THE ARCHITECTURE OF S2

Ardeshir Mahdavi
Mustafa Emre Ilal, Paul Mathew, Robert Ries, Georg Suter, Rohini Brahme

School of Architecture,
Carnegie Mellon University
Pittsburgh, Pennsylvania, 15213, U.S.A.

ABSTRACT

We present in this paper the essential aspects of the S2 system. This is the internet realization of SEMPER: An active, multi-domain, space-based, object oriented design environment for integrated building performance modeling. We begin with an overview of SEMPER principles. We then present the component based system architecture of S2, which makes use of the Common Object Request Broker Architecture (CORBA) for communication between components including our analysis applications (written in C++) and other S2 components (written in Java). We describe the underlying approach to and implementation of the shared object model. This serves as the shared building representation from which the analysis modules derive their domain object models using a homology-based mapping process.

INTRODUCTION

The key features of the S2 environment are as follows: A user can access the system regardless of the computer hardware, operating system or the location on a network; geographically distributed users can asynchronously generate a building model through the user interface; this building model can then be simultaneously accessed by multiple simulation applications running on remote simulation servers; persistent storage is provided for project data and evaluation results; designers using the system have access to multiple libraries that contain building information such as material data, construction types, schedules, and weather data.

S2 is a reengineered version of SEMPER for the internet to support collaborative design scenarios. SEMPER is a multi-aspect prototype design environment that incorporates an object-oriented, space-based building model, with *dynamic* links to different building performance evaluation applications (Mahdavi 1996). It is thereby intended to computationally support the evaluation of buildings across *multiple* performance mandates concurrently, with a view toward achieving total building performance and systems integration.

SEMPER incorporates the unique synthesis of four principles:

1. *A methodologically consistent performance modeling approach through the entire building design process*
   The performance simulation modules incorporated within SEMPER rely on a consistent (first-principles based) modeling of the physical processes relevant to the building's performance. SEMPER currently includes modules for energy, air flow, HVAC, thermal comfort, lighting, acoustics, and life-cycle analysis.

2. *Seamless and dynamic communication between the simulation model and the general building representation*
   This provides "on-line" building performance feed-back to the user while eliminating the need for explicit definition and updating of the underlying simulation model. SEMPER exemplifies this by dynamically linking the building's general representation (essentially a shared object model) with structurally homologous representations of the simulation modules (essentially the domain object models) (Mahdavi and Wong 1998, Mahdavi et al. 1997). Figure 1 illustrates, for example, the homology-based mapping from the shared building representation to the thermal and air flow simulation domains. The shared building representation embodies topological information on the architectural spaces and their relationships. The energy and air flow simulation modules utilize a spatial representation consisting of spatial units (cells) with nodes that define finite control volumes. These nodal representations of the building are configurationally homologous to and are directly derived from the space-based representation of the building in the shared object model. This homology-based mapping mechanism allows for rapid feedback and is therefore a powerful design instrument, since evolving building designs can be made subject to comprehensive parametric studies in multiple domains without

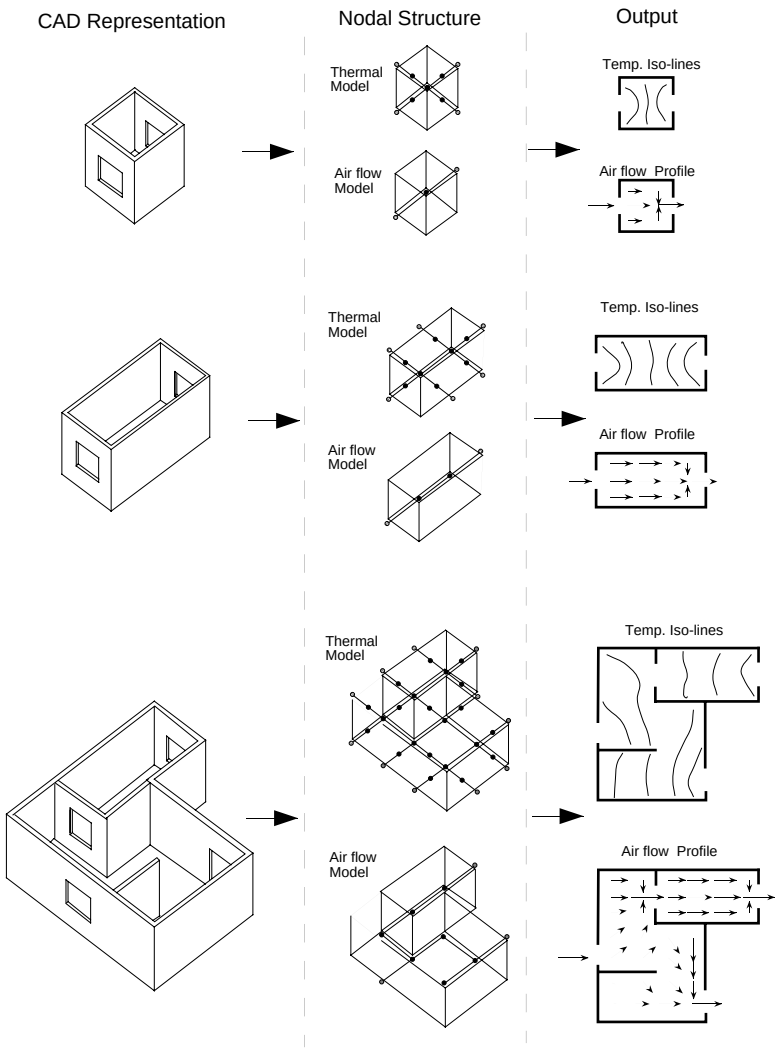

FIGURE 1. Homology-based mapping of the space-based architectural representation (left) to the nodal building representations in SEMPER's thermal and air flow simulation modules (middle) with their corresponding simulation results (right)

having to input the building model separately for each domain.

3. *Comprehensive performance modeling*
   In order to analyze the implications of design decisions within and across multiple performance domains, SEMPER incorporates modules to deal with thermal quality, indoor air quality, visual quality, acoustic quality as well as environmental impact (Mahdavi et al. 1996).

4. *Active design support using a "preference-based" performance-to-design mapping technology*
   Active design support enables a designer to interactively and dynamically explore the relationships between design decisions and performance variables (Mahdavi 1993). The active design support in SEMPER would provide two functionalities: *a)* the user makes a change in a performance variable in order to see the corresponding changes in building design variables; and *b)* the user makes a change in a building design variable and observes resulting changes in other building design variables when one or more relevant performance variables are constrained.

The proof-of-concept SEMPER prototype 1 was implemented on UNIX with a Tk-TCL user interface, which uses AutoCAD for geometry input and display. This prototype incorporates two simulation modules for energy and thermal comfort analysis.

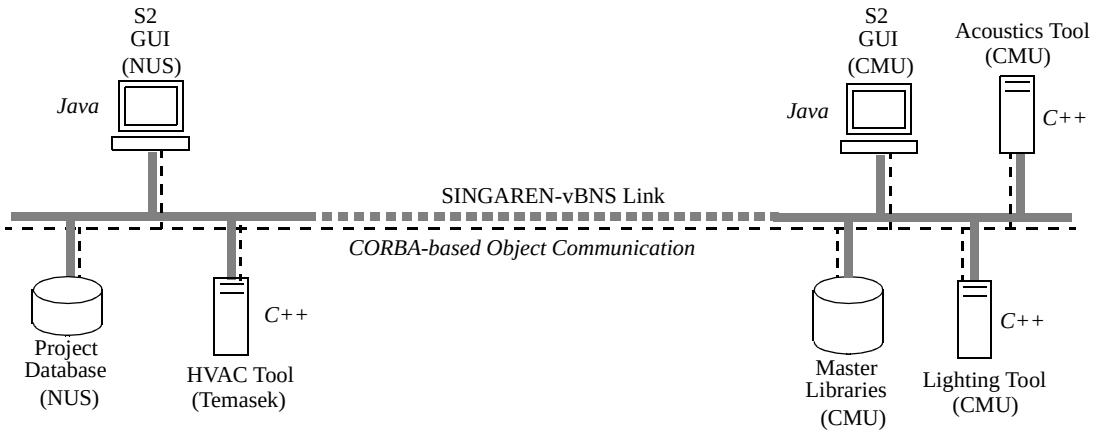

FIGURE 2. Schematic illustration of components of S2 in the case of two distributed users in Singapore and the U.S. collaborating on a building design/engineering project

S2: SEMPER ON THE INTERNET

Building on the proof-of-concept prototype 1, SEMPER Prototype 2 (S2) seeks to realize SEMPER on the internet, toward supporting collaborative performance-based building design between geographically distributed users. The various components of S2 (graphical user interface, S2-Kernel, simulation modules, etc.) are distributed over a network, using Common Object Request Broker Architecture (CORBA) as their communication framework. A preliminary implementation of S2 on the internet will involve users at Carnegie Mellon, and two institutions in Singapore - National University of Singapore and Temasek Polytechnic (figure 2).

S2 SYSTEM ARCHITECTURE

The distributed system architecture of S2 is made up of the following components:

- *Graphical User Interface (GUI)*:
  The GUI allows users to create and edit building descriptions. It is implemented in Java and includes a simple geometry editor. It can be seen as the mediator between the user and the S2-Kernel. Whenever a user wants to modify the existing design, the GUI forwards the request to the S2-Kernel, which conducts the requested operation. The result is sent back to the GUI, which displays it.

- *Gateway*:
  The Gateway is the entry point for any GUI. It holds administrative functions and coordinates component interactions. It is a connection manager whose uniform resource locator (URL) is published and known by all users.

- *S2-Kernel*:
  The S2-Kernel is the functional core of the system and is implemented in Java. It populates the Shared Object Model (SOM) and maintains its consistency. It provides access to the SOM which is stored persistently in the database and also provides all necessary editing mechanisms for it. SOM represents the design project and reflects the building representational requirements of seven simulation modules.

- *Geometric Modeling Engine*:
  The Geometric Modeling Engine is a common utility purely for geometric processing. It is a library of functions available for any other component.

- *Database*:
  The Database provides persistent storage for project information (SOM) as well as shared resources (libraries).

- *Simulation modules*:
  These are applications that can derive their own building representations i.e domain object models (DOM) from the SOM for domain specific processing. The first release of S2 will include modules for thermal analysis, HVAC system simulation, thermal comfort analysis, lighting, and environmental impact analysis.

The system architecture of S2 with its major components and their underlying communication mechanisms is shown in figure 3.

CORBA was chosen as the communication framework between components. CORBA provides a connection mechanism that is independent of implementation language. This mechanism enables objects implemented in different languages to

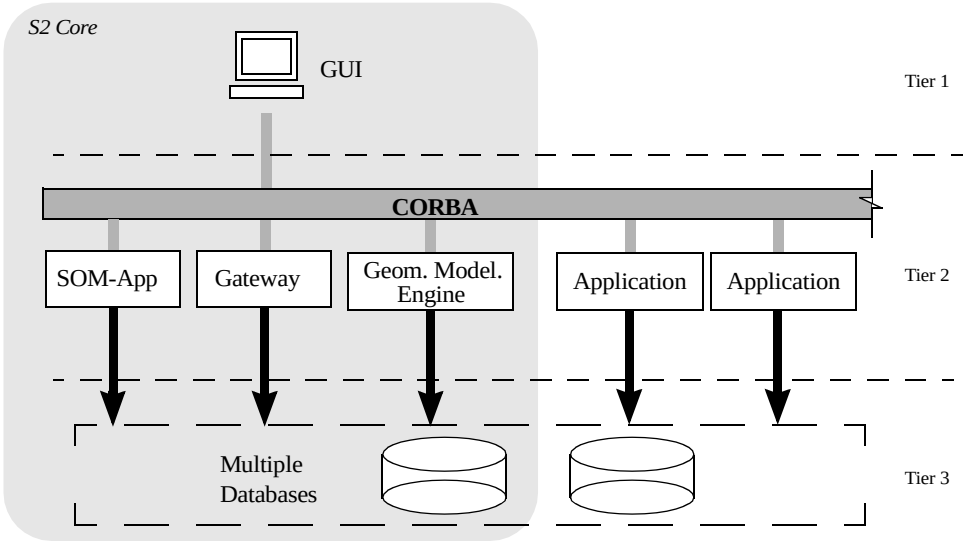

FIGURE 3. S2 system architecture - components and communication

communicate without the need for any language mapping code. This is an important feature considering S2's implementation which is partially in C++ (Simulation Modules) and partially in Java (S2-Kernel, GUI). CORBA development also supports a modular structure that simplifies the addition of future components to the system, which is essential for S2.

In S2, the role of each component and their overall interaction scheme is summarized below:

On start-up, a GUI first contacts the Gateway for authentication. Once the user is authenticated, the Gateway releases an Attendant and assigns it to the GUI. The Attendant acts as a higher level liaison for the GUI. Attendants function as process coordinators and file managers.

For example, during the opening of a project (Open Project use case), when the GUI requests the available databases, the Attendant dynamically creates a list of databases which the user has access to and returns the list to the GUI. Similarly, when a user wants to browse the projects inside a selected database, the Attendant conducts its queries and creates the appropriate list. When the user wants to start working with a given project, the GUI again sends this request to its Attendant, which in turn, selects an available server, spawns an S2-Kernel process (which initializes the database connection for the selected project) on it, and connects the GUI with this S2-Kernel. Figure 4 shows the interaction diagram.

During the editing of the project (SOM), the GUI contacts S2-Kernel directly. However, for the execution of simulations, the GUI turns again to its Attendant which finds (dynamically, through CORBA services) the appropriate, available simulation modules and initiates the simulation processes. At the end of such a process, the results are directly sent to the database (attached to the SOM) and the GUI-Attendant pair is notified of this event through an event service.

SHARED OBJECT MODEL

As stated earlier, one of the key features of SEMPER is that a user can input a building description and run simulations in multiple domains without having to manually create domain-specific building representations for each simulation tool. S2 facilitates this through the use of a Shared Object Model (SOM), from which the domain object models (DOM) for each simulation module can be automatically derived without any additional user intervention. The SOM was developed through an iterative bottom-up approach. As such, it is not intended to be a general building model for a broad array of building applications. Rather, it has been specifically developed to reflect the building representational requirements of S2's seven simulation modules. The SOM can, of course, evolve to support additional applications, and additionally, can inform the integration of representations that support simulation into more general models.

The following key features of the SOM bear mentioning:

- The SOM uses a space-based representation i.e. the building is composed of non-intersecting spaces aggregated into sections. This feature was deemed necessary in order to generate the domain representations needed for detailed simulations.

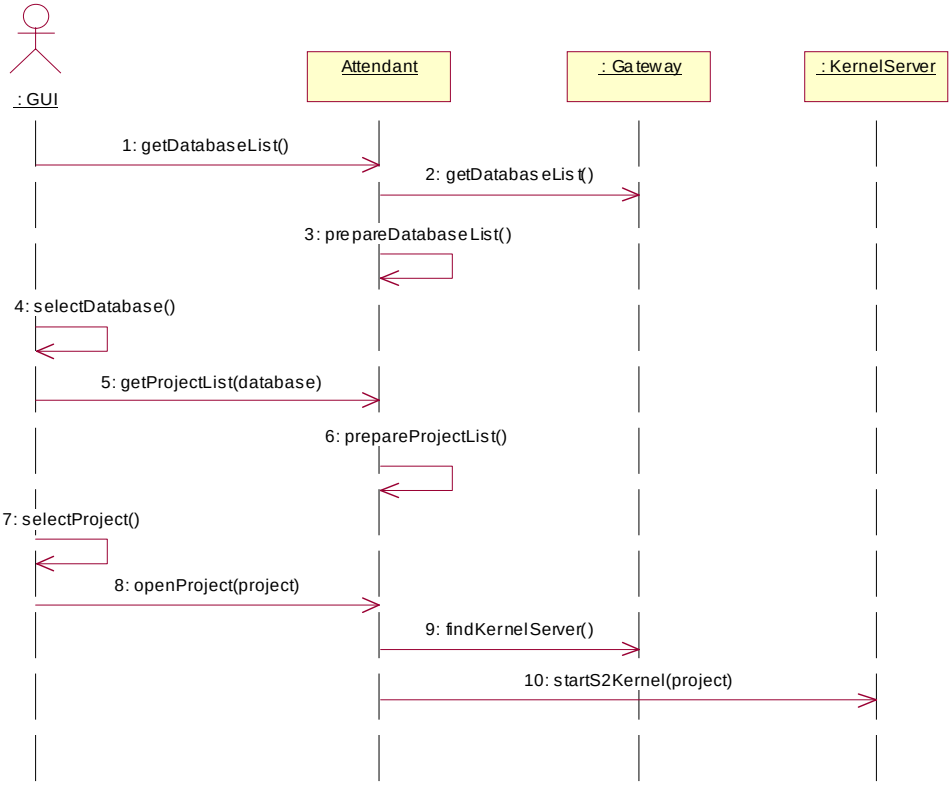

FIGURE 4. The interaction diagram for the Open Project use case.

- The primary elements of the building representation each have a geometry (described via an associated geometry class), and a set of non-geometric attributes (described via a "type" class, referenced by type name). For example, a "SOMWall" object has an associated "SOMPolygon" geometry object and a "SOTConstruction" type object that contains layer and material specifications. The type classes, which essentially represent specification objects, are organized into libraries (e.g. wall construction types, space occupancy types, window shade types, etc.).

- The SOM has a number of "visitor" classes (Gamma et al. 1994) that encapsulate the knowledge for an operation that requires coordination among multiple objects in the SOM. For example, adding a space to the SOM involves a complex series of queries and operations on many objects, requiring a complex traversal of the SOM hierarchy. Visitor classes in the SOM extend from a common superclass so that operation specific visitor subclasses can be added without change to the primary SOM elements.

Table 1 describes the package structure for the SOM. Figure 5 shows the SOM class hierarchy for the primary physical elements.

TABLE 1: Package structure for the Shared Object Model

Package	Description
Semper	Root package - does not contain any classes
Semper.som	Classes that describe the primary elements of the building model - spaces, walls, etc.
Semper.som.visitor	Visitor classes that encapsulate the knowledge for operations that require coordination among multiple objects in the SOM
Semper.geometry	Classes that describe geometric entities used in the SOM (e.g. points, lines, polyhedra, etc.)
Semper.sot	"Type" classes that encapsulate specification information for the primary physical elements of the building model
Semper.util	Utility classes

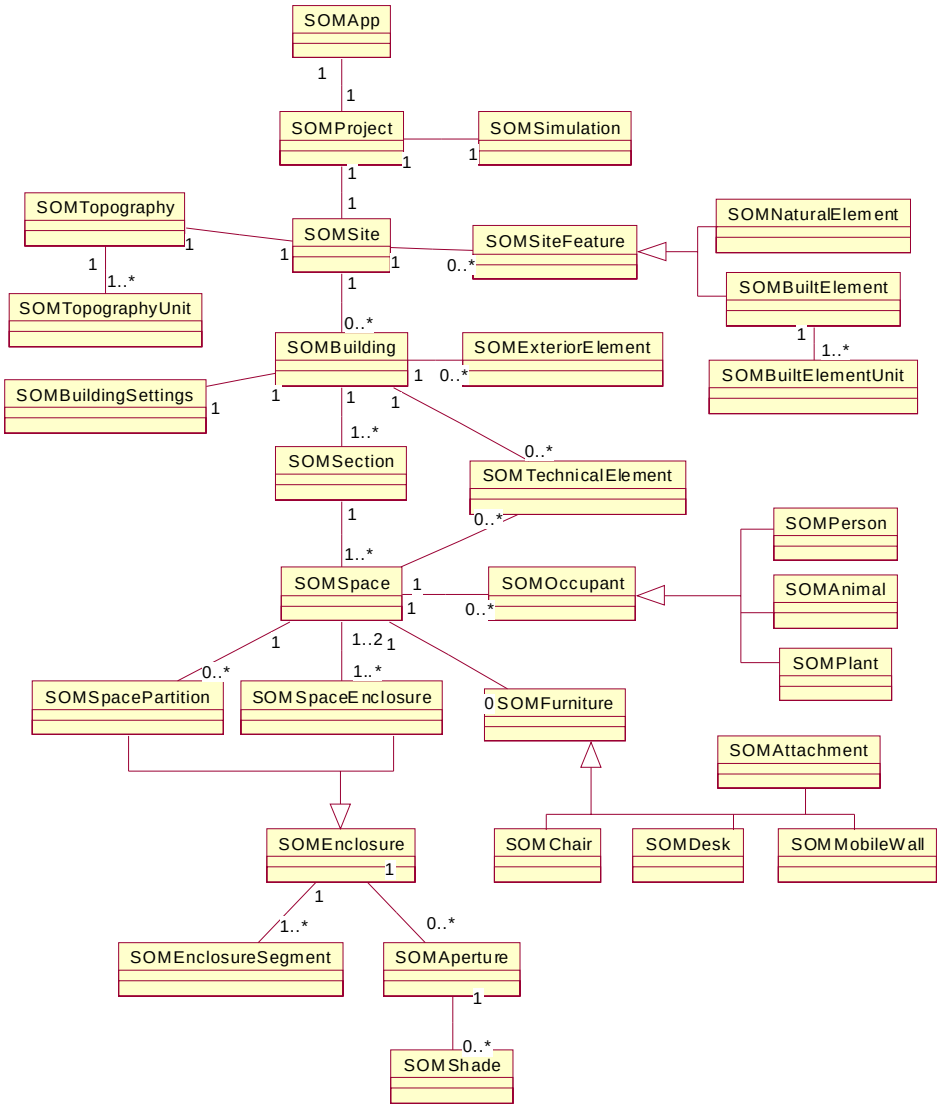

FIGURE 5. The class hierarchy for primary elements of the SOM

DOMAIN OBJECT MODELS

It has been noted earlier that the building representations for specific simulation modules i.e. the domain object model (DOM), is derived from the Shared Object Model (SOM) without any user intervention. However, with regard to SOM-DOM mapping and information flow, an important distinction needs to be made between: *a)* "analysis-only" domains, which do not involve the detailed design of physical components (e.g. thermal analysis, air flow modeling, environmental impact analysis); and *b)* "analysis+design" domains, which involve the detailed design of physical components (e.g. HVAC). The following discussion explains this distinction further, using as examples SEMPER's thermal analysis module NODEM ("analysis-only"), and its HVAC module ("analysis+design").

SOM-DOM Mapping in NODEM

NODEM uses homology-based mapping to automatically derive its DOM from the SOM. As such, the SOM contains all the information necessary for the derivation of the DOM. The information in NODEM's DOM is simply a restructured subset of the information in the SOM. Figure 6 schematically illustrates the mapping between the primary objects in SEMPER's shared object model and NODEM's domain object model (Mathew and Mahdavi 1998). Two types of mapping operations are used generate the domain object model:

- Object-Object mapping: An object in the shared model has a corresponding object in the domain model (e.g. the "Space" object in SEMPER has a corresponding "Space" object in NODEM). However, these objects do not necessarily have an identical set of attributes.

- Object-Attribute mapping: An object in the shared model is transformed into attributes of an object in the domain model (e.g. the "Occupant" objects in SEMPER are transformed into attributes of the "Space" object in NODEM).

SOM-DOM Mapping in HVAC

Domains that involve detailed design of physical components, such as HVAC, present a complex set of issues vis-à-vis the relationship between the SOM and the DOM. Indeed, the role of the HVAC analysis within the overall SEMPER environment raises two important questions in terms of the shared object model, domain object model, and GUI:

- *To what extent should the object class hierarchy of the primary shared building model reflect the requirements of various detailed applications?*
  Building HVAC energy analysis can be performed at various levels of depth and resolution. From the software engineering point of view, this represents a number of problems. A building model that is too restricted, may allow only for a limited and ultimately less useful set of analysis options. On the other hand, a model that would capture all the requirements of disciplinary analysis may become too large, leading to the classic problems of massive product models.
- *How will users with different interest and expertise communicate with a multi-aspect software systems that should allow for different level of analysis?*
  Certain levels of building energy analysis are more relevant to the types of questions that need to be asked and answered by primary building designers (particularly architects). For example, decisions pertaining to the effects of enclosure and glazing, massing, orientation, and natural ventilation, should be covered by the core analysis capabilities of a building design tool. However, analysis of detailed building support systems and controls may be more relevant to the activities of the domain specialist (e.g. HVAC designer).

Our recent research findings have lead us to the following position in the case of the HVAC module's relationship to the overall S2 environment:

- *Distinction between a general user (interacting with the S2 SOM) and the specialist (interacting with DOM)*
  A distinction needs to be made as to the level of analysis appropriate for the general system user. SOM must include all necessary classes for such analysis. However, the type of HVAC analysis that can be initiated at the S2 level by the general user would be limited to default system

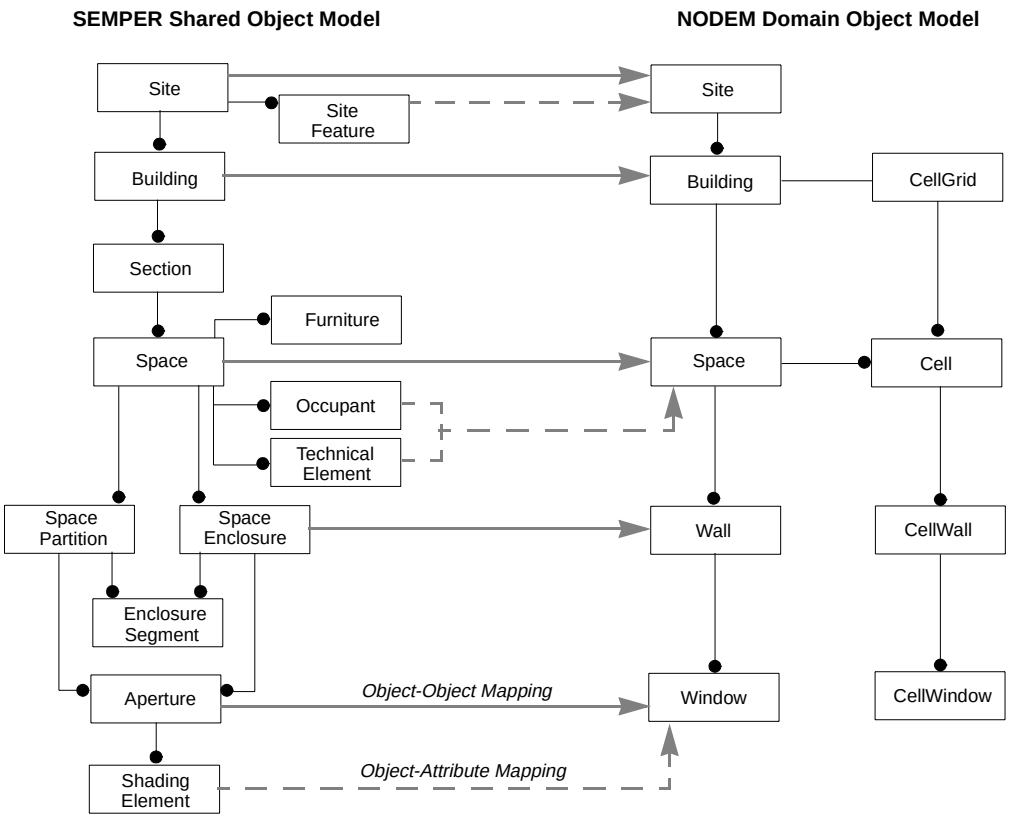

FIGURE 6. Schematic illustration of mapping between primary objects in SEMPER's shared object model and NODEM's domain object model

types. The user selects the spatial domain to be supplied with HVAC, the type of the system, and optionally the thermal zone boundaries. This information is (apart from the simulation results) the only HVAC-relevant specification included in SOM. It is not appropriate to include the detailed-design classes of all applications in a single SOM. The resulting size and complexity is likely to hamper modularity, make interacting with legacy applications difficult, reduce ease of system modification, and dilute structural transparency of the system architecture.

- *HVAC module can derive "default" DOM based on SOM.*
  Beyond SOM's objects, DOM will include domain-specific objects such as generators (e.g. chillers and boilers), distribution components (e.g. ducts and fittings), and terminals (e.g. diffusers and radiators). The basic configuration of these elements is established by HVAC domain's expert system, based on the previously mentioned, limited, HVAC-relevant information in the SOM.

- *Domain specialist can "customize" the DOM*
  A domain specialist (or a general user interested in other-than-default HVAC analysis) can directly interact with DOM objects thus circumventing the HVAC domain expert. This interaction does not occur via S2's general GUI, but rather via an independent user interface (or a plug-in to the general GUI). As a general rule, objects that are not defined and generated in SOM are not "visible" to it and cannot be manipulated by it. Objects and system designs defined by the specialist in DOM are not transparent to SOM, nor are they saved as part of the SOM. They must be saved as part of the DOM and maintained by the specialist. However, simulation results due to such designs can be properly tagged and sent back to SOM as simulation results.

- *Concurrent viewing of DOM and SOM*
  While various DOMs will not be integrated within a single unified SOM, applications are conceivable that would allow for concurrent viewing of all object information (e.g. their geometric properties), regardless if they are SOM or DOM objects. This integrated display utility would allow for detection of system integration problems (i.e. conflicts between elements of various domains). Active manipulation of domain elements, however, can be only realized via each domain's user interaction mechanisms.

CONCLUDING REMARK

The S2 system has been designed, pilot-tested, and as is currently under implementation. S2 is designed to support collaborative performance-based building design between geographically distributed users, over the internet. This is done by modularizing and distributing the components of S2, using CORBA as a communication framework. The S2 system will be further tested and refined as part of a collaborative project between Carnegie Mellon University and two academic institutions in Singapore.

ACKNOWLEDGMENTS

The authors would like to thank the S2 development team which, besides the authors, also includes: Seongju Chang, Beran Gurtekin, Youngsoo Kwon, Prechaya Mahattanatawe, Zhengchun Mo, Vineeta Pal and Nyuk-Hien Wong.

REFERENCES

Gamma, E., R. Helm, R. Johnson, and J. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software (Addison Wesley), 1994.

Mahdavi, A., "Computational Support for Performance-based Reasoning in Building Design", Proceedings of the CIB-ASTM-ISO-RILEM International Symposium "Applications of the Performance Concept in Building", Tel Aviv, Israel. Vol. 1, pp. 4.23 - 4.32, 1996.

Mahdavi, A., "'Open' Simulation Environments: A 'Preference-Based' Approach", Proceedings of CAAD Futures '93, CMU, Pittsburgh, Pennsylvania, USA. pp. 195 - 214, 1993.

Mahdavi, A. and N. H.Wong, "From Building Design Representations to Simulation Domain Representations: An Automated Mapping Solution for Complex Geometries", Computing in Civil Engineering; Proceedings of the International Computing Congress, 1998 ASCE Annual Convention, pp. 1-10, 1998.

Mahdavi, A., P. Mathew, and N.H. Wong, "A Homology-based Mapping Approach to Concurrent Multidomain Performance Evaluation", Proceedings of the The Second Conference on Computer Aided Architectural Design Research in Asia: CAADRIA '97. Hsinchu, Taiwan. pp. 237 - 246, 1997.

Mahdavi, A., R. Brahme, S. Kumar, G.Liu, P. Mathew, R. Ries, and N.H. Wong, "On the Structure and Elements of SEMPER", Design Computation; Collaboration, Reasoning, Pedagogy: Proceedings of the 1996 ACADIA (The Association for Computer Aided Design in Architecture) conference. Tucson, Arizona, USA. pp. 71 - 84, 1996.

Mathew, P. and A. Mahdavi, "High-Resolution Thermal Modeling for Computational Building Design Assistance", Computing in Civil Engineering; Proceedings of the International Computing Congress, 1998 ASCE Annual Convention, pp. 522-533, 1998.