Proceedings of the 2009 Winter Simulation Conference
M. D. Rossetti, R. R. Hill, B. Johansson, A. Dunkin and R. G. Ingalls, eds.

AN OVERVIEW OF THE COSYE ENVIRONMENT FOR CONSTRUCTION SIMULATION

Simaan M. AbouRizk

Dept. of Civil & Environmental Engineering
University of Alberta
Edmonton, AB T6G 2W2, CANADA

Stephen Hague

Dept. of Civil & Environmental Engineering
University of Alberta
Edmonton, AB T6G 2W2, CANADA

ABSTRACT

As the degree of complexity in construction systems increases, the concepts of construction simulation must be advanced and formalized in practice as the primary means for designing, analyzing, planning, and controlling construction projects and facilities. Whereas the prevalent approach for simulating construction operations has traditionally been discrete-event process interaction simulation, the authors propose a strategic re-engineering of simulation methods in dynamic Construction Synthetic Environments (COSYE). The High Level Architecture (HLA) is identified as one of the most promising approaches to address the challenges faced by construction simulation and decision support. This architecture has been adopted as the basis of a new generation of modeling and simulation tools. In this paper, the authors propose a model for developing virtual environments to capture all features, resources and processes required to design, build, and maintain a facility, outlining the COSYE environment and federation structure, and describing a test application in construction bidding simulation.

1 INTRODUCTION

The long term vision of the researchers is to achieve a fully integrated, highly automated construction execution environment that can be used across all project phases and throughout the facility's life cycle as articulated in Figure 1. Accordingly, we need to advance the concepts of construction simulation and formalize its use in practice as the primary means for designing, analyzing, planning, and controlling construction projects and facilities, with dynamic modeling at the core of this vision. The outcome will be a seamless integration between various processes required for design, construction, and operations of the built infrastructure. Overall, the researchers seek to develop virtual environments that capture all features, resources and processes required to design, build and maintain a facility. Those virtual worlds will enable levels of planning, optimization, learning and control that are not possible with today's technologies. This research directive rests on advancements in the areas of real-time data collecting and operation simulation modeling. The two advancements are shown as blocks A and B in Figure 1.

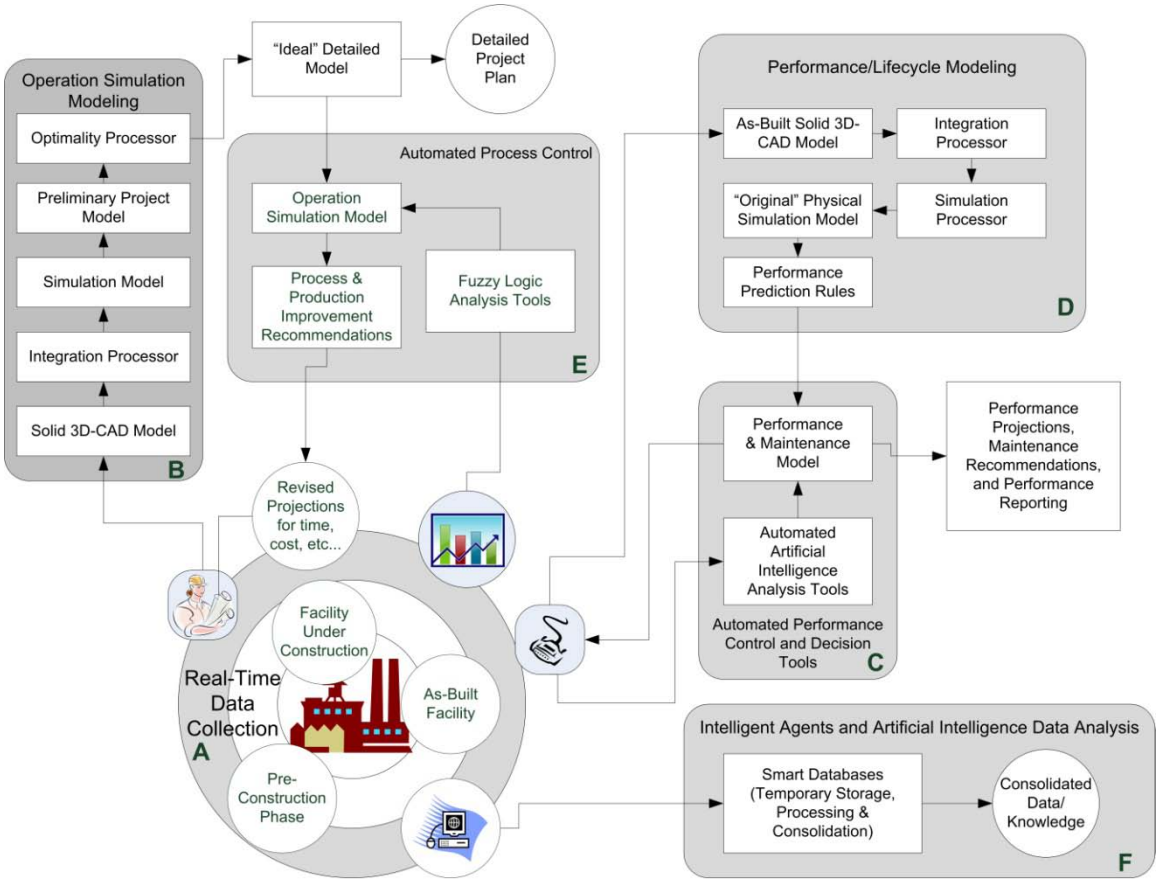

Figure 1: Life cycle of an engineered facility

2 BACKGROUND AND STATE OF THE ART

Simulation in the context of this research is defined as the science of modeling a construction production systems and experimenting with resulting models on a computer. The prevalent approach for simulating construction operations has traditionally been discrete-event process interaction simulation. Using this approach, a simulationist creates a model of a construction operation using specific modeling constructs. This is an inherently complex undertaking in which the simulationist has to describe, in a given simulation language, the production process under consideration. Much advancement has taken place to facilitate the application of discrete-event process interaction simulation in construction.

Halpin's first significant effort in simulation language, CYCLONE (1976), opened up a wide range of simulation work, until Shewchuk and Chang (1991) introduced object-oriented concepts to construction simulation modeling. Object-oriented simulation models resemble their real-life counterparts. The use of the object-oriented approach leads to reduced coding and improved simulation model readability (Oloufa 1993). Marzouk and Moselhi (2003) developed a simulation engine for earthmoving operations utilizing object-oriented features as a component in an automated simulation system. Shi (1999) introduced a simulation language called ABC (Activity-Based Construction), which uses a single element (e.g. activity) for modeling general construction processes instead of multiple elements as required by previous simulation systems. AbouRizk and Mather (2000) were able to further simplify simulation modeling with the introduction of CAD to generate models from high-level descriptions. The automation and generalization of this approach relied heavily on advancements in the area of computer simulation, specifically in the use of modular and hierarchical simulation concepts.

With the RISim (Resource-Interacted Simulation) approach, Chua and Li (2002) proposed a solution to time consuming and knowledge-dependent discrete-event simulation by simplifying and speeding up the model development cycle, so as to enable users without much knowledge of simulation technology to easily generate a model in a relatively short period of time by focusing on the resource and process levels of the model. Kamat and Martinez (2003), introduced VitaScope as a 3-D visualization extension for discrete-event systems. In their work, a 3-D post-processing of a simulation model can take place once the 3-D model is created and the links with the process model are defined.

Many other researchers have focused on 4-D modeling, as simulation methods have continued to evolve, emphasizing time constraints by developing 3-D models and linking them to the time element using a variant of CPM. Zhang et al. (2008) developed an algorithm, integrated with activity-scanning simulation systems, to model time-constrained construction activities. This research has implications for operation planning and analysis.

Aside from commercial simulation software, two construction-specific developments are considered mainstream simulation tools in the construction domain today, namely, STROBOSCOPE and Simphony. Martinez and Ioannou (1999) introduced STROBOSCOPE as a modeling and simulation language capable of modeling a wide variety of systems. It contains implementations of CYCLONE and an open platform for creating models using source code in its open environment. Hajjar and AbouRizk (2002) introduced Simphony as a platform for developing customized simulation tools for construction application as well as for use as a general purpose simulator. Simphony provides customizable visual interfaces for modeling templates, extensibility, and hierarchical modeling. Lu et al. (2003) extended the scope of simulation models to include multiple sites and interactions with HKCONSIM, a simulation system for planning concrete plant operations suitable for use by practitioners and non-experts. Lu (2003) further refined the methodology of discrete-event simulation through a simplified discrete-event simulation approach that extracts the constructive features from existing event/activity-based simulation methods, streamlining both the algorithm and the model structure so that construction simulation achieved simply by applying the critical path method. This approach is particularly geared for adaptation to special purpose simulation tools.

Researchers have begun to look at hybrid modeling, which incorporates more traditional discrete-event approaches with system dynamics, as a valuable tool for comprehensive project planning. Peña-Mora et al. (2008) developed a hybrid model, incorporating both processes, for optimizing performance in large-scale civil infrastructure projects. By considering both strategic and operational aspects, hybrid simulation can produce more complex models better attenuated to construction scenarios.

VIRCON, the interactive system for teaching construction management developed by Jaafari et al. (2001) follows a similar development model, tested and validated through student group projects on construction planning and schedules. This simulation system provides a dynamic way of teaching planning fundamentals. Regarding bidding models, the case-based approach put forth by Chua et al. (2001) suggests a tool that can solve new problems by matching against similar problems that have been encountered and resolved in the past. However, Rojas and Mukherjee (2005) found a number of limitations with special-purpose simulations for training in construction problems, particularly related to related to their application breadth, flexibility, and promotion of collaborations. They proposed a general purpose environment, Virtual Coach, to promote context-based learning.

3 METHODS AND APPROACH

Construction projects are dynamic, subject to randomness, and often complex. They involve many operations and several interacting resources, including equipment, labour, and materials. They take place in a diverse physical environment and involve a wide range of participants that do not generally use same standards for data representation or common standards for information exchange. In order to develop a better understanding of the challenges associated with complex construction systems, we will analyze the application area of construction project bidding.

We have greatly refined our approach to developing construction synthetic environments through our project to develop SuperBid as part of our training CSE. We used its draft development as a means of learning about the complications involved in developing CSEs. We explored CSE concepts and developed the SuperBid training federation as a test case for the application of these concepts. Construction management games have been developed as training tools to improve students' decision making skills and prepare them for managing real and complex construction projects. Bidding games, as one of many simulation applications in construction, try to depict a real bidding process to help users examine different bidding strategies. The working model proves that the concepts will work and shows the advantages of synthetic environments when accessed by multiple users. As well, the flexibility of COSYE in developing this project has given us the security to migrate our other simulation work to this platform as our research program goes forward.

The novelty of this collective approach to delivering research has shown a great deal of success, as the step-by-step process we were following required deferring the application of our results to a practical, usable tool to a future date.

4 MODELING FRAMEWORK FOR INTEGRATED AND AUTOMATED CONSTRUCTION SYSTEMS

Better simulation environments are needed to facilitate more complex modeling situations that will not limit the simulationist in the same ways as today's systems. By carefully analyzing existing limitations, we are certain that the simulation approaches used in construction today need to be strategically re-engineered on a fundamental level. A search for simulation technolo-

gies used by other industries that can meet such challenges determined the High Level Architecture (HLA) to be one of the most promising approaches to address the challenges we have identified. We have studied this architecture and have adopted it as the basis of our new generation of modeling and simulation tools.

To facilitate the implementation of our new modeling framework we need to tackle three issues: 1) researching and designing proper technologies to facilitate establishments of construction synthetic environments, 2) standardization and 3) building intelligence into the environment to facilitate automated decision support.

4.1 Construction Synthetic Environments

The new simulations we intend to develop can be described as "Construction Synthetic Environments" (CSEs) to differentiate them from the low-end simulations we carry out today. CSEs mirror the vision in Figure 2 for a given application and will achieve an integrated representation of all processes, provide means for automation, and incubate intelligence through their design. For example, the tunneling CSE will include the 3-D models of the tunnel and its components, all primary and secondary processes involved in the building of a tunnel, a representation of the environment where the project takes place, all decision-making processes, automated data acquisition from the site, intelligent corrective actions, and a repository for housing the information for use in maintenance and rehabilitation. The implementation of this framework in a software environment will provide tools that others can use to develop specific synthetic models, which can then be used to study construction systems, improve their performance, and provide decision support.

The research under this objective is mostly focused on furthering our research in simulation systems and in implementing the findings in a software environment that will be the foundation for building CSE applications. The implementation of the framework in software will be referred to as COSYE (**Co**nstruction **Sy**nthetic **E**nvironment). The framework itself will be based on the High Level Architecture (IEEE 1516) standard for developing large scale models. The HLA approach is suited for complex applications such as the ones we are dealing with in construction. The HLA architecture supports building complex virtual environments (called "federations") using distributed simulation technologies. In addition, it provides standards for building the individual components ("federates") of such environments by different developers while maintaining interoperability between them. The HLA standards facilitate the reuse of the developed components as part of the new federations. We will build the "Run Time Infrastructure", the required core simulation services, and a modeling toolkit specifically suited for construction applications and implement those in the COSYE to later use in building the desired CSEs.

4.2 Standardization

The uniqueness of the construction product poses many difficulties in modeling, as each project requires a multitude of differing components to produce a built facility. There are several existing product models or construction concept models (e.g. CAD models, UniClass, Talo90, and NAICS), but none of these models particularly serve simulation purposes. Additionally, most of them are classification systems, instead of being object-oriented. In this research, we seek to understand the requirements for standardizing data models for product information. To achieve this task, we will undertake concurrent approaches. First, we will build upon the standardization efforts of leading researchers in the field (e.g. Froese 1999) and international standardization committees. Secondly, we will investigate and develop data-mining systems with sufficient intelligence to capture design information from electronic shop and construction drawings databases, interpret this data, and associate it with processing information (possibly partially automated) to use in the CSE. Groundwork along those lines was completed by El-Ghandour (2007).

5 OVERVIEW OF THE COSYE ENVIRONMENT

5.1 COSYE RTI Server

COSYE RTI Server is a .NET implementation of the RTI described in the Federate Interface Specification. Currently, it implements approximately 50% of the services described in the standard. COSYE RTI Server runs as a Windows service on either a server or a workstation. It can be controlled via the Services applet found under Administrative Tools in the Windows Control Panel, or via the COSYE RTI Service Manager applet which resides as an icon in the notification area of the Windows Taskbar.

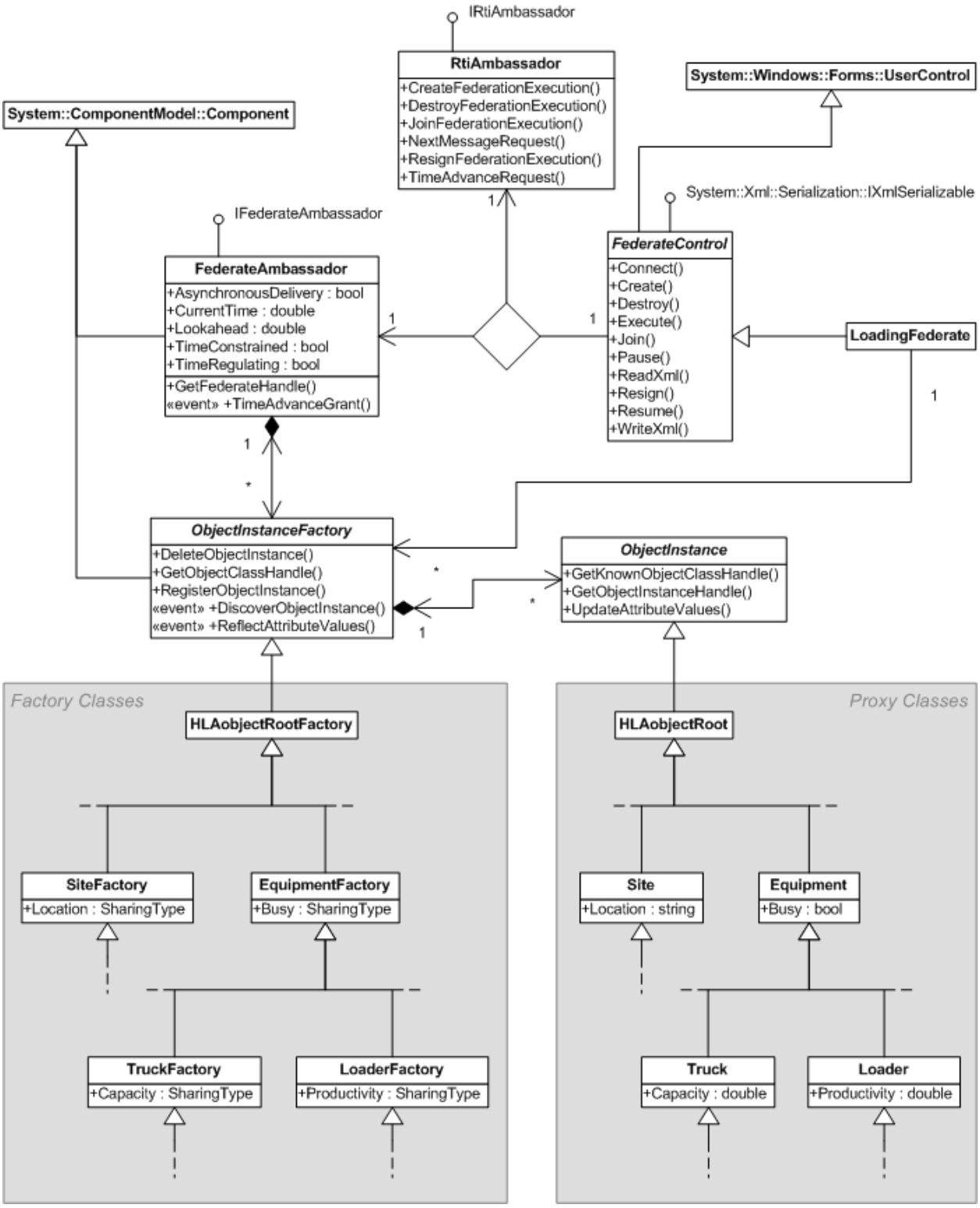

Figure 2: Overview of COSYE framework

5.2 COSYE Federate Host

COSYE Federate Host is an application that hosts federates developed using the COSYE Framework. It can host multiple federates at once and makes the process of starting a federation execution much simpler. The federate host can also save and load its configuration to and from an XML document making the setup of a complex federation easy.

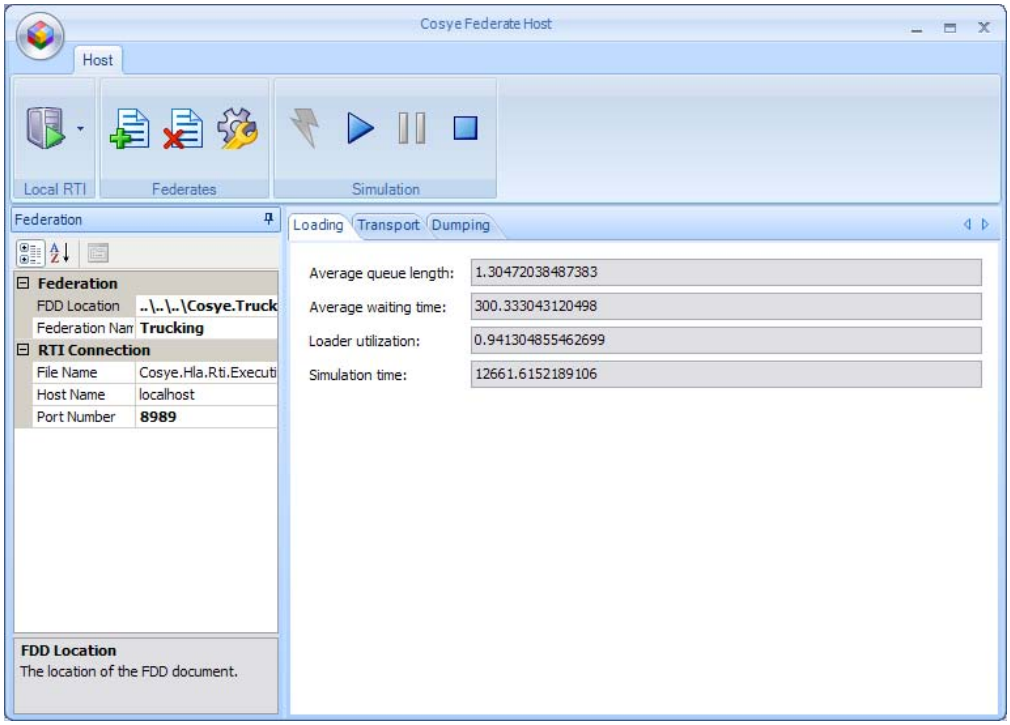

Figure 3: COSYE federate host

5.3 COSYE OMT Editor

COSYE OMT Editor is a plug-in for Microsoft's Visual Studio development environment. The OMT Editor serves two purposes: First, it allows federate developers to visually edit OMT documents within Visual Studio rather than editing them as XML files. Secondly, it generates from an OMT document the C# or Visual Basic source code for the object instance factories, object instance proxies, and interaction class terminals that the COSYE Framework uses to represent the object classes and interaction classes defined in the document.

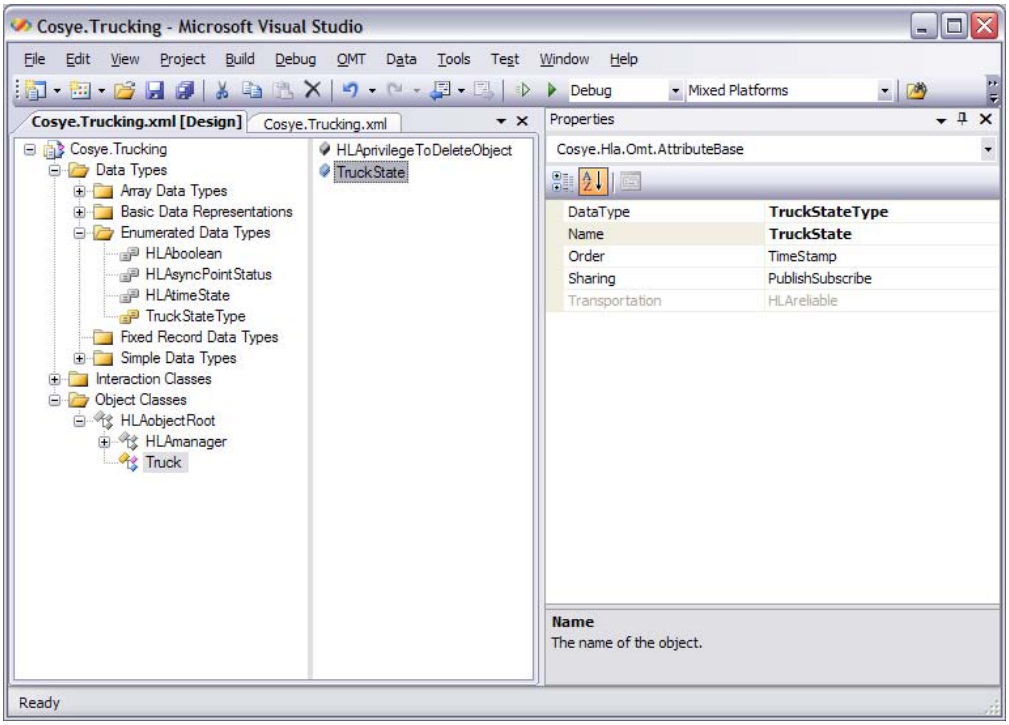

Figure 4: COSYE OMT Editor

5.4 COSYE Framework

The COSYE Framework is an application programming interface (API) intended to make development of COSYE Federates a rapid and simple process. The design philosophy of the API is to take advantage, as much as possible, of the visual nature of Microsoft's Visual Studio development environment. Thus:

- Federates are represented as user controls that can be hosted within COSYE Federate Host;
- Object instance factories and interaction class terminals (components representing object classes and interaction classes respectively) can be added to federates and their properties edited to specify publication and subscription requirements;
- RTI-initiated services are represented as events of the components residing on the federate which the federate developer can tie to code.

5.5 Simphony Core Services

Simphony Core Services is a class library distributed with the Simphony.NET simulation system. It provides the basic services required by any simulation environment. Developers of COSYE federates typically utilize the following:

- A discrete event simulation engine;
- Resources and waiting files to be used with the discrete event simulation engine;
- Sampling and fitting of probability distributions; and
- Statistics collection and summarization.

5.6 COSYE Test Federate

The COSYE Test Federate (see Figure 6) is intended to be used as a testing tool and as a learning tool. It provides a user interface from which a user can interact with COSYE RTI Server. Federate-initiated services can be invoked through menu items, and both federate and RTI-initiated services are logged to the main window.

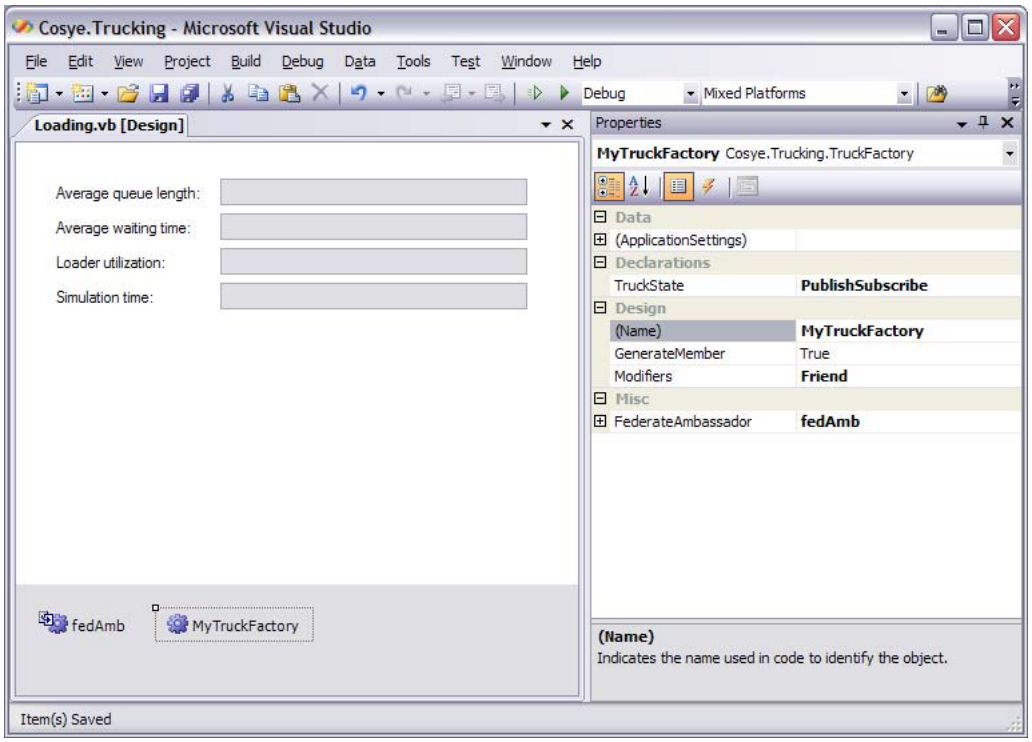

Figure 5: COSYE federate form

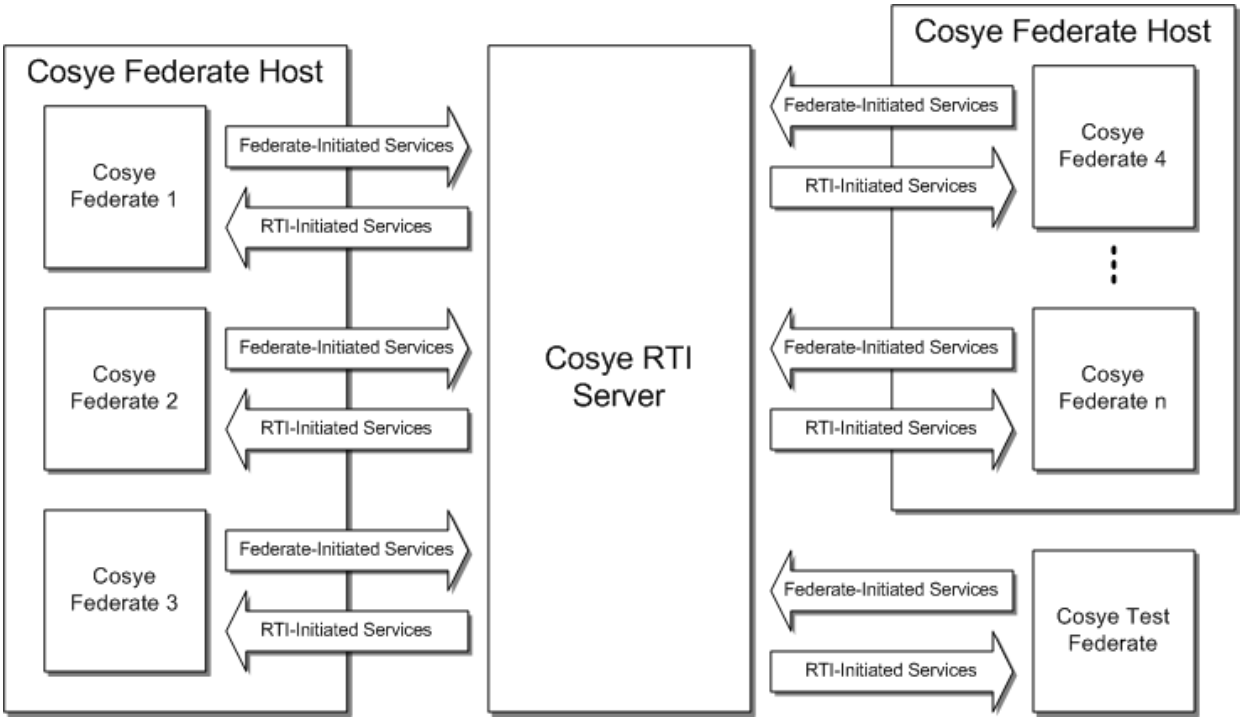

Figure 6: COSYE federation

6 USING COSYE OMT EDITOR

Federate developers use COSYE OMT Editor to develop OMT documents visually within the Microsoft Visual Studio environment. Developers can switch between the designer (visual) view of the document and the code (XML) view as they work. Changes made to one view are reflected in the other. As development proceeds, C# or Visual Basic code is generated to represent the objects and interactions defined in the document.

6.1 Developing an FOM or SOM Document

Federate developers begin by adding an FOM or SOM document to a C# or Visual Basic project. The new document will automatically contain all of the predefined object classes, interaction class, and data types as specified in the Object Model Template Specification. The OMT Editor represents the OMT document in a hierarchical fashion using a tree-type interface. Developers can add items to the tree from the OMT menu in the Visual Studio menu bar and support is provided for the cut, copy, paste, and delete operations on the Edit menu. The properties of the items in the tree can be edited from Visual Studio's Properties Window.

6.2 Generating C# or Visual Basic Code

Generation of the C# or Visual Basic code representing the object instance factories, object instance proxies, and interaction class terminals is seamless. No particular action needs to be taken on the part of the federate developer. Visual Studio automatically updates the code whenever necessary.

7 CLASSES IN THE COSYE FRAMEWORK

7.1 FederateControl

The abstract base class for federates, FederateControl derives from the .NET Framework class System.Windows.Forms.UserControl. It contains an RtiAmbassador and a FederateAmbassador through which the federate communicates with the RTI, together with various methods that allow it to be hosted within the COSYE Federate Host application. To create a federate, a federate developer adds a new user control that derives from FederateControl to a class library project. The developer then adds to this control the object instance factories and interaction class terminals that represent the object classes and interaction classes of interest to the federate and edits their properties to specify the federate's publication and subscription requirements. Finally, the developer writes code to handle the events raised by the FederateAmbassador, the object instance factories, and the interaction class terminals.

7.2 RtiAmbassador

The first of the two interfaces between the RTI and the federate, RtiAmbassador provides public methods corresponding to the federate-initiated services described in the Federate Interface Specification. For example, it contains a method called UpdateAttributeValues that corresponds to federate-initiated service 6.6, *Update Attribute Values*, and a method called NextMessageRequest that corresponds to 8.10, *Next Message Resquest*. While RtiAmbassador provides a method corresponding to every federate-initiated service supported by COSYE RTI Server, in practise federate developers only invoke methods that fall under the *Federation*, *Declaration*, and *Time Management* categories. For services in the *Object* and *Ownership Management* categories, federate developers normally invoke the corresponding method of the appropriate object instance factory, object instance proxy, or interaction class terminal, which will in turn invoke the method on RtiAmbassador.

7.3 FederateAmbassador

The second of the two interfaces between the RTI and the federate, FederateAmbassador provides public events corresponding to the RTI-initiated services described in the Federate Interface Specification. For example, it contains an event called ReflectAttributeValues that corresponds to RTI-initiated service 6.7, *Reflect Attribute Values†*, and an event called TimeAdvanceGrant that corresponds to 8.13, *Time Advance Grant†*. While FederateAmbassador provides an event corresponding to every RTI-initiated service supported by COSYE RTI Server, in practise federate developers only subscribe to events that fall under the *Federation*, *Declaration*, and *Time Management* categories. For services in the *Object* and *Ownership Management*

categories, federate developers normally subscribe to the corresponding event of the appropriate object instance factory or interaction class terminal. FederateAmbassador raises these events prior to raising its own event. When responding to service invocations from the RTI, FederateAmbassador ensures that the corresponding event is raised on the main thread of the federate application, thus alleviating the federate developer of the need to deal with multi-threading issues. Because FederateAmbassador derives from the .NET Framework class System.ComponentModel.Component, the various events it supports can be subscribed to visually within Visual Studio's Form Designer.

7.4 ObjectInstanceFactory

The abstract base class for object instance factories, ObjectInstanceFactory provides methods and events corresponding to the federate and RTI-initiated services in the *Object* and *Ownership Management* categories that apply to object classes and object instances. Only federate-initiated services that do not act on an existing object instance are provided however, the remainder are provided by the ObjectInstance class. For example, 6.4, *Register Object Instance* is provided by ObjectInstanceFactory, while 6.6, *Update Attribute Values* is provided by ObjectInstance. In addition, ObjectInstanceFactory implements the Factory Method design pattern which is a standard method in object oriented design of deferring instantiation of objects to subclasses. It will create object instance proxies in response to invocation of the RegisterObjectInstance method by a federate developer, or in response to an invocation of DiscoverObjectInstance by the RTI. In either case, the factory will add the proxy object to a dictionary it maintains that is indexed by object instance handle. Thus, given an object instance handle, a federate developer can query an object instance factory for the corresponding proxy object. In a sense, this dictionary represents the known object instances of the class the factory represents. Concrete subclasses of ObjectInstanceFactory contain properties that correspond to the class attributes of the object class the factory represents. These properties are all of type SharingType and federate developers use them to specify the publication and subscription requirements of their federates. Because ObjectInstanceFactory derives from the .NET Framework class System.ComponentModel.Component, concrete subclasses can be interacted with in a visual manner within Visual Studio's Form Designer when placed on a user control derived from FederateControl.

7.5 ObjectInstance

The abstract base class for object instance proxies, ObjectInstance provides methods corresponding to the federate-initiated services in the *Object* and *Ownership Management* categories that apply to existing object instances. Concrete subclasses of ObjectInstance contain properties that correspond to the instance attributes of the object instance the proxy object represents. These properties are all of the type specified in the federate's SOM document and federate developers use them to read and write the values of the instance attributes.

7.6 InteractionClassTerminal

The abstract base class for interaction class terminals, InteractionClassTerminal provides methods and events corresponding to the federate and RTI-initiated services in the *Object* and *Ownership Management* categories that apply to interaction classes. Because InteractionClassTerminal derives from the .NET Framework class System.ComponentModel.Component, concrete subclasses can be interacted with in a visual manner within Visual Studio's Form Designer when placed on a user control derived from FederateControl.

8 POTENTIAL DEPLOYMENT

Demonstration projects are actual implementations of the three CSE within the industrial partner's setting. We already have commitments from three leading companies in the respective areas to work with them towards the deployment of CSEs within the next five to seven years. The challenge is to receive commitment from other significant players such as the design firms, the supply firms and the support contractors in order to fully demonstrate the integrated approach. While setting up such an experiment is challenging, we are confident that given our strength within the construction community we can bring together a consortium to participate in the proof of concept in due time. Furthermore, we have the potential of working with FIATECH on demonstration projects to prove the validity of their roadmap (FIATECH 2004), which closely resembles our own vision.

REFERENCES

AbouRizk, S.M., and K. Mather. (2003) Simplifying Simulation Modeling through Integration with 3D CAD. *J. Constr. Engrg. and Mgmt.* 126(6): 475-483.

Chua, D.K.H., and G.M. Li (2002) RISim: Resource-Interacted Simulation Modeling in Construction. *J. Constr. Engrg. and Mgmt.* 128(3): 195-202.

Chua, D.K.H., D.Z. Li, and W.T. Chan. (2001). Case-Based Reasoning Approach in Bid Decision Making. *J. Constr. Engrg. and Mgmt.* 127(1): 35-45.

FIATECH (2004). Capital Projects Technology Roadmap. *Element 1: Tactical Plan, Scenario-based Project Planning.* Austin, TX 78759 <www.fiatech.org>

Hajjar, D., S. and AbouRizk (2002). Unified Modeling Methodology for Construction Simulation. *J. Constr. Engrg. and Mgmt*. 128(2): 174-185.

Halpin, D. W. (1976). CYCLONE: Method for Modeling of Job Site Processes. *Journal of the Construction Division*. 103(3): 489-499.

IEEE (2000) *Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)-Federate Interface Specification* 1516.1-2000

Jaafari, A., K.K. Manivong, and M. Chaaya. (2001) VIRCON: Interactive System for Teaching Construction Management. *J. Constr. Engrg. and Mgmt.* 127(1): 66-75.

Kamat, V.R. and J.C. Martinez. (2003). Validating Complex Construction Simulation Models Using 3D Visualization. *Systems Analysis Modelling Simulation*. 43(4): 455-6.

Lu, M. (2003) Simplified Discrete-Event Simulation Approach for Construction Simulation. *J. Constr. Engrg. and Mgmt.* 129(5): 537-546.

Lu, M., M. Anson, S.L. Tang, and Y.C. Ying. (2003) HKCONSIM: A Practical Simulation Solution to Planning Concrete Plant Operations in Hong Kong. *J. Constr. Engrg. and Mgmt.* 129(5): 547-554.

Martinez, J.C. and P.G. Ioannou (1999). General-Purpose Systems for Effective Construction Simulation, *J. Constr. Engrg. and Mgmt.* 125(4): 265-275.

Marzouk, M. and O. Moselhi. (2003) Object-oriented Simulation Model for Earthmoving Operations. *J. Constr. Engrg. and Mgmt*. 129(2): 173-181.

Oloufa, A.A. (1993). Modeling Operational Activities in Object-Oriented Simulation. *Journal of Computing in Civil Engineering*. 7(1): 94-106.

Peña-Mora, F., S. Han, S. Lee, and M. Park. (2008) Strategic-Operational Construction Management: Hybrid System Dynamics and Discrete Event Approach. *J. Constr. Engrg. and Mgmt.* 134(9): 701-710.

Rojas, E.M., and A. Mukherjee. (2005) General-Purpose Situational Simulation Environment for Construction Education. *J. Constr. Engrg. and Mgmt.* 131(3): 319-329.

Shewchuk, J. and T. Chang. (1991). An approach to object-oriented discrete-event simulation of manufacturing systems. *23rd Winter Simulation Conference*, Phoenix, Ariz., 302-311.

Shi, J. (1999) Activity-Based Construction (ABC) Modeling and Simulation Method. *J. Constr. Engrg. and Mgmt.* 125(5): 354-360.

Zhang, H., H. Li, and M. Lu. (2008) Modeling Time-Constraints in Construction Operations through Simulation. *J. Constr. Engrg. and Mgmt.* 134(7): 545-554.

AUTHOR BIOGRAPHIES

SIMAAN M. ABOURIZK, PhD, PEng, is the NSERC Industrial Research Chair in Construction Engineering and Management and the Canada Research Chair in Operation Simulation at the University of Alberta. He is renowned in the academic construction community for his research in computer simulation and its applications in construction planning, productivity improvement, constructability reviews and risk analysis and was recently recognized with the ASCE Peurifoy Award. He can be contacted by email at <abourizk@ualberta.ca>.

STEPHEN HAGUE is primarily involved in implementing modeling and simulation structures developed by the research team at the NSERC Industrial Research Chair in Construction Engineering and Management at the University of Alberta. He also develops software technologies for a range of construction project applications. He holds a Bachelor of Science in Honours Mathematics. He can be contacted by email at <hague@construction.ualberta.ca>.