A Component-Based Framework for Integrated AEC/FM Project Systems

Mahmoud R. Halfawy and Thomas Froese

This article was originally presented at the:

Congrès annuel de la Société canadienne de génie civil / Annual Conference of the Canadian Society for Civil Engineering

Montréal, Québec, Canada

5-8 juin 2002 / June 5-8, 2002

https://csce.ca/en/publications/past-conferences/

Congrès annuel de la Société canadienne de génie civil

Annual Conference of the Canadian Society for Civil Engineering

Montréal, Québec, Canada
5-8 juin 2002 / *June 5-8, 2002*

A COMPONENT-BASED FRAMEWORK FOR INTEGRATED AEC/FM PROJECT SYSTEMS

Mahmoud R. Halfawy[A] and Thomas Froese[A]
A Department of Civil Engineering, University of British Columbia, Canada

ABSTRACT: This paper presents a component-based framework that provides a set of reusable components to facilitate the implementation of distributed AEC/FM integrated project systems. The framework addresses the specific characteristics and requirements of AEC/FM projects, and provides reference architecture as well as a development platform for implementing integrated AEC/FM project systems. The framework defines three-tier architecture: the AEC/FM applications tier, the common domain services tier, and the data/knowledge repository tier. The applications tier includes a set of function-specific software tools. Legacy software tools can be integrated into the framework by implementing adaptors. The common domain services tier defines a number of components that implement a wide range of generic services such as data management, process management, and project team collaboration. The data/knowledge repository tier represents a store for the project information and an AEC/FM knowledge base that can be used by users and applications. The framework supports tools interoperability and data sharing through the use of the standard IAI Industry Foundation Classes (IFCs). The paper discusses the framework architecture as well as several issues related to the development of integrated projects systems. A prototype integrated project system is also presented.

1. INTRODUCTION

Architecture, Engineering, Construction, and Facilities Management (AEC/FM) project processes comprise a wide range of activities to plan, design, construct, and manage a facility. Fragmentation of the AEC/FM industry has caused many problems that could be primarily attributed to the "gaps" between the design, construction, and facility management phases of a project. Project information typically flows from the design phase to the construction to the facility management phase, with very costly and time consuming feedback loops in the form of change orders during the construction phase, or excessive maintenance work during the facility management phase. Project gaps have often resulted in project cost and time overruns, reduced quality and maintainability, loss of design intent, and the inability to access and communicate project information in a timely fashion. More frequently, this has resulted in contractual disputes, claims, and litigation, and sometimes in failures and life loss.

Experience with the project gaps and the fragmentation of the industry has created an increasing demand for integration. In recent years, the AEC/FM industry has been trying to find solutions through adopting integrated project approaches. Most notable is the use of design-build and partnering as a way to integrate the design and construction processes. Many study groups and committees were formed to analyze these problems and needs. The AEC/FM industry has been steadily moving towards adopting methods that can potentially integrate different project processes to overcome the problems associated with these gaps, to reduce design-construction cycle time, and to improve life cycle performance and maintainability of constructed facilities. The demand and the trends for integration are expected to become increasingly pervasive throughout the industry.

It is widely recognized that adopting a Concurrent Engineering (CE) approach to AEC/FM projects would potentially alleviate many of the problems associated with the project gaps. In essence, this approach emphasizes addressing various AEC/FM project life cycle issues, including construction and facilities management aspects, at early design stages (Halfawy, 1998). However, adopting a CE approach would require extensive computational support in the form of integrated project systems that can address various aspects of AEC/FM projects life cycle. With the increase of projects complexity and sophistication, integrated project systems are becoming more of a necessity to improving the quality, and reducing the time and cost of such projects.

Developing integrated AEC/FM project systems is an area of research that has seen a surge of activity in recent years. Froese et al. (2000) defined a reference architecture for distributed AEC/FM project systems. The Construction Modeling and Methodologies for the Intelligent integration of Information (COMMIT) project defined object-oriented models of the rights and responsibilities, notification, versioning support and representation of design intent (COMMIT). The ToCEE (Towards a Concurrent Engineering Environment) project addressed the issues of product and document modeling, conflict management, version management, and cost control (ToCEE). CONCUR is another project that aims to integrate design, engineering, and construction support tools, and to implement concurrent design and engineering in a distributed, multi-partner projects.

In this paper, we present a computational framework that aims to support the implementation of integrated AEC/FM project systems. The framework supports the integration of project processes by adopting a concurrent engineering approach. The framework serves as an infrastructure to support managing and controlling large-scale AEC/FM projects. The framework defines a flexible architecture that can be adapted to specific classes of AEC/FM projects, supports sharing and management of the project information, enables the integration and interoperation of legacy software applications, and allows the collaboration of project teams.

2. REQUIREMENTS AND DESIGN CONSIDERATIONS OF AEC/FM FRAMEWORKS

Integrated project systems are complex software environments that need to address a wide range of requirements inherent in AEC/FM projects. The usefulness and effectiveness of a framework would primarily depend on how it addresses and meets each of these requirements. An integrated AEC/FM project system would be required to support: (1) management of project information across all project disciplines and throughout the project life cycle; (2) interoperation of an array of functional-specific software tools that support various project activities; (3) management and coordination of project activities and workflow; (4) collaboration of project teams; and (5) the ability to customize the system to specific project or organization policies, and to accommodate various operations that reflect industry practices. These requirements have enormous implications on the design of integrated project frameworks.

AEC/FM projects involve handling large and dynamic data sets with complex inter-dependent objects. Probably the most important requirement of a framework is to maintain the consistency and integrity of project data, especially when users could access the data concurrently. The framework should prevent inconsistencies or any violation of the integrity constraints, and should implement procedures to propagate and track all changes to project data. Also, the framework should support industry-wide data modeling standards wherever possible. The framework should support different modes of project data access and

exchange such as file exchange, centralized database, application-to-application data exchange, and online web access.

AEC/FM project processes are becoming increasingly knowledge-intensive activities that require accessing and managing a multitude of knowledge sources. The framework should enable project teams to easily share, manage, and reuse knowledge in their respective domains. The framework should also support efficient representation and management of different forms of AEC/FM knowledge.

AEC/FM projects typically involve tens or hundreds of highly inter-dependent activities that need to be managed and coordinated. The framework should assist in identifying the inter-relationships and to enable the efficient flow of information among various project activities. The framework should also enable efficient access, management and tracking of project documents.

Project teams need to communicate to coordinate their tasks, exchange project information, provide feedback to upstream project activities, and to evaluate the impact of various decisions. Team collaboration involves the communication of different forms of information (e.g. textual, graphical, audio, video) and different interaction modes (e.g. synchronous, asynchronous). Supporting team collaboration is a major requirement for AEC/FM frameworks.

The framework should be able to support the interoperation of various function-specific applications as well as the integration of legacy software tools. This is an important requirement to ensure the framework acceptability by the industry. An important implication of reusing legacy software tools is the reduction of the system implementation and maintenance time and cost. Also, given the availability of a large number of commercial function-specific tools addressing a particular domain, frameworks would be expected to enable upgrading the tool set, either by adding new tools or replacing existing ones, without impacting the overall operation of the framework.

Giving the wide range of possibilities that AEC/M projects could be handled, the framework should be generic, flexible, and customizable to enable the implementation of various organization and project-specific procedures, as well as to accommodate different project delivery systems. For example, design-bid-build projects would generally involve different procedures from design-build projects. The framework should also enable organizations to manage a portfolio of concurrent projects.

The aforementioned requirements have enormous implications on the design of AEC/FM frameworks. A number of design considerations needs to be addressed. The framework should have a modular and flexible architecture. A component-based architecture seems particularly suitable to the domain requirements we outlined above. Also, the framework needs to be flexible to accommodate future modification, extension, and technology improvement. Another major design consideration is the necessity to separate the functionality between the function-specific tool set and the framework components. Tools would provide users with the functionality to perform specific project activity, while the framework components would provide the functionality to integrate and manage different project processes. The framework functionality should be accessible to individual tools as well as to the project management team. Tools would access the framework through standard interfaces, and therefore the framework should make no assumptions about the implementation of individual tools (i.e. the framework is tool-independent).

3. A FRAMEWORK FOR INTEGRATED AEC/FM PROJECT SYSTEMS

A buzzword that has been in use recently and still does not have a clear definition is “framework.” Various camps of researchers often have different notions or views on the definition of “frameworks.” In a recent workshop (Workshop, 2001) on “Theoretical Foundations of Engineering Frameworks,” several, somewhat incompatible, definitions were given by leading researchers in the area. One such definition viewed a framework as a “roadmap” to guide the R&D work towards developing software systems. Another view was that a framework is an “architectural model” of integrated project systems. (Szyperski, 1998) defines a framework as “a set of cooperating classes, some of which may be abstract, that make up a reusable design for a specific class of software.” We believe that these views need to be considered together while

defining a more comprehensive interpretation. In that spirit, we present our definition of frameworks. We view a framework as an architectural model (i.e. a reference model that developers would use to implement the software) and also as a “development platform” that provides a number of common domain-specific services that systems developers can use. Both aspects would provide a “roadmap” to identify the main research issues we need to address, and the technologies we need to develop.

Our notion of frameworks is consistent with all three given definitions. The architectural view of the framework presents a high level description of integrated project systems that deals primarily with the organization, overall functionality, and interfaces of the components. Further refinement of that architecture would yield a blueprint to build integrated project systems. The development platform view of the framework defines a number of common services that are implemented as a set of reusable components that would offer a wide range of runtime generic domain-specific functionality. Tool developers would use the services of these components to ensure the interoperability of their tools and to access the services offered by the framework components. Example of these generic services would include data management, process management, and team collaboration functionality.

This section describes the architecture of a component-based framework that aims to facilitate the implementation of integrated AEC/FM project systems. The main goal of the framework is to achieve efficient coordination and management of project processes through enabling sharing of project information, integration and interoperation of software tools, and collaboration of project teams. The framework defines a flexible three-tier architecture and implements mechanisms that enable the sharing and exchange of project information. The framework adopts a concurrent engineering product-centric approach to AEC/FM projects.

Components are coarse-grained, reusable, software subsystems that implement a specific set of services and could be accessed through standard published interfaces. Components are self-contained subsystems that can be developed, maintained, and deployed independently. The framework comprises a set of components that provides a set of generic AEC/FM-specific runtime services such as data management, process management, and document management. Components interoperate to achieve the overall framework functionality. Components interaction is primarily achieved by accessing interfaces exposed by each component or through the use of a common project information model.

The framework also serves as a “development platform” for function-specific tools developers. The tools could exploit the runtime services offered by the framework components as it fits the domain and functionality of each tool. The framework would also serve as a reference model that application developers can use to enable their tools to interoperate with other products. Project management team could also use the framework as a “project environment” to manage, control, and track various aspects of the project. The framework would be configured to suit the specific project or organizational procedures.

3.1 Approach

A distinguishing characteristic of the framework is that it employs a product-centric approach to support a concurrent engineering methodology. This approach implies the use of the facility product model as the central part to represent project information and linking the aspect models of various project activities (e.g. construction process, cost schedules, specifications sections, etc.) with that product model (Figure 1). The resulting “project model” would represent a comprehensive view of the project centered on the facility product model. The integrated project model would become the glue that binds together various views and perspectives of the project information. Project teams from different disciplines could use the facility product model to access project information in their respective domains and to assess interactions and dependencies between various project activities. The resulting project model would include cross-references to various project data, and thus could be used to manage and communicate the entire project data in an efficient and effective way.

The product-centric approach would potentially achieve many benefits to various project activities. The approach enables efficient sharing of project information and the ability to adapt the project model to different views. A major benefit of using this approach is the ability to view the project models from multiple

perspectives. Making various aspects of project information accessible from the facility product model would significantly improve the communication among project teams. Representing the facility products as 3D objects, though not required by the framework, would enhance various project activities. Using 3D models that are connected to the integrated project model, project teams could walk-through the facility model and construction site, identify potential problems, evaluate the design from multiple perspectives, and interactively access project information as needed. Also, various project activities, such as construction planning, cost estimating, and site planning, could be supported and enhanced using the 3D representation of the facility objects. The 3D graphical representation of the facility objects would also facilitate the communication of project information and enable efficient collaboration of project teams.

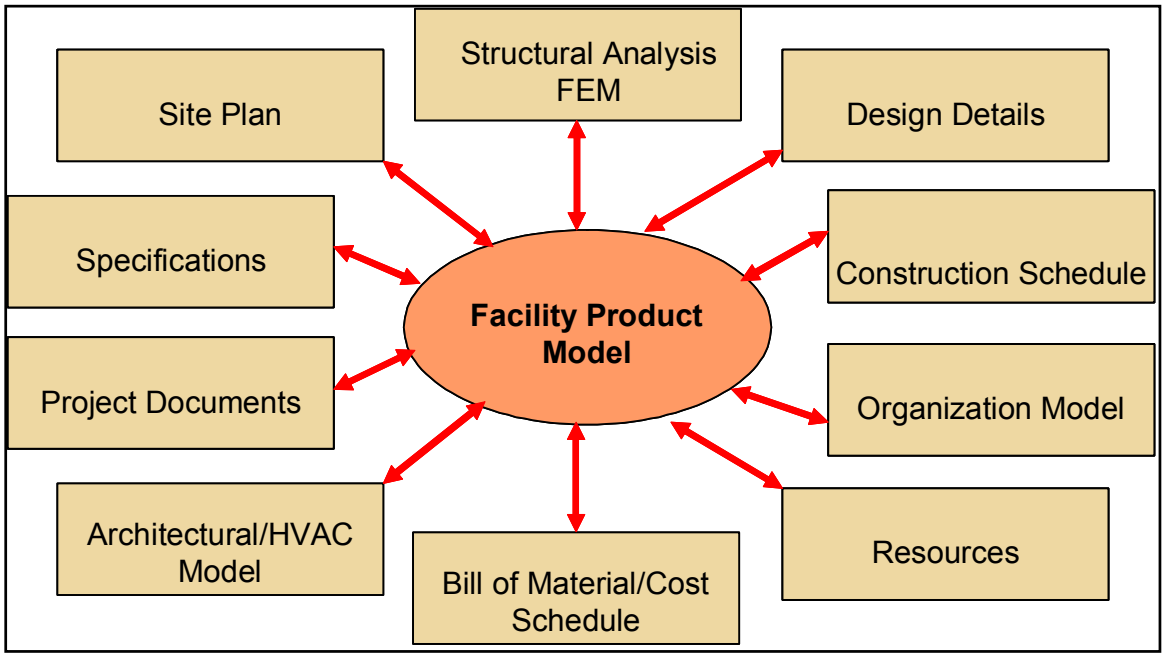

Figure 1: The Model-Based Product-Centric Approach in the Framework

3.2 Three-Tier Component-Based Architecture for AEC/FM Integrated Project Systems

The framework is intended to serve as a reference architecture and infrastructure to support the requirements of developing integrated project systems. The framework should provide a wide range of services such as data management, interoperability, process management, document management, team collaboration tools, etc. The complexity and scope of these functions necessitates their break down into a set of component functions that can be performed by various components. The framework assigns specific groups of functions to specific components, and specifies a number of interfaces for each component. The framework has identified a number of components and organized these components in a flexible architecture that would enable the integration of their functionality. The multi-tier component-based architecture has been chosen because of its potential to support the development of flexible and modular large-scale integrated project systems. The framework architecture has been created by breaking down the framework functionality into a set of specific group of services and then mapping the service groups into a set of coarse-grained domain-specific components to perform these functions; each component (or set of components) provides functionality to implement specific service (e.g. data management). The three tiers include (Figure 2): the AEC/FM applications tier, the common domain services tier, and the data/knowledge repository tier.

The primary purpose of the tiered architecture is to separate the required functionality of the function-specific applications from the common domain-specific functionality. The function-specific tools at the applications tier implement the former, while the common services components at the middle tier

implement the latter. The framework adopts the Industry Foundation Classes (IFC) 2.0 schema for modeling project information (IAI 2002). The IFC model, based on the ISO 10303 Standard for The Exchange of Product Model Data (STEP), provides an abstract and conceptual representation of the structure and organization of project data in the form of a class hierarchy of AEC/FM objects. Adapters serve to map the internal data models of each function-specific tool to and from the standard IFC data model in order to enable applications interoperability and to exchange data in a standardized format.

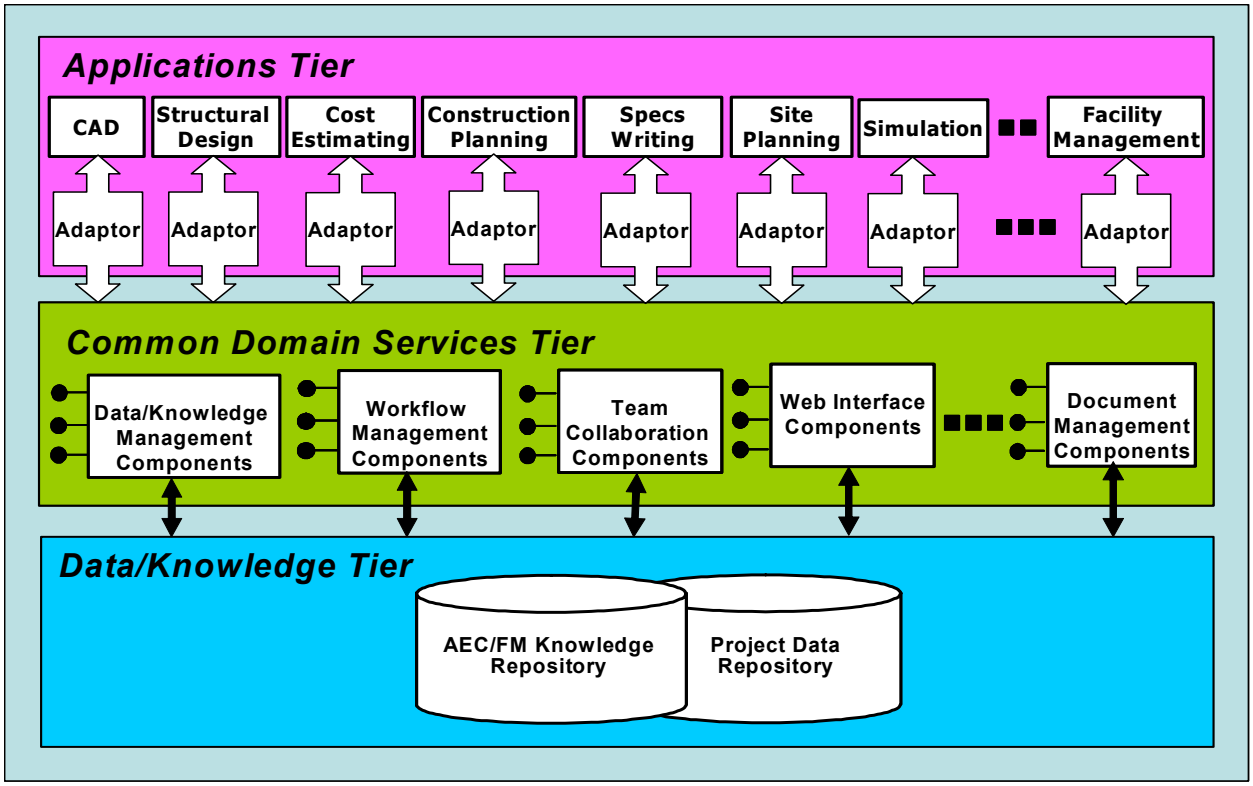

Figure 2: The Three-Tier Component-Based Architecture of the Framework

The common domain services tier is represented by a set of components and interfaces. From an implementation viewpoint, componentizing the domain-specific functionality have several advantages over other approaches: systems will be highly maintainable and can be easily upgraded and extended; upgrading or replacing function-specific tools would have little or no impact on the framework; the framework components could be reused to support the development of specialized integrated project systems that address a particular class of facilities or projects, resulting in reducing the time and cost of developing such large systems and making it more feasible for smaller organizations to deploy. The component-based architecture also helps to maintain the flexibility and extensibility of the framework. An important feature of the components architecture is that parts of the framework could be running in a distributed environment. That is, components and applications could be running over a cluster of distributed workstations over the Internet. Most existing component technologies (e.g. DCOM, CORBA) provide mechanisms to shield the details of data marshaling and accessing remote components so that the access can be performed transparently as if the components are residing on local machines.

The project data repository contains the specific project objects instances. The repository is typically implemented on top of a Database Management System (DBMS) that could potentially support distributed data sources and implement concurrency control mechanisms. The repository schema is generated using an integrated project schema such as the IFC schema. Since IFC is an object-oriented schema, implementing the repository using an object-oriented DBMS would provide benefits over using a relational

DBMS which would require mapping of the object-oriented schema to a number of relations (i.e. tables), and vice versa, with no direct correspondence between the relations and the actual project objects.

The knowledge repository would serve to enable the capture, representation, sharing, and reuse of domain knowledge to support the increasingly knowledge-intensive project activities. Developing such repositories would require systematizing AEC/FM knowledge and representing this knowledge in a computable form. The knowledge repository is distinguished from the project data repository in that the former contains general domain knowledge such as design standards, products catalogs and specifications, design heuristics, organization-specific policies and work practices, while the latter contains data specific to a particular project.

5. CURRENT STATE OF FRAMEWORK IMPLEMENTATION

This section describes the ongoing effort to implement a prototype integrated project system to demonstrate the application of the proposed framework. It is worth noting that the concepts embedded in the framework could be adapted and applied to address different types of facilities (e.g. buildings, bridges, etc.). However, although AEC/FM projects share many characteristics, they also have significant differences that need to be addressed at the implementation level. The prototype system is developed to support buildings projects.

There are many different methods and technologies that could be used to implement a component-based architecture. Examples include CORBA, JavaBeans, or COM/DCOM. In our implementation effort, we tried to base our prototype on technologies that are widely available in the AEC/FM industry in order to reduce the development and deployment costs of such systems. We implemented a set of middle-tier binary interoperable components using the Microsoft COM/DCOM technology. We are currently investigating the use of .NET since it promises a more robust, extensible, and stable development platform. The components are programmed using Visual C++ and Visual Basic languages.

The AEC/FM applications tier integrates a number of function-specific software tools to extend the functions provided by the framework into specific project domains. The function-specific tools integrated into the prototype have been selected to support architectural design (Architectural Desktop), construction scheduling (Microsoft Project), and cost estimating (Timberline Precision Estimating). Other tools are being developed to support specification writing, construction progress simulation, and facilities management.

The core functionality of the framework is to be implemented at the middle-tier common services components. Most of the implementation work to date has been focused on the data management components. The prototype currently supports data sharing and tools interoperability in the form of file exchange. Applications can import/export project data using the standard IFC data model schema. Documents can be formatted either as STEP Part 21 Physical Files (SPF) or as XML files. The data management components are used by different applications to interface with the IFC integrated project schema. The components also enable mapping of project information from one view (or discipline) to another and hence allow the exchange of project information between project team members representing various disciplines within the project. The data management component also supports accessing a wide range of generic data sources such as relational database management systems (through the use of the standard ODBC interface) or XML files (using the XML DOM interface). The component supports transparent access to local or remote data sources.

Applications could be specifically developed to support the standard IFC data model. These applications could be integrated into the framework without the need to use adapters. Legacy applications, however, would require the use of adapters to map their representation of the project data to the standard IFC schema. The implementation detail (e.g. language, methodology) of the adapter depends on the specific Application Programming Interface (API) provided by the application as well as on the data-mapping interfaces exposed by the data management components. Since the framework components are developed as COM components, adapters could access the interfaces of these components, which are

published through the type library of each component. For example, the Architectural Desktop adapter is developed using the Object ARX API, and accesses the framework data management components after importing their type libraries. Figure 3 shows a sample project view inside this adaptor. Users could navigate through the IFC project model using the "project explorer" interface. This interface provides a hierarchical view of various pieces of project information such as the facility product model, resources, schedule, cost estimate, specifications, and documents. Users could also associate pieces of information between different views to indicate "relationships" between different project elements. For example, users could drag schedule activities and drop them onto facility products to indicate that a particular product is dependent on one or more activities. As a result, the construction process could be visually simulated.

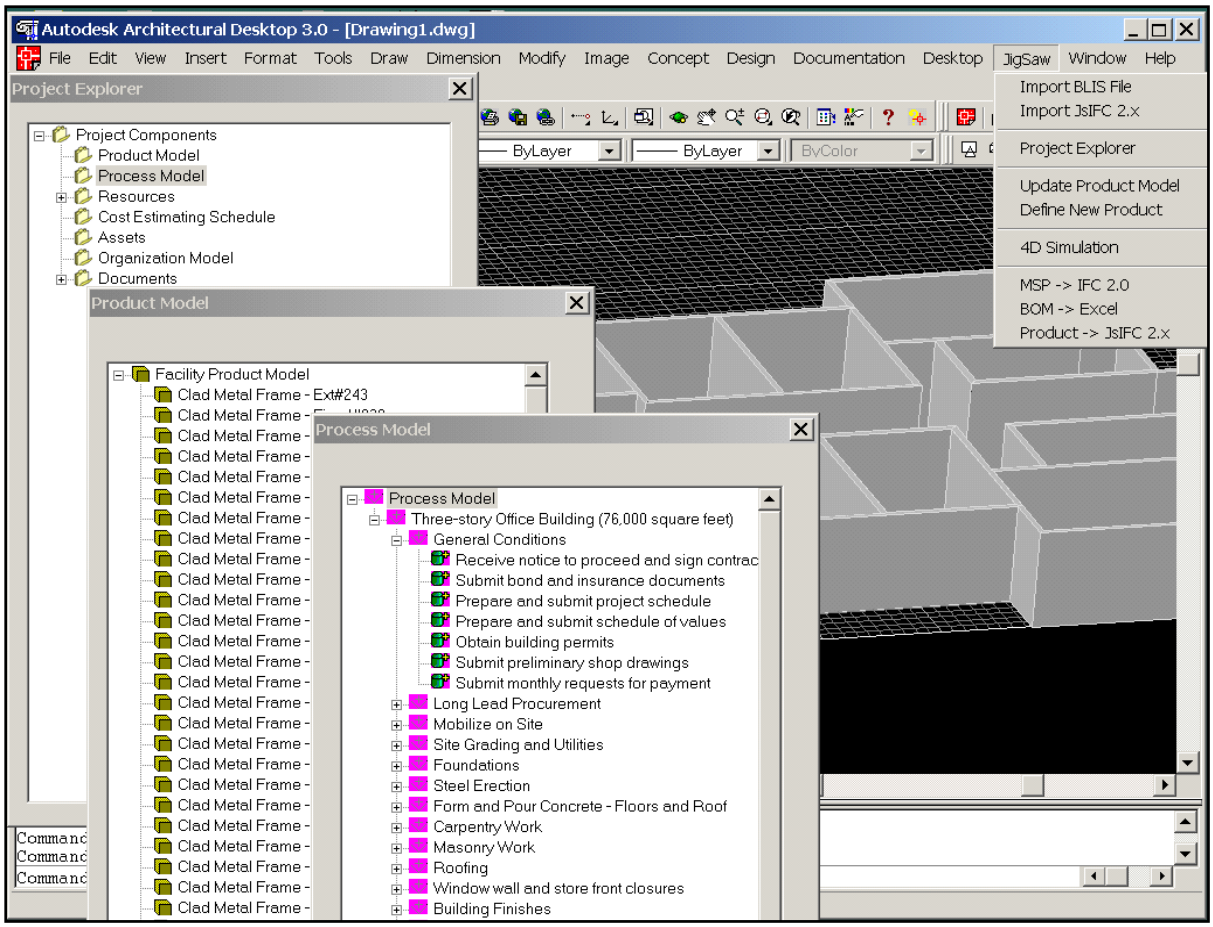

Figure 3: Different Project Views of the Integrated IFC-Based Project Model inside the CAD Adaptor

Implementation work is ongoing to develop the workflow management and the knowledge management components. The workflow management component emphasizes defining the basic coordination mechanisms, in the context of AEC projects, which could be used to organize and manage the interdependencies among various project activities and to ensure efficient flow of project information across all activities. It also aims to define a standard for AEC/FM transactions to formalize the context and requirements for specific technical and business data exchange transactions.

5. CONCLUSIONS AND FUTURE WORK

The future of the AEC/FM industry is becoming more focused on the integrated multi-disciplinary project aspects. Given the current practices and integration trends in the industry, we believe that the

development of integrated project systems will become increasingly crucial for efficient management and control of AEC/FM projects. Integrated project systems seem to be the key to reducing the design-construction cycle time and improving the projects quality. In this paper, we described a general and extensible framework to facilitate the implementation of such large-scale systems, and presented specific primitives required to realize this framework. The framework supported integrated AEC/FM projects by: (1) Providing the infrastructure and common AEC domain services that is required by various project activities; (2) Supporting the interoperation of software tools and defining techniques to integrate legacy software tools into the framework; (3) Integrating the information models of various disciplines to support information sharing and exchange through adopting a standard integrated project model; and (4) Providing tools to coordinate project activities and to enable the collaboration of project teams. The framework would potentially provide several benefits to AEC/FM projects. It would enable project information to flow across various disciplines throughout the project life cycle. It would also improve the availability, reliability, and consistency of project information, resulting in better decisions and less design iterations and construction rework. It would also help in promoting a multi-disciplinary team-oriented approach to AEC/FM projects.

In light of the proposed framework model, we view the AEC/FM software market to evolve into two main distinct categories: generic AEC/FM components developers, and function-specific tools developers. Tool developers will be using component libraries developed by the component developers in order to access generic domain functionality. The components will help to free tools developers from the interoperability, data management, and process management, and other generic domain specific issues.

In addition to the need to more thoroughly address the many research issues highlighted by the framework, we can identify some industry-oriented research activities that would potentially lead to improving the framework. Field studies are needed to assess the impact of implementing integrated project systems and how they could impact the organization structure, team interaction, communication, and productivity of project teams. We also need to conduct studies to evaluate the costs and benefits of fully implementing and deploying such systems. Only when time and cost savings are clearly demonstrable, the industry will be convinced of the value of integrated project systems. Towards this end, we need to cooperate with the AEC/FM industry to start introducing these systems into actual operation.

ACKNOWLEDGEMENT

We gratefully acknowledge support for this work from the Natural Sciences and Engineering Research Council of Canada, Collaborative Research Opportunities Program.

REFERENCES

Clemens Szyperski, (1998) “Component Software: Beyond Object-Oriented Programming,” ACM Press, New York, Addison-Wesley.

COMMIT, http://salford.ac.uk/iti/projects/commit/

CONCUR, http://ivope.ivo.fi/concur/

Froese, T., Yu, K., Liston, K., and Fischer, M., “System Architectures for AEC Interoperability,” Proceedings of Construction Information Technology (CIT) 2000, Reykjavik, Iceland, 28-30 June, 2000, G.Gudnason (Ed.), Icelandic Building Research Institute, Vol.1, pp.362-373.

Halfawy, M.R. (1998), “A Multi-Agent Collaborative Framework for Concurrent Design of Constructed Facilities,” Ph.D. Dissertation, Department of Civil and Environmental Engineering and Geodetic Science, the Ohio State University.

IAI (2001), “International Alliance for Interoperability, http://www.iai-na.com/

ToCEE, http://wwwcib.bau.tu-dresden.de/tocee/

Workshop (2001), “Theoretical Foundations of Engineering Frameworks Workshop”, http://construction.civil.ubc.ca/events/2001/frameworksworkshop/