International Alliance for Interoperability (Home Page) [online]". URL: http://www.interoperability.com/. Accessed March 22, 1999.

Intellicorp (1999). Knowledge-based Tools. [online] http://www.intellicorp.com/prod_kbt.html#PowerModel, accessed March 22, 1999.

ISO (1994). "Industry Automation Systems–Product Information Representation and Exchange – Part 1, Overview and Fundamental Principles", ISO TC184/SC4 Project Management Advisory Group.

Laitinen, J. (1999). "Model based construction process management," Durability Of Building Materials And Components 8. National Research Council of Canada: Ottawa. Vol. 4, pp. 2844-2863.

Sowa, J. (1984). Conceptual Structures: Information Processing in Mind and Machine. Addison Wesley.

Staub, S., Fischer, M., and Spradlin, M. (1998). "Industrial Case Study of Electronic Design, Cost, and Schedule Integration." Working Paper, Nr. 48, CIFE, Stanford.