Building Simulation

Seventh International IBPSA Conference
Rio de Janeiro, Brazil
August 13-15, 2001

DISTRIBUTED WEB-BASED BUILDING PERFORMANCE COMPUTING: A SINGAPORE-US COLLABORATIVE EFFORT

Lam K. P.[1] , Mahdavi, A.[2,1] , Brahme, R.[2] , Kang, Z.[3] ,
Ilal, M. E.[2] , Wong, N. H.[1], Gupta, S.[1,2] , Au, K. S.[3]

[1]Dept. of Building, School of Design and Environment, National University of Singapore, Singapore 117566

[2]School of Architecture, Carnegie Mellon University, Pittsburgh, PA 15213, USA

[3]Department of Intelligent Building Technology, School of Eng, Temasek Polytechnic, Singapore 529757

ABSTRACT

This paper reports on the progress of the S2 project. This is an ongoing effort toward an Internet-based environment for distributed collaborative performance-based building design and evaluation. A user can access the S2 system regardless of hardware, operating system or location on a network. Geographically distributed users can generate and edit building models via a platform-independent user interface. These building models can then be made subject to concurrent analysis by multiple simulation applications running on remote servers. Persistent storage is provided for project data and evaluation results. Designers using the system have access to multiple libraries with semantic building information.

1 INTRODUCTION

We present an ongoing initiative to develop and realize a computational environment toward provision of multi-disciplinary building performance simulation services on the Internet. This initiative is based on technologies developed in an earlier prototype, i.e., the SEMPER system. SEMPER is an active, multi-domain, space-based, object oriented design tool for integrated building performance computing (Mahdavi 1999). Building on the proof-of-concept prototype 1, SEMPER Prototype 2 (S2) seeks to realize SEMPER on the Internet, toward supporting geographically distributed users in collaborative performance-based building design (Mahdavi et al. 1999). Currently, the S2 development is pursued within the framework of an international collaboration between three institutions, NUS (National University of Singapore), CMU (Carnegie Mellon University), and TP (Temasek Polytechnic, Singapore) (Mahdavi et al. 2000). In this paper, we describe the main concepts, components, and architecture of this technology. Furthermore, we provide some detailed information on recent advances in the area of thermal performance analysis within the S2 framework.

2 BUILDING REPRESENTATION

Building simulation applications typically require a large amount of information on the building, including information on the geometry, construction materials and properties, and context (e.g. weather conditions). When we speak of the building representation in S2, we mean the "Shared Object Model" (or "SOM"). This is a hierarchically structured template (a class hierarchy in object-oriented programming terms) to capture the essential elements of a building and their properties, to the extent required by the simulation applications in the S2 environment (Mahdavi et al. 1999). Note that the shared object model in S2 is not a "universal building model", i.e. it is not an application-independent quasi-platonic representation of a building as such. Rather, SOM has emerged via a bottom-up approach out of the informational requirements of a number of technical analysis applications adopted in S2 (Mahdavi 2000, 1998). We share neither the naïveté of those who claim universal building models are realistically attainable, nor do we share the skepticism of those denying any form of shared multi-disciplinary schemes for building information representation. In our experience, a shared building model can be arrived at for a number of technical analysis applications, and for performance inquiries of a certain range of informational resolution. S2's SOM illustrates such a possibility. Moreover, SOM in itself does not contain the entire building information. Rather, it contains a tightly structured "notation" of constitutive building elements, with pointers to (addresses for) the detailed information on such elements in the data repository for the persistent storage of such information.

3 DOMAIN REPRESENTATIONS

While SOM may allow retrieving the necessary building geometry, material, and context information that the S2 applications require, it is not sufficient on its own for a building performance simulation application to function. For each disciplinary domain, the simulation application's representation, or the

"Domain Object Model" (DOM), must be generated upon filtration and modification of information in SOM according to the specific view of the building in that domain. Moreover, domain specific entities (e.g. finite control volumes in numeric heat and mass transfer computation) may have to be added to what is inherited from SOM. While developing technologies for automated SOM-to-DOM mapping is non-trivial, the S2 project has shown that, for certain applications and inquiries within a certain range of informational resolution, such mapping is both possible and effective.

Currently the "thermal suite", an application for building energy modeling including simple HVAC and multi-zone airflow analysis (Mahdavi et al. 1997a, Mathew and Mahdavi 1998, Wong and Mahdavi 2000), as well as thermal comfort assessment (Kumar and Mahdavi 1999) is being integrated within the S2 system (cp. APPENDIX A for more details on the thermal suite in S2). Domain representations (DOMs) have been also developed for lighting simulation (Pal and Mahdavi 1999), environmental impact analysis (Mahdavi and Ries 1998), and room acoustics (Mahdavi et al. 1997b).

4 MAPPING

An integrated system for building performance evaluation that seeks to eliminate data input redundancy (to generate and modify the basic building model, and subsequently all the domain building models) cannot do without some form of representational mapping. Within the framework of the SEMPER project, we developed a functional "homology-based" SOM-to-DOM mapping technology for a number of applications. This technology uses the configurational isomorphism between SOM and various DOMs to derive the latter – automatically – from the former (Mahdavi and Mathew 1995, Mahdavi et al. 1997a, Wong and Mahdavi 2000). We have not made a claim that such a mapping technology works for all domains and independent of the informational resolution of the pertinent inquiries. Instead, we have argued that the question, if and to which extent such mappings can be automated, must ultimately be decided on an empirical basis. For a certain class of applications working with information at a certain level of resolution, the S2 project has successfully demonstrated the feasibility of homology-based SOM-to-DOM mapping.

5 THE 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. Based on The S2-Kernel's functionality, the S2 applications that depend on the information embodied in SOM, are guaranteed a valid representation of the building. The input to the S2-Kernel is provided by the Graphical User Interface (GUI). S2-Kernel processes this input in view of SOM's informational requirements. This processing typically involves the services of a geometry reasoning engine.

S2-Kernel has the ability to use different databases and run on either the server or the client machines. On the server, a multi-user object database is utilized to store the SOM. This makes the SOM available to the rest of S2 (e.g. applications). When simulations are to be run, the applications communicate with the S2-Kernel. This allows the GUI to go "off-line". On the client side, The S2-Kernel enhances the GUI by providing off-line viewing, editing and preparation capabilities. On users' computers, S2-Kernel uses a lightweight single-user database for SOM persistence. Furthermore, as a client, S2-Kernel provides a caching mechanism that optimizes connections between the server and remote clients. This cache is used to keep the SOM until it is ready for upload to the server. The transfer happens with a single call (objects are sent by-value), incurring the network delay penalties associated with the call only once. This cache is not needed and may be by-passed in network environments where delays are negligible.

As mentioned earlier, SOM holds mostly topological information. The semantic information is kept in "type" objects (e.g. construction and program information, material properties) that qualify the building elements. These type objects are kept in separate databases that can be customized and/or maintained by manufacturers. Only a tag to the "type" is appended to a list of tags for each object and there is no restriction on the number of tags that a building element can hold. This provides a simple plug-in mechanism for future applications.

6 GEOMETRIC REASONING

The Geometric Modeling Engine is a common utility for geometric processing. The S2 environment currently uses a research tool (GRAIL) for solid geometry modeling (Stouffs 1994). The GRAIL kernel, on its own, provides the following functionalities: *a)* determining if two polyhedrons overlap, touch or one contains the other, *b)* adding/combining two or more polyhedrons into one (union), *c)* determining the common polyhedron of two sets of polyhedrons (intersection), *d)* subtracting one set of polyhedron from another (difference), *e)* determining the intersection of a polyhedron and an infinite plane (section) and similar functionalities on plane and line segments. However, these generic operations had to be made useful for S2 requirements, i.e.: *a)* facilitating the treatment of complex building designs, *b)* ensuring the geometric/topological integrity of building representation (SOM) based on the user input, *c)* providing the geometric/topological operation needed to generate domain-specific

building representations (DOMs) based on SOM. Toward this end, a number of query schemes (sequence of computational steps) were developed.

To exemplify such a query scheme, consider the derivation of simulation representation (DOM) for airflow analysis based on SOM. This involves the generation of a 3-dimensional mesh of finite control volumes, whereby nodes belong to indoor space, space boundaries, or outdoor space. The essential idea is to use indoor-outdoor node couples as the starting point. Once such couples are identified, the two cells associated with this node couple can be merged to create an expletory transient double cell volume. The intersection of these expletory volumes with SOM spaces allows for the derivation of the necessary geometric attributes (shape, area, tilt, etc.) of enclosure elements of the building's spaces.

For airflow simulation, the pre-simulation processing commences by the automatic generation of the nodal network for the building. This involves the identification of the indoor and outdoor nodes. The indoor nodes that belong to various spaces are also tagged. The linkage paths joining the nodes are established by applying the pertinent boundary conditions. For example, in the case of building envelope and internal partitions, the linkage paths joining the nodes adjacent to the envelope/partitions are disconnected (Wong and Mahdavi 2000).

7 USER INTERFACE

The graphical user interface (GUI) allows users to create and edit building descriptions. The GUI could eventually be a CAD system. Current CAD models typically lack sufficient information to provide the necessary input for the S2-Kernel. For example, most CAD systems do not have a "space" (room) entity and frequently represent the building only as a configuration of walls and slabs. However, most simulation applications depend on an understanding of "space" as a "void" with enclosure components' surface as its boundary. Moreover, the space element is used as a container for many semantic attributes that are used during simulation (e.g. type of activity, occupancy pattern). S2 has currently its own GUI. This component was designed to be simple and lightweight for simple editing and/or viewing on computers where CAD systems may not be available. Even if, in the future, CAD systems are integrated, this GUI could find use as an auxiliary component of the system.

The current GUI uses a 2½D representation. The input is simplified. The user inputs the building by drawing space elements on a canvas. The space properties define the height of a space as well as its function, schedules, construction of walls, sizes of the windows, etc. The user interface provides defaults for the properties of design elements. This allows the input to be ready for simulation immediately after the completion of building geometry input. Of course the user can override the defaults and specify his/her own semantic component information prior to simulation. When the design is ready for analysis, the user can select the simulation(s) to be run, confirm the default settings, or choose others. The GUI also allows for viewing the results after simulations are completed. The extensive use of default is meant to simplify the user input and ensure that S2 keeps a consistent model (ready for simulation) at all times. In future, these defaults could be replaced with "consistency checking" algorithms that are employed prior to simulation.

In a sense, S2's current GUI allows users to work with a "simulation view" of a building design. Future CAD systems may integrate this "simulation view" allowing users to "switch" to this view on-demand, connect to simulation engines, and receive performance feedback. This is in fact analogous to rendering a model, itself a kind of simulation.

The current S2 GUI is designed modularly and is able to work "on-line" as well as "off-line". In the "off-line" mode, for the time being, the GUI is restricted to editing/viewing projects that exist on the local machine. Once "on-line", the GUI is able to upload/download projects to/from servers and initiate simulation sessions. The GUI does not need to wait for simulations to be completed and can disconnect after simulations start. The results can be viewed later. In the future users may be able to download simulation applications and be allowed to run simulations in "off-line" mode as well. Of course, this will be dependant on platform availability of the application.

8 HUB AND THE DATA REPOSITORY

The Hub is the communication coordination center. It is the entry point to the rest of the system for users. Its address is made known to everyone. It authenticates users, keeps track of available resources, and connects users to resources on-demand.

S2-Kernel uses a modular interface for data access. Although any database can be used with the system, currently S2 is using an object-oriented database for its central, multi-user repository. This repository not only holds project (design) information, but also organizes "type" objects that provide various semantic qualifications to building elements. Currently, S2 uses a simple strategy to make sure users receive the latest version of a project. Only one user is allowed to modify a project at any given instance. When a project is checked out for modification, other users are only allowed to "read" the project.

9 COMMUNICATIONS

S2 may be seen as an attempt to establish an "open" system that could grow into a standard information bus for building simulation applications. It has a modular architecture that allows for future extensions. S2 uses the Object Management Group's (OMG) Common Object Request Broker Architecture (CORBA) technology for communication between its various components (OMG 1999). The major advantage of CORBA is its language and platform-independent nature. Most of S2 applications are developed using C++, however the core components and the user interface is developed using Java. CORBA development has enabled these components to come together with relative ease and without any user-written language-mapping code and without restricting users to a specific platform. Thus, it is possible to envision applications – new or existing – from any domain and with any development environment to make use of S2 capabilities.

A first prototypical version of S2 was implemented and tested at CMU (Mahdavi et al. 1999). Currently, the S2 development is pursued within the framework of an international collaboration between three institutions, CMU, NUS, and TP (Mahdavi et al. 2000). The initial configuration of this implementation is as follows (cp. the schematic illustration in figure 1):

The central system is located in NUS. This includes the project database running on one server, the Hub on another. S2-Kernels are spawned on-demand by the Hub for each project that remote GUIs open. Building performance simulation applications for lighting (LUMINA) and environmental impact analysis (ECOLOGUE) are deployed in CMU. Any number of servers may be used. The thermal suite applications (including energy analysis (NODEM), HVAC and air-flow analysis (BACH) modules) are deployed in TP. Users in all institutions have access to a GUI regardless of their location. Although all institutions can maintain the "type" libraries that are relevant to their applications, a master set is maintained at NUS.

When an application or a database server starts up, it registers with the Hub and makes its location on the network known. Users access S2 by establishing a connection and authenticating with the Hub at NUS. To run simulations, users retrieve a list of available applications from the Hub and specify those to be employed. The Hub coordinates the simulation process by sending the project information to the appropriate servers. It also notifies the user as results become available. Users need not know the actual locations of applications.

10 ILLUSTRATIVE USE CASES

SEMPER Prototype-1, provided a proof-of-concept for integration of multiple simulation applications within a design tool (Mahdavi 1999). But it still was a stand-alone, single-user system, designed to run on a single computer. This restricted its use in the real world where the building delivery process often involves distributed teams of professionals that must constantly exchange information. S2 is thus engineered to support scenarios where a single design alternative can be accessed by multiple – geographically distributed – users. Moreover, S2 is designed with both users and developers in mind. While it is important to provide a usable system, it is just as important to make it maintainable and extendable.

User Perspective

Users are not limited to any given location. They can be anywhere and can use any computer. With the current GUI, the minimum requirement from a user computer is the ability to support a Java runtime environment. In the future, when S2 will be integrated with commercial CAD systems, users will be able to access S2 through their own CAD system. However, even then, on computers where a CAD system does not exist, users will be able to utilize other "viewers" with different capabilities. One could be a direct descendant of our current GUI where editing is possible. Others might be even simpler applets running inside web browsers for dedicated tasks such as exploration of simulation results or reviewing building characteristics.

With a GUI, the user can enter the building design information. When access to S2 services such as simulation applications, project storage and/or comprehensive type libraries is needed, a network connection will be established (if one does not already exist). The user will connect to any S2 system by specifying the URL of its Hub. Upon authentication, the user will be able to use any of the services offered by the system. A project can be downloaded or uploaded enabling information sharing with other users, simulations can be started or results can be downloaded for review. When a simulation is started, the user does not need to wait for it to be completed and can disconnect at any time. The hub will automatically coordinate the collection of results and their persistent storage for the user.

This architecture relieves users of software maintenance and keeps this responsibility where it should be: With system administrators and developers. However, the system's architecture does not stop a specialist who wants "off-line" access to one or more specific applications from installing them on his/her computer along with the GUI. This is of

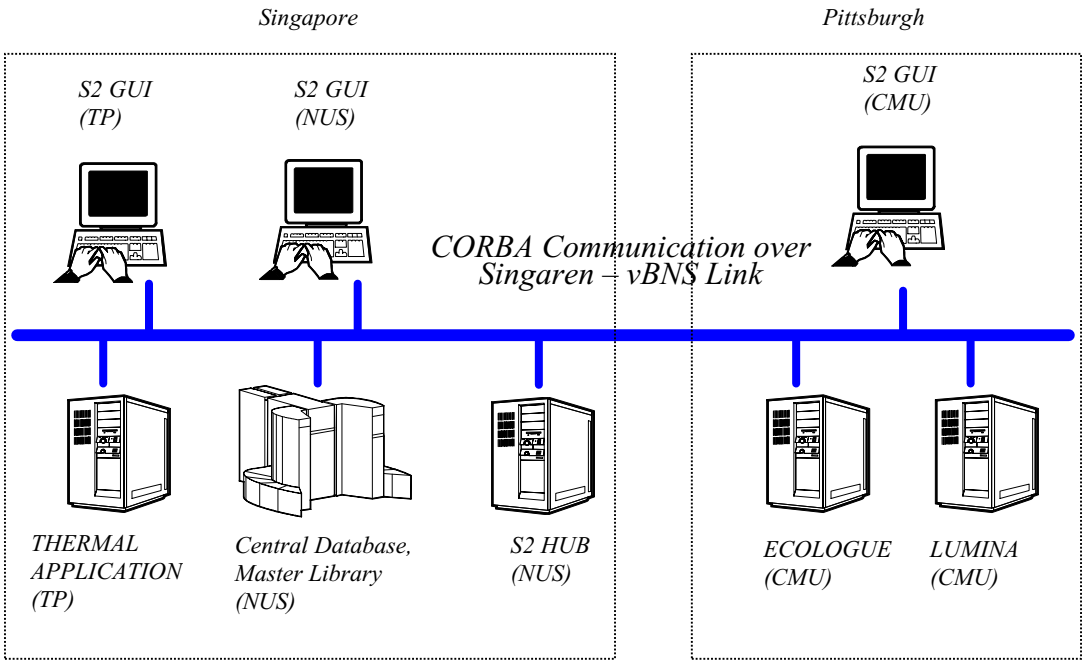

Figure 1- Schematic illustration of components of S2 in the case of two users in Singapore and the U.S. collaborating on a building design/engineering project.

course limited to the availability of the application for the user's platform of choice. Also, the user would be responsible of updating the application.

Currently, a user in Singapore is able to connect to a system in US and download a project, edit the design, upload it, start a simulation on an application running in Singapore, disconnect, and later return to review the results.

Developer perspective

Today, a user can open a spreadsheet file inside a word-processing document. This is made possible by Object Linking and Embedding, a technology that enables applications to treat each other as ordinary objects. These objects become service providers for each other. Similarly, distributed computing aims to achieve this beyond the boundaries of a single machine. S2 follows this idea to create an "object-web" of simulation applications that is available to developers regardless of geographical location. A developer in Asia will be able to make use of an application in North America and extend its functionality and provide a richer set of services to the users. For example, an energy analysis application might make use of a remote daylight simulation application to predict the effects of daylight-based dimming of the electrical lighting and building energy use.

The S2 development was informed by the obvious lack of collaboration among the building performance community where, each year, many applications are developed, to be abandoned and forgotten soon thereafter. Because of the lack of proper computing resources and time constraints, most research still produces stand-alone solutions unprepared for integration within larger schemes. S2, by providing a communication infrastructure that includes a standard user interface along with input and output mechanisms, allows developers to concentrate on their own domain and produce reusable services by building on already available ones.

A developer who is working on a new simulation application (or integration of a legacy application) is able to develop and test the application by following these steps:

a) Download an S2 software development pack. This pack will include the following: A CORBA IDL (Interface Definition Language) specification for SOM, a client version of the S2-Kernel, and the basic GUI.

b) Using the IDL specification, generate language-binding code (depending on the language of choice for the application) through a preferred compiler. Using the generated language binding, develop the domain mapping code. The IDL specification will act as an API (application programming interface) reference document throughout this stage.

c) Through an interface that connects to the remote S2 server, prepare and register necessary containers (if any) that will hold simulation-specific parameters in the database. For example, if a fire-safety

application is being developed, new objects such as *inhabitant* and new attributes such as *throughput capacity* (for doorways) may be needed. The flexible nature of SOM accommodates for such additions without necessitating an intervention by administrators.

d) Develop the user interface components that are necessary to input these new simulation parameters as JavaBeans (plug-in) and activate them in the user interface.

e) Run the GUI, and test the application.

Currently S2 does not fully support steps *c* and *d*. Since S2 infrastructure itself is still under development, such dynamic behavior is currently not needed.

12 FUTURE WORK

Once the S2 Infrastructure is fully developed and tested, the research will have to focus on its use especially in collaborative environments. Efficient versioning strategies are needed within the database to allow for the exploration of alternative design solutions and to facilitate parametric studies. The implications of use patterns arising from the interactions of designers and disciplinary experts for the future of S2 evolving distributed system architecture are of highest interest.

Also, S2 must be tested in the classroom. Its potential as an educational tool has to be explored in detail. Such testing will not only benefit S2's future development, but could also contribute to educating a new generation of designers and engineers who understand the significance of readily useable simulation tools and demand for CAD systems that would provide simulation services on an integrated basis.

APPENDIX A: THE THERMAL SUITE IN S2

S2 currently has applications for building energy modeling (including simple HVAC and multi-zone airflow analysis), thermal comfort assessment, lighting simulation, and environmental impact analysis. *The thermal suite* is a "super-application", involving three simulation modules, namely NODEM, HVAC and BACH; for building energy analysis, HVAC system simulation, and air flow analysis, respectively. These modules are realized as distributed simulation objects, accessible through the web. CORBA technology (Common Object Request Broker Architecture) is used as the communication framework between the modules. In addition, a central coordinating application (Thermal_App) is implemented, which acts as a controller and mediator between the modules. The three modules of the thermal suite can run independently or in tandem (cp. Table 1), depending on the nature of the user's performance query.

Table 1 – Activated thermal suite modules as a function of the performance assessment scenario

Simulation scenarios	***NODEM***	***HVAC***	***BACH***
Comprehensive thermal analysis	✔	✔	✔
Climatized building with sealed envelope	✔	✔	
Passive building with natural ventilation	✔		✔
Load calculation	✔		

The modules

NODEM (Mahdavi and Mathew 1995, Mathew 1996), the energy analysis module, uses the same heat-balance technique as detailed heat-balance simulation programs (Clarke 1985), but is designed in a manner that allows for operation with a "coarse" representation of the building. According to the local weather data and operation schedules, NODEM computes heating, cooling, and electrical loads as well as temperature profiles.

The HVAC module (Brahme 1999) has a consistent modeling approach throughout the building design process, which is applicable in the early design process and yet comprehensive enough as an evaluation tool. Its use of a component-based modeling approach makes it scalable and allows designers to model new systems in addition to modeling pre-defined system types. The HVAC module computes the properties of the supply and return air for a given operation schedule. It calculates the system energy consumption and is also capable of generating a default terminal layout and distribution network of typical HVAC systems.

BACH (Mahdavi et al. 1997a, Wong and Mahdavi 2000, Wong 1998) airflow model follows a hybrid multi-zone and CFD approach. It allows three-dimensional analysis of airflow and contaminant dispersal. Given building geometry information (spaces, enclosures, and openings), BACH uses wind and temperature difference information to compute the air change rate for each space. Automated conversion of general building design representation into an appropriate domain representation for air flow analysis is provided for both orthogonal and non-orthogonal building geometries.

Since different simulation modules may or may not run in tandem, the Thermal_App regulates the sequence of execution. It *a)* allows CORBA-based connections to other S2 components, *b)* is responsible for the time step coordination, and *c)* facilitates data exchange between the thermal suite modules at each

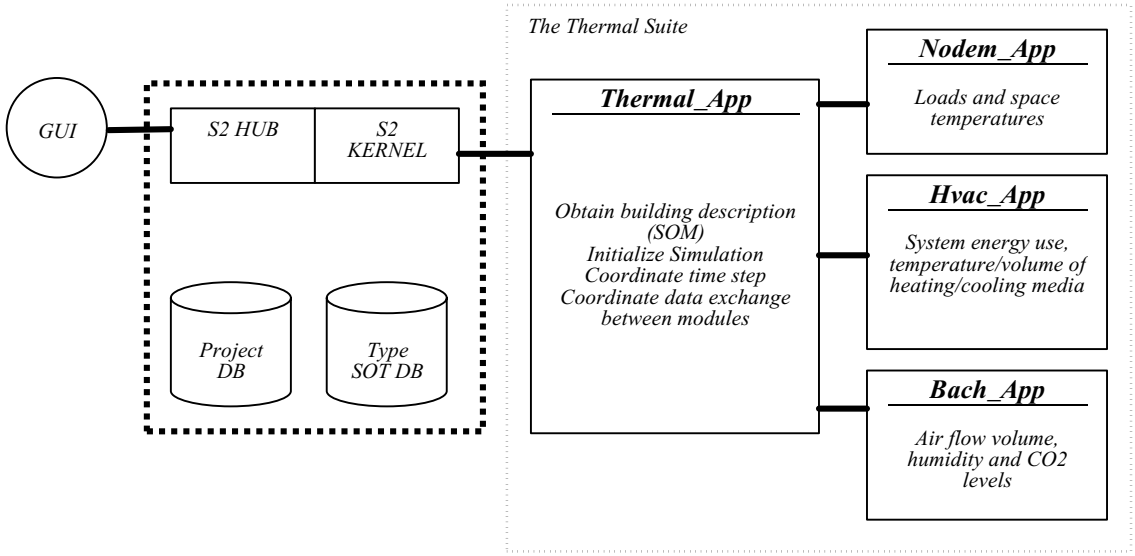

Figure 2 - Schematic illustration of the thermal suite's architecture

time step over the network. The Thermal_App obtains the specifics of the client's thermal assessment query and instantiates the pertinent module(s) of the thermal suite accordingly.

Distributed communication

Using CORBA, the distributed modules of the thermal suite can communicate independent of the language they were written in, following the design philosophy of the S2 system. Therefore, the modules that are all written in C++, can be called from a client GUI written in Java. The wrapper provided through CORBA's IDL (Interface Definition Language) allows for this communication. In order for a client to run the simulation modules via Thermal_App, the three modules themselves have to be "objectized" first. The IDL-wrapped modules are called Nodem_App, Hvac_App and Bach_App, respectively. A schematic diagram of the thermal suite's architecture and its position within the S2 environment is given in the figure 2.

Data coupling and module interaction

There are various methods for coupling interactive applications. Specifically, the sequential coupling or the so-called "Ping-Pong" coupling and direct coupling have been discussed in detail in Wong 1998. The "Ping-Pong" coupling, which is an inter-module "iteration" coupling technique, is used for data exchange between the modules of the thermal suite. This method involves running the three simulation modules sequentially over a series of discreet time steps.

Figure 3 illustrates schematically the data exchange between the three modules of the thermal suite (for an air-based HVAC system). NODEM_App sends the temperature data of each cell to both HVAC_App and BACH_App. In return, HVAC_App provides NODEM_App with information on heating/cooling media temperature and volume, while BACH_App provides NODEM_App with air flow volumes due to ventilation and infiltration. HVAC_App also provides BACH_App with supply air conditions (relative humidity, CO_2 concentration) for the purposes of air quality analysis.

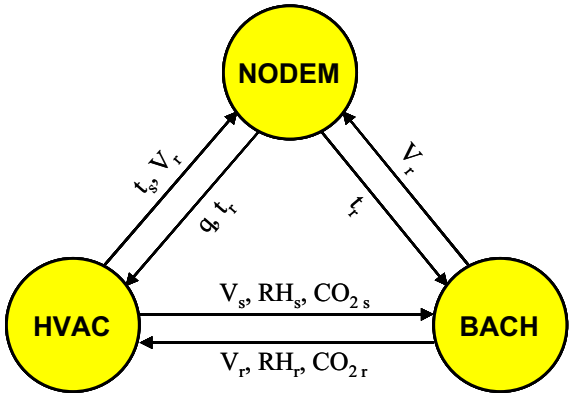

Figure 3 - The data exchange between the thermal suite modules. (q: Space load, t_r: Space temperature, t_s :Supply air temperature, V_r: Air flow volume, natural ventilation V_s: Supply air flow volume, RH_r: Space relative humidity, RH_s: Supply air relative humidity,$CO_{2\,r}$: Space carbon dioxide level, $CO_{2\,s}$: Supply air carbon dioxide level)

In thermal suite's operation, NODEM_App typically initiates the process with the prediction of the zone temperatures based on default assumptions regarding output values of HVAC_App and BACH_App. These temperatures are then passed to the HVAC module, which uses assumed BACH data to calculate

a revised set of output values. Following the initial round of interactions, modules exchange each other's output from the previous time step as their input for the subsequent time step computation. The main advantage of this method is that the different simulation modules can be used separately for specific queries while more complex analysis requests may be accommodated via coupled execution of all modules.

ACKNOWLEDGMENTS

Three teams of researchers are involved in the collaborative effort described in this paper. The members of the CMU team are: A. Mahdavi, R. Brahme, B. Gurtekin, M. E. Ilal. The members of the NUS team are: K. P. Lam, N. H. Wong, and S. Gupta. The members of the TP team are S. K. Hor, K. K. Chan, K. S. Au, and Z. J. Kang. Support for this research has been provided in part by NSTB (Singapore) and ABSIC (US).

REFERENCES

Brahme, R. (1999): Computational Support for Building Energy System Analysis, PhD Thesis, School of Architecture, Carnegie Mellon University, Pittsburgh, PA, USA.

Clarke, J. A. (1985): Energy Simulation in Building Design, Bristol, UK, Adam Hilger Ltd.

Kumar, S. – Mahdavi, A. (1999): A combined analytic and case-based approach to thermal comfort prediction in buildings. Proceedings of Building Simulation ’99. Sixth International IBPSA Conference. Kyoto, Japan. Vol. I. pp. 369 - 376.

Mahdavi, A. (2000): Supporting collaborative design via integrated building performance computing. InterSymp-2000, 12th International Conference on Systems Research, Informatics and Cybernetics, Baden-Baden, Germany. Proceedings: Advances in computer-based and WEB-based collaborative systems. ISBN: 0-921836-88-0. pp. 91 – 102.

Mahdavi, A. (1999): A comprehensive computational environment for performance based reasoning in building design and evaluation. Automation in Construction 8 (1999) pp. 427 – 435.

Mahdavi, A. (1998): A Middle Way to Integration. Proceedings of the 4th Design and Decision Support Systems in Architecture and Urban Planning Conference. Maastricht, The Netherlands.

Mahdavi, A. – Ries, R. (1998): Toward computational eco-analysis of building designs. Computers and Structures 67 (1998) pp. 357 - 387.

Mahdavi, A. – Mathew, P. (1995): Synchronous Generation of Homologous Representation in an Active, Multi-Aspect Design Environment. Proceedings of the Fourth International Conference of the International Building Performance Simulation Association (IBPSA). Madison, pp. 522 - 528.

Mahdavi, A. – Lam, K. P. – Ilal, M. E. – Wong, N. H. (2000): A multi-institutional initiative for distributed collaborative performance-based building design. InterSymp-2000, 12th International Conference on Systems Research, Informatics and Cybernetics, Baden-Baden, Germany. Proceedings: Advances in computer-based and WEB-based collaborative systems. pp. 215 - 224.

Mahdavi, A. – Ilal, M. E. – Mathew, P. – Ries, R. – Suter, G. – Brahme, R. (1999): The architecture of S2. Proceedings of Building Simulation ’99. Sixth International IBPSA Conference. Kyoto, Japan. Vol. III. pp. 1219 - 1226.

Mathew, P. (1996): Integrated Energy Modelling for Computational Building Design Assistance. PhD thesis, School of Architecture, Carnegie Mellon University, Pittsburgh, PA, USA.

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

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

Mahdavi, A. – Liu, G. – Ilal, M. E. (1997b): CASCADE: A Novel Computational Design System for Architectural Acoustics. IBPSA (International Building Performance Simulation Association) Conference, Prague. Vol. II, pp. 173 - 180.

OMG (1999): The Common Object Request Broker: Architecture and Specification; Object Management Group formal documentation. Formal/99-10-07.

Pal, V. – Mahdavi, A. (1999): A comprehensive approach to modeling and evaluating the visual environment in buildings. Proceedings of Building Simulation ’99. Sixth International IBPSA Conference. Kyoto, Japan. Vol. II. pp. 579 – 586.

Stouffs, R. (1994): The Algebra of Shapes, Ph.D. thesis, School of Architecture, Carnegie Mellon University, Pittsburgh, PA, USA.

Wong, N.H. (1998): Computational Air Flow Modeling for Integrative Building Design, PhD Thesis, School of Architecture, Carnegie Mellon University, Pittsburgh, PA, USA.

Wong, N.H. – Mahdavi, A. (2000): Automated Generation of Nodal Representations for Complex Building Geometries in the SEMPER Environment. Automation in Construction 10 (2000) pp. 141 – 153.