An Expandable Software Model for Collaborative Decision-Making During the Whole Building Life Cycle

K. Papamichael, Lawrence Berkeley National Laboratory, USA

V. Pal, Lawrence Berkeley National Laboratory, USA

N. Bourassa, Lawrence Berkeley National Laboratory, USA

J. Loffeld, Lawrence Berkeley National Laboratory, USA

I. G. Capeluto, Technion - Israel Institute of Technology, Israel

Abstract

Decisions throughout the life cycle of a building, from design through construction and commissioning to operation and demolition, require the involvement of multiple interested parties (e.g., architects, engineers, owners, occupants and facility managers). The performance of alternative designs and courses of action must be assessed with respect to multiple performance criteria, such as comfort, aesthetics, energy, cost and environmental impact. Several stand-alone computer tools are currently available that address specific performance issues during various stages of a building's life cycle. Some of these tools support collaboration by providing means for synchronous and asynchronous communications, performance simulations, and monitoring of a variety of performance parameters involved in decisions about a building during building operation. However, these tools are not linked in any way, so significant work is required to maintain and distribute information to all parties.

In this paper we describe a software model that provides the data management and process control required for collaborative decision-making throughout a building's life cycle. The requirements for the model are delineated addressing data and process needs for decision making at different stages of a building's life cycle. The software model meets these requirements and allows addition of any number of processes and support databases over time. What makes the model infinitely expandable is that it is a very generic conceptualization (or abstraction) of processes as relations among data. The software model supports multiple concurrent users, and facilitates discussion and debate leading to decision-making. The software allows users to define rules and functions for automating tasks and alerting all participants to issues that need attention. It supports management of simulated as well as real data and continuously generates information useful for improving performance prediction and understanding of the effects of proposed technologies and strategies.

Keywords: Decision making, integration, collaboration, simulation, building life cycle, software.

1 Introduction

The phenomenal growth of information technologies has revolutionized the way we do business. These technologies now give us the ability to collect, manipulate, and disseminate massive amounts of data, offering decision makers the opportunity to access detailed information, often at the speed of thought. Current multimedia and networking technologies allow formatting and communication of information so that, using the Internet, anyone in the world can broadcast information that is instantly accessible to anyone else in the world!

The ability to quickly and inexpensively generate, store, and communicate vast quantities of information means that decision making can be based on access by many parties to extensive information. The building industry envisions using this technological capacity to make information available about all details of buildings' design, construction, and operation throughout their life cycle. The key question is how software can make this data management possible.

In this paper we identify key elements involved in decision making and then model them in ways that support the development of an expandable software model for collaborative decision making about buildings throughout their life cycles.

2 Background

Currently, many software applications can individually address partial needs of the building industry (e.g., visualization, lighting, energy, construction management, etc.). Digital drawings have become the norm, and are becoming increasingly "smarter" through links to object-oriented representations of building components and systems and their descriptive and performance characteristics. Computer-based simulations often allow very accurate performance prediction for a variety of criteria, such as comfort, aesthetics, energy, safety, environmental impact, and economics (Birdsall et al. 1990; Feustel 1992; Ward and Shakespeare 1998; http://www3.autodesk.com/adsk; http://www.lightscape.com/; http://www.lighting-technologies.com/Lumen_Micro.htm; http://www.primavera.com/). This information is critical to decision making during the entire life cycle of the building.

Various individual applications are now available, and more are on the way, to assist with tasks and decisions at different stages of the life cycle of the building, from schematic to detailed design, to construction, commissioning, operation, renovation, retrofit, and demolition (Cambell 1998; Clayton et al. 1998; Piette 1996). However, most available applications are stand-alone, without means of exchanging information with other related programs.

Several attempts are under way to integrate such applications or at least to allow them to exchange information. Although development is ongoing, significant potential has already been demonstrated for tremendous increases in efficiency and effectiveness of the decision making process (Mahdavi et al. 1996; Pohl et al. 1992; Papamichael et al. 1997; Jokela et al. 1997; http://iaiweb.lbl.gov/). Some efforts have been focusing on the development of applications that facilitate collaboration over networks (Kalay 1997; McCall et al. 1998). Some of these capabilities have already been integrated into commercial applications (http://www.bentley.com/products/projbank/dgn/index.htm).

In this paper, we describe an integrated approach that supports collaborative decision making throughout a building's life cycle. This approach involves the use of a very abstract decision-making model as the basis for the development of data and process models to address the specific needs of the various disciplines involved in building design, construction, and operation. The high degree of abstraction means that the model can, in theory, expand infinitely as the application-specific data and process information about a building grows.

3 Theoretical Conception of the Decision Making Process

In this section we present the theoretical considerations that are the foundation for the proposed model. These include our conceptualization of the decision making process and the required inputs to it.

3.1 Decisions

Decisions can be abstracted into *selections among options*, and thus require comparison. From this viewpoint, the main elements of decision making are options (at least two) and selection criteria. In other words, we make decisions by evaluating options with respect to various performance criteria and then choosing the option that best fits our preferences. For example, an architect may select from a number of glazing options based on aesthetics, view, and/or energy implications. Depending on the nature of the selection criteria, we use various processes to predict the performance of alternative options and then evaluate predicted performance by comparing among options.

Decisions become increasingly difficult as the options and trade-offs among selection criteria increase in number. Most building-related decisions involve multiple criteria and significant trade-offs among them. Moreover, they involve multiple players who have varying concerns and priorities and need to collaborate to predict and evaluate building performance. For example, a dynamic relationship is necessary among a building's architect, HVAC engineering consultant, and structural engineer.

3.2 Debate

The performance of alternative options can be abstracted into *advantages* and *disadvantages*, or *pros* and *cons*, which decision makers *weight* to form *preferences* among options. We cannot quantify how people go through the complex choosing process that entails both thinking and feeling (Papamichael and Protzen 1993). Decisions made by multiple collaborating parties often involve *conflicts* among preferences, i.e., different parties prefer different options. Person A may prefer glazing option 1 because of its aesthetic appeal while person B may prefer glazing option 2 because of its superior energy performance. The final decision is made by the most powerful player(s) in the decision making process who is generally influenced by the debate.

Decisions may also involve *factual* conflicts related to the predicted performance of options. These types of conflicts occur because of different *assumptions*, either inherent in different performance prediction methods or related to the information that is used as *input*, e.g., rule-of-thumb variations for HVAC design, or use of inappropriate weather data for computation of thermal loads.

All conflicts result from the availability of multiple values for a single parameter, or multiple *positions* on an *issue*. Issues are resolved through debate, i.e., formulation of arguments *for* and *against* positions, which is the equivalent of the *pros* and *cons* described above for the performance of alternative options (Kunz and Rittel 1970).

3.3 Parameter Types

The parameters that characterize the options considered in decision making for buildings are usually referred to as *design* parameters. The parameters considered when selecting among options are referred to as *performance* parameters. These depend not only on design parameters but on *context* parameters as well, i.e., parameters that characterize the conditions under which an option is considered.

Context parameters describe not only the *existing* conditions at the time of the decision but also the *assumed* conditions during the upcoming phases of the building's life cycle when *actual* performance will be realized. The values of design parameters are usually chosen with a goal of improving performance relative to one or more performance criteria. This relationship between design and performance parameters is referred to as *design intent*. The intent to improve performance with respect to certain criteria usually results in degrading performance with respect to other criteria, which introduces trade-offs among available options. For example, darkening the color of a wall finish to meet an aesthetic criterion will likely result in less reflected light and potentially greater demand for electric lighting.

Design intent is usually formed through combinations of several design parameters into a *strategy*. This is true at any level, from building components and systems, to the whole building itself, where the combination of the values of multiple design parameters is expected to produce the desired performance, rather than the value of any single parameter alone.

3.4 The Building Life Cycle

Although decisions are identical in nature throughout the building life cycle, they vary dramatically, not so much with respect to performance parameters but mostly with respect to the design and context parameters. During the design phases of the building life cycle, design parameters reflect mostly building characteristics and context parameters reflect mostly site characteristics. After construction, design parameters mostly reflect the operation of the building and relatively small changes in the details of the building, while the building itself becomes part of the context for these decisions. For example, the size of a window, which was a design parameter during the building design, becomes the context for decisions to control glare or temperature once the building is built. However, most of the parameters don't change, they simply switch from describing design to describing context.

The fact that the parameters involved in decisions throughout the building's life cycle are unchanging allows formulation of a model that can be used both during design and once a building is built. It also introduces the potential for a third type of conflict, which occurs when the actual value of a parameter is different from the one assumed during earlier phases. Such *expectation* conflicts can occur in any of the parameters involved in decision making. The major objective of commissioning a building is to identify such conflicts and make decisions based on the actual context.

Even though most parameters involved in decision making are unchanging during the entire life cycle of a building, the detail required for and the persons involved with each one may change. In most buildings, even today, a very large amount of information that is generated during the building design is not available to decision makers during construction, commissioning, and operation. Changes are often made without knowledge of the original design intent, which may have a significant negative effect on performance. This is especially true when the changes involve individual parameters that are part of a larger strategy. For example, darkening the color of a wall may interfere with a daylighting strategy that depended on the reflectance of the lighter-colored wall.

3.5 Formulating Options

The formulation of options is the creative part of design and a prerequisite to decision making. Although creativity can be defined as the invention of new approaches, there are strong arguments to suggest that many building-related decisions entail combinations of existing approaches and components. For example, the design of an HVAC system can be seen as the combination of readily available components. Even when custom-made components are used, they usually have similar characteristics to standard ones. This is less true for the form of the building and the arrangement of spaces.

Today, many building design decisions involve selection among readily available building components and systems, which have traditionally been available in the form of catalogs and are now becoming available in electronic form either through centralized efforts (http://www.sweets.com/; http://www.thomasregister.com/) or individual manufacturers. The Internet offers the opportunity for continuously updated information on building components. In some cases, the information is already in digital form that is compatible with available tools, such as CAD or simulation software, which can further automate consideration of options during design.

3.6 Model Requirements

A model that will address decision making during the entire life cycle of the building must include the design, context, and performance parameters used in all methods of performance prediction and evaluation of characteristics of a building at any time in the building's life cycle. Because most design and context parameters affect multiple performance aspects and remain the same during the life cycle of the building, the model must be *integrated*. It must address the data needed for predicting performance by multiple tools during the entire life cycle of the building.

Since the number of performance prediction methods is large and new methods and options are being developed continuously, the model needs to be *expandable*. This requires the development of a *meta-schema* (i.e., a structural framework for expansion of the model) that can incorporate data to allow the building schema (data and processes) to grow. The meta-schema must also include a *model of time* to support not only the dynamic parameters related to the building's operation (e.g., occupancy), but the static parameters of components (e.g., a window) that may be replaced during the life cycle of the building.

Because performance evaluation requires comparison, the model must support maintenance of multiple options. Moreover, it must support the integration of information about existing buildings, which forms the general context for evaluation of the predicted performance of proposed designs. Finally, the model must address opportunities for *automation*, not only for preparation of input and handling of the output of performance prediction methods, but also for assigning values to design and context parameters. For example, a designer may define aesthetic facade rules for a project and the model will then automatically assign properties, such as glazing color.

From a user's point of view, the model should answer questions, which means providing the values of design, context and performance parameters, along with the sources of these values and the arguments that support or negate them.

4 Proposed Implementation

In this section we describe the proposed implementation of the theoretical considerations described in the previous section for the development of an expandable model that will support collaborative decision making throughout a building's life cycle.

4.1 The Data Meta-Schema

The foundation of the proposed model is an integrated, object-oriented representation of both data and processes, in the form of a data meta-schema. The meta-schema is used to define and create data and process objects necessary for decision making during the entire life cycle of the building.

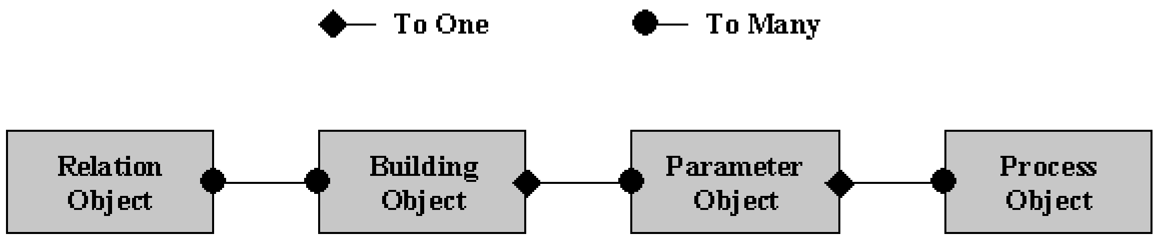
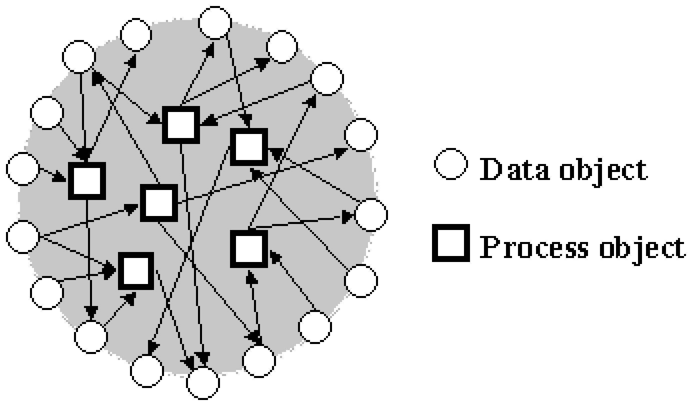


Figure 1. A meta-schema where building objects are related to each other through relation objects, and process objects are related to parameter objects through input and output relations.

The building is modeled in terms of *building objects* that are related through *relation objects* and are characterized by *parameter objects*. Processes are also modeled as objects related to parameter objects through input/output relations. (Figures 1 and 2). A process may vary from a complex simulation engine that accepts a large number of input data and computes a large number of output data, to a simple if-then-else rule with minimal input and output. Data and processes can be added to this environment without restructuring the code of the meta-schema because the code operates on a model with a very generic conceptualization (or abstraction) of processes as relations among data rather than on the specific contents of data and processes.

Modeling of Time

The issue of how to model the passage of time is especially critical for addressing the needs of the whole life cycle of a building. A building can go through multiple states during its lifetime. Each parameter can therefore take on multiple values according to the states of the building through time. In addition, each of these values may have resulted from different processes or measurement equipment, which may each use different representations of time. We need a model that is general enough to be mapped on variations of time models used by different software tools. In response to this, we include time in the data meta-schema.



The proposed time model includes a *starting* point in time, an *ending* point in time, a *reference* point in time, and a *time step* (Figure 3). The reference point can be fixed in real time, e.g., 12 AM, January 1, 1973. The time step can be described in terms of a multiplier and a time unit, e.g., 0.001 seconds, or 20 years. Making the time step (i.e., the time resolution) into a variable is helpful for addressing multiple processes with varying time step requirements. Some processes, such as the DOE-2 building energy analysis tool, may need a time resolution on the order of hours. Other tools, such as the SPARK HVAC modeling tool, may need a time resolution on the order of seconds or milliseconds to model HVAC controls. The proposed model allows translation to and from any representation of time, handling even time periods such as seasons, weekdays, and weekends.

Figure 2. A data schema based on the meta-schema results in processes being modeled as links among data.

4.2 The Building Data Schema

In this section we describe how the data meta-schema is used to define and create the data schema that holds the data objects and processes to address the data needs for decision making during the building's life cycle. The data schema contains data and processes that address the specific need of the building industry. Process object instances (e.g., simulation tools, rules, data queries, etc.) serve as relations among data object instances (e.g., spaces, walls, windows, etc. and the parameters that characterize them).

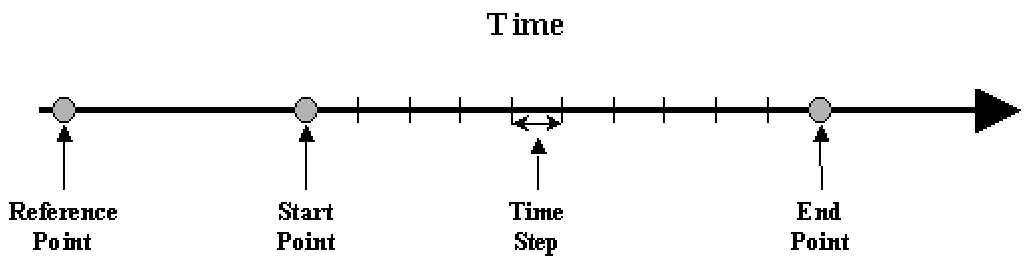
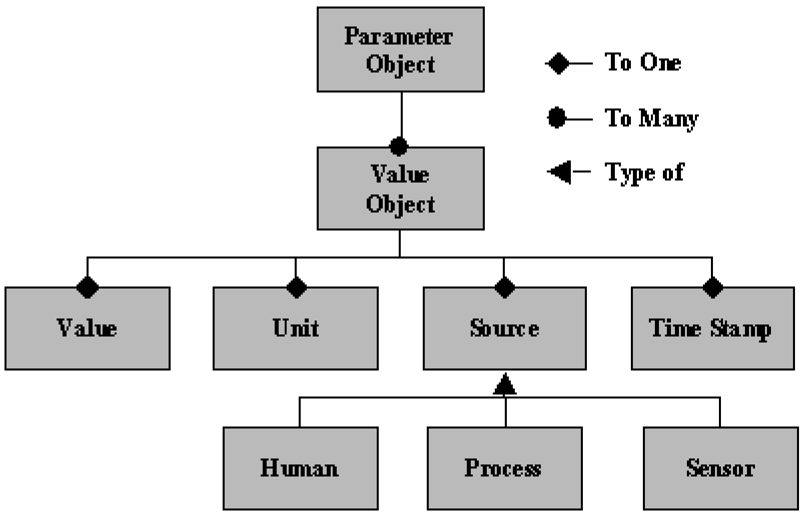


Figure 3. A general model for time that allows translation to and from any representation of time.



Modeling Data

The objects that will hold the building data are modeled in the form of interrelated objects. Building objects, such as spaces, walls, and windows are modeled as software objects that are linked to each other through relations (composed-of / part-of, has / owned-by, etc.). Building objects can then be created at any point and related to the rest of the building objects. The relations among objects are also modeled as software objects, so new relations can be defined as needed. The same is true for parameters that characterize building objects. Modeling parameters as software objects allows the creation of new parameter objects at any point, as required by the addition of new processes.

Figure 4. The data meta schema allows the creation of multiple values, each with its own source, for each parameter.

To support links to multiple processes and address the data needs of the whole life cycle of a building, even the values of parameters are modeled as software objects. Thus, each parameter can have multiple values, which may come from different sources at different times during the building's life cycle. The "value" data type includes data fields for the source of the value, the unit, a time stamp, the value of the parameter itself, and the time(s) for which the value is relevant. Value sources include humans, processes, and sensors (Figure 4).

Because evaluating predicted performance requires comparison among options, the whole building model is part of a "project" object, so that multiple options can be created and then compared for any of the parameters that characterize them. The creation of a new option for the whole project happens only when any user assigns different values to the same object or parameter. In that case, the system automatically alerts all interested parties, who may respond by "arguing" for and against options. If the new values for an object or parameter come from processes or sensors, these can be used to either create new project options, or value ranges (rather than single values as explained in Section 4.3).

External Databases

The definitions of building objects (e.g., space, wall and window) in the schema database are used to create alternative options and store them in external databases. *Composite* objects, i.e., objects composed of other objects, such as a "window" composed of a "frame" and "glazing", are stored in object-oriented databases. *Terminal* objects, i.e., building objects such as the "glazing" characterized only by parameter objects such as "transmittance", "reflectance" and "U-value" can be stored either in object-oriented or relational databases. These databases can be distributed and dynamic, that is, available on the Internet and continuously updated by manufacturers of building components and systems (for design information), or services and organizations (for context and performance information). External databases can be used to select options for building components and systems as well as to specify the values of context parameters during the development of the project database for a particular building, as illustrated in Figure 5.

Because external databases are closely tied to the building schema database, any expansion of the latter must be reflected in the external databases. An expansion of the building model is required when a new process is added to it and needs data (input or output) that are not available in the model. These data may be added to the building model, but the values for them will not be automatically available in the external libraries of building components and systems, or contextual databases. This is especially true for input parameters. For output parameters (e.g., heat flow through window glazing) the processes themselves provide the values. For example, if the schema includes glazing parameters for transmittance and reflectance and we add a process that requires the U-value of the glazing, the model will be expanded to include U-values. The external databases for glazing will also have to be updated to include U-values.

Modeling Processes

Processes are treated in the same way as data. A process is abstracted as an object that has one essential characteristic: when given values for a set of input parameters, it produces values for a set of output parameters. A process is created in the model in the same way that data objects are created, by defining relationships between it and the parameter objects of the data model. If a process needs data that are not already available in the building model, then the latter is expanded by adding the required data objects.

A process may be a complex simulation engine, such as DOE-2 or Radiance, or a low-level piece of procedural code such as the formula for computing surface area. Even simple rules can be modeled as processes: the input parameters to the rule are the parameters involved in the “if” block of the rule, and the output parameters are the ones involved in the “then” and “else” blocks of the rule (Figure 6). Even data queries can be modeled as processes. The search criteria are the input parameters, and the query results are the output. The output of a query may be a list of options to be considered as input to other processes.

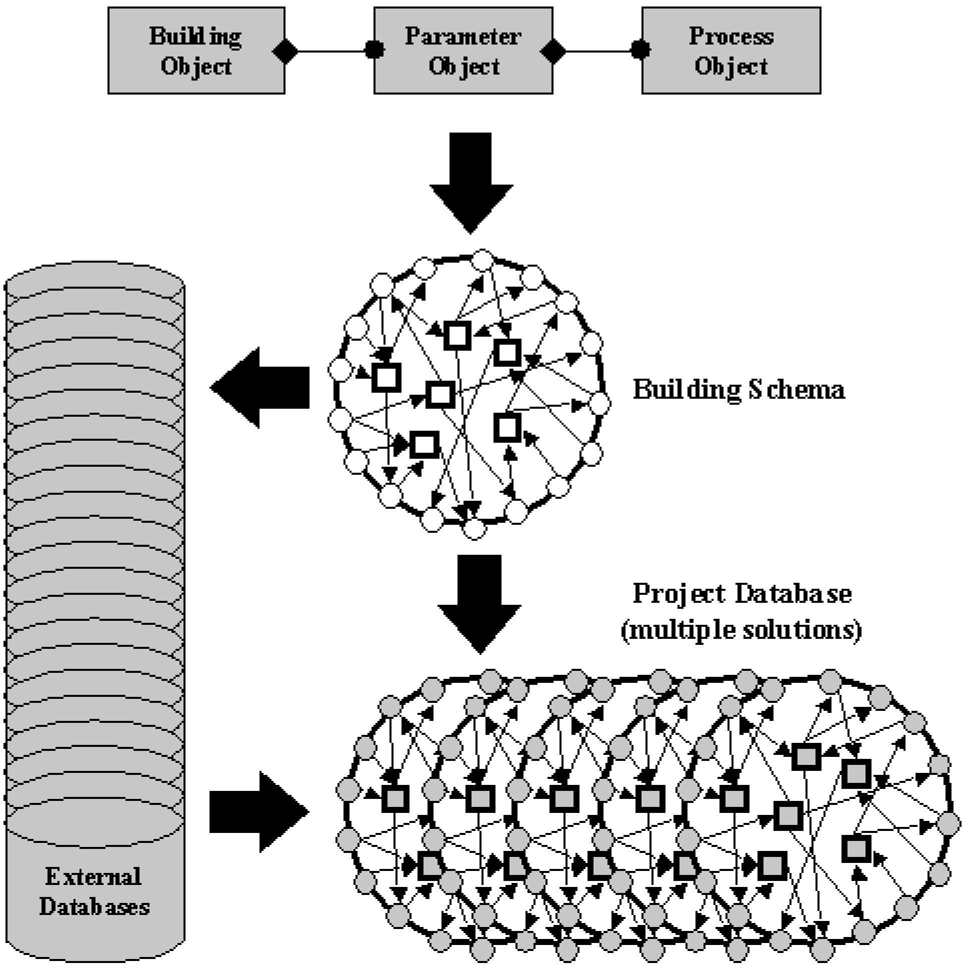


Figure 5. The relationships among the data meta-schema, the building data schema, the project database, and the external databases.

This very abstract model supports the automatic activation of processes and the use of the output of one process as input to others. Thus, the operation of the model can be managed by a relatively simple kernel that is independent of the contents of data and processes.

4.3 *The Operation Kernel*

The operation kernel manages the data and processes for the maintenance of the model. Two abstract types of user actions activate the kernel: the “assignment to” and “request for” values of data objects. Users, sensors, or processes assign values, and users or processes request them. When a value is assigned and/or requested, a chain of processes is activated to reflect the specified change in the design or to compute the requested value. This chain follows the input and output links among data and processes, as is explained in the following sections.

Assigning Values to Data Objects

Users and processes can either create new data objects within a project or change the values of existing data objects. When a new data object is created, the related objects and parameters required for that object are automatically generated following the definition of the data object in the data schema. When a new window is created, for example, the data schema is queried for the required objects and parameters, such as the frame and glazing, which are, in turn, created automatically.

The data objects that are created reflect the data needs (input and output) of the processes that have been already defined as part of the model. The values for these data objects are either entered by the users or specified by default through the use of *preference* rules. Preference rules assign values to data objects following design practices of users or design firms and can grow as users and firms gain experience based on feedback from the system, i.e., simulation results and actual measurements during building operation. Preference rules can also represent codes and standards, such as ASHRAE 90.1, and Title 24. Finally, preference rules can also be used to activate other rules or sets of rules.

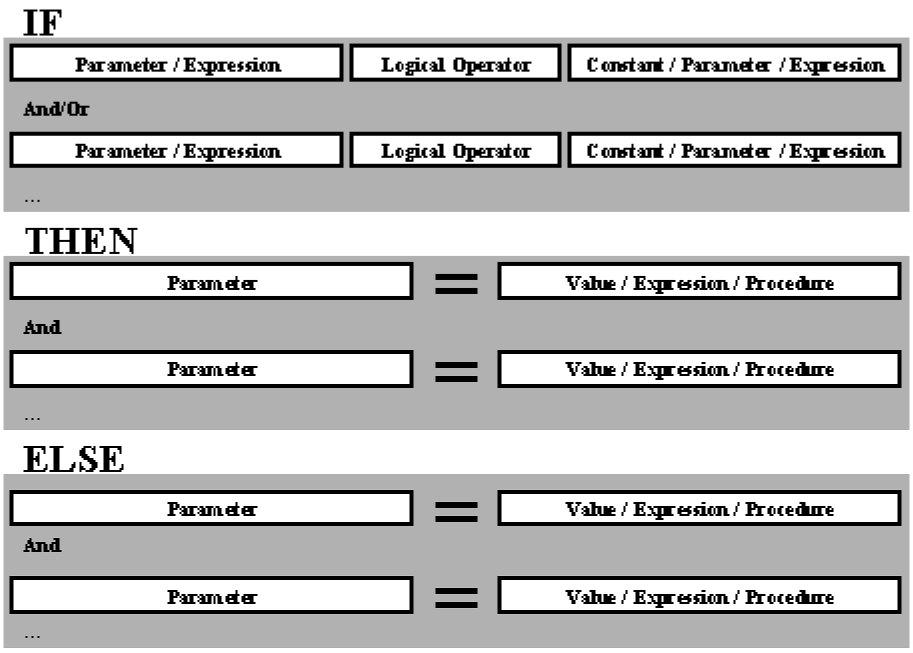

IF
Parameter / Expression
Logical Operator
Constant / Parameter / Expression
And/Or
Parameter / Expression
Logical Operator
Constant / Parameter / Expression
...
THEN
Parameter
=
Value / Expression / Procedure
And
Parameter
=
Value / Expression / Procedure
...
ELSE
Parameter
=
Value / Expression / Procedure
And
Parameter
=
Value / Expression / Procedure
...

Figure 6. If-Then-Else Rule structure.

In addition to preference rules, *constraint* rules may be activated by the assignment of values to data objects. Constraint rules do not assign values to data objects. Instead, they check the validity of assigned values and notify users when discrepancies occur, e.g., when the window width is assigned a value that is greater than the width of the parent wall. When not used for the assignment of default values, preference rules can also play the role of constraint rules, notifying users when a preference is not met. This is further explained in Section 4.3.3., which describes the handling of conflicts and the assignment of values by sensors.

Preference and constraint rules may also activate simulation processes when the values of performance parameters are required as input. Processes are activated automatically when the value of an output parameter is needed. They can also be set for automatic activation when the value of one of their input parameters is changed. This triggers a forward chaining inference mechanism as illustrated in Figure 7.

Requesting Values Of Data Objects

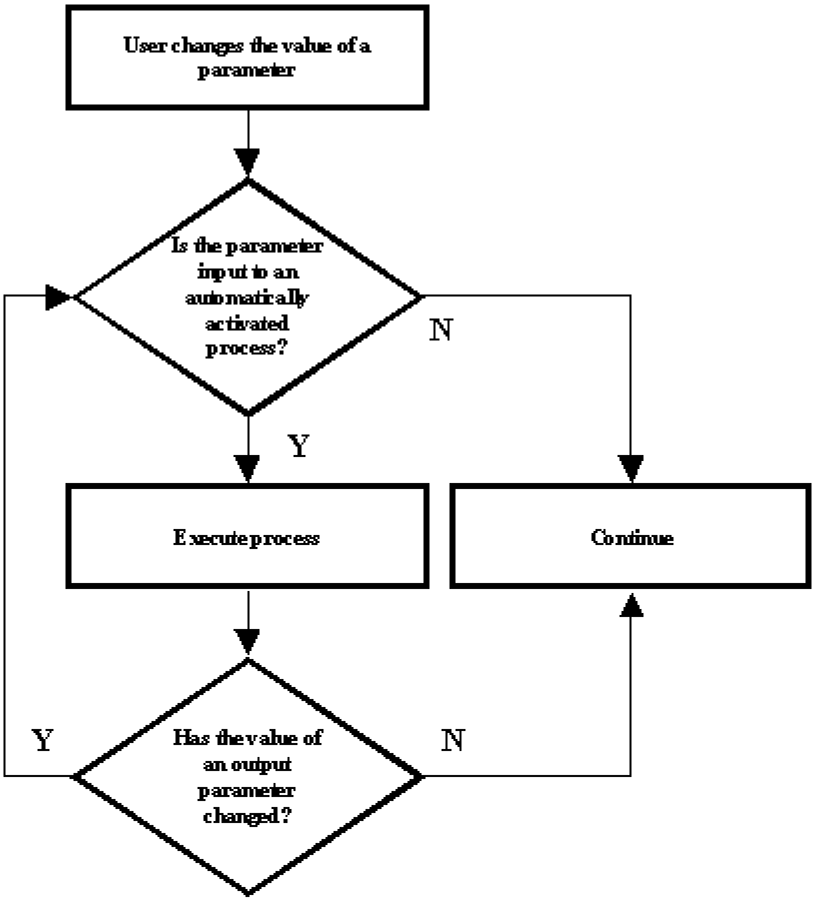

User changes the value of a parameter
Is the parameter input to an automatically activated process?
N
Y
Execute process
Continue
Has the value of an output parameter changed?
Y
N

Figure 7. Activation of processes following the assignment of a value to a data object.

The value of a data object may be requested either by the user or by a process that needs it as input. In either case, the kernel first checks to see whether the value is already in the project database. If it is not, then the kernel looks for processes that compute the requested value as output. The processes can either be simulations (for values of performance parameters) or rules (for values of design and context parameters). If no process is available, the kernel prompts the user for a value. If one or more processes are available, the kernel recursively requests the values for all input parameters, stacking processes as necessary (Figure 8).

If there is more than one process that computes a requested value as output, the user is notified and can select the process or processes to be activated. If more than one process is selected, the kernel either generates *new project options* (when the output values are for design or context parameters) or *new values* (when the output values are for performance parameters) generating a value range for the expected performance.

Addressing Conflicts

A conflict occurs when a data object gets different values from different sources. Having multiple values is not in itself a problem because the system allows for multiple values. In many cases it is helpful to maintain multiple values that allow for model validation, parametric analyses, and systematic comparison of design solutions. However, users need collaborative control over the selection of multiple values and design options. The system allows each user to select which values to retain and which values to reject and to associate arguments in the form of comments with each value. The following four conflict scenarios are possible:

1. Different users and/or preference rules assign different values for the same design parameter.
2. Different users or processes, e.g., data queries or rules, assign different values to context parameters.
3. Different performance prediction simulations compute different values for performance parameters.
4. Measured values, e.g., those noted during construction and/or operation of the building, are different from the intended, assumed, or expected values of design, context, and performance parameters, respectively.

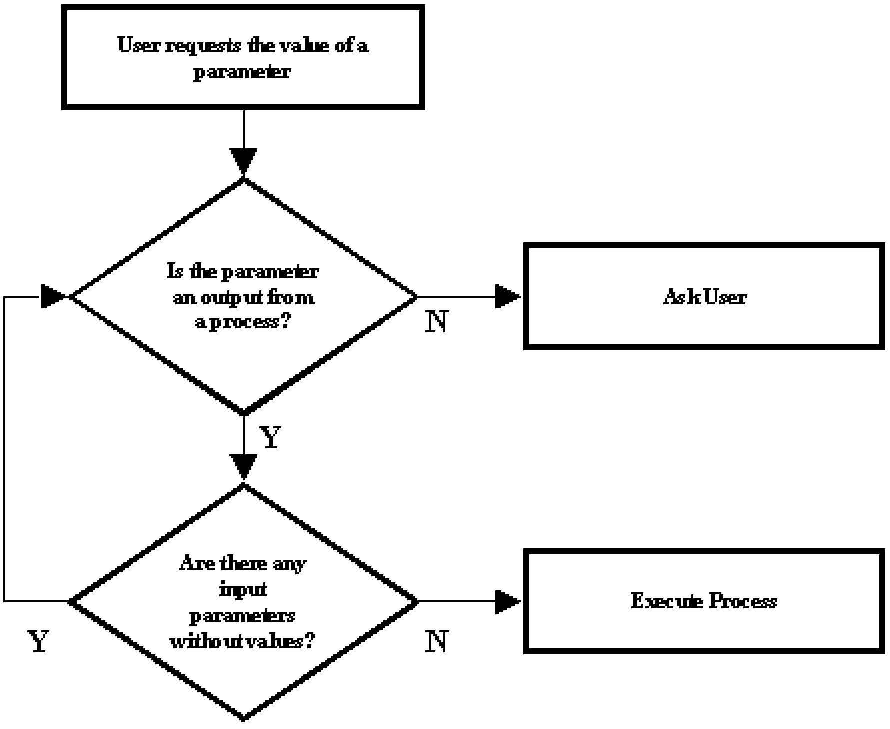

User requests the value of a parameter
Is the parameter an output from a process?
N
Ask User
Y
Are there any input parameters without values?
Y
N
Execute Process

Figure 8. Activation of processes following a request for the value of a data object.

In the first case, the kernel alerts the users, who may agree on one of the assigned values or define different project options, i.e., alternative designs that can be further explored. Users are also given the opportunity to argue for and against the different options, referencing the values of performance parameters. In the second and third cases, the kernel alerts the users, who may select one of the available values as the most valid, or retain multiple values to create a confidence range in place of a single value (Figure 9).

In the fourth case, the kernel alerts the users, who have several options for resolving the issue, depending on the type of parameter under consideration. Conflicts in design parameters indicate discrepancies in the design and are resolved through reconsideration of specific decisions, which may involve requesting values for performance parameters and considering additional design options. Conflicts in context parameters once a building is constructed indicate discrepancies in assumptions about the context and can be used to improve context assumptions in future projects. Conflicts in performance parameters once a building is built indicate discrepancies in expected performance and can be used to reconsider and/or adjust the performance prediction methods for better accuracy in future projects.

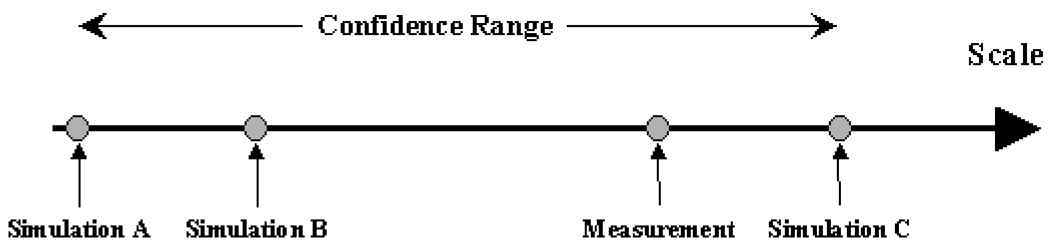


Figure 9. Multiple values for context and performance parameters are used to create confidence ranges.

The number of values for a particular parameter can grow over time and be stored in the database as life cycle information. Storage of values facilitates systematic comparison among them at different stages of the building's life cycle. Being able to compare values from different sources and times can facilitate troubleshooting and validation/improvement of performance prediction methods. For example, a simulation tool may provide values that are consistently higher than the corresponding measured values, or trends may be observed for measured values over time and can be used to diagnose problems with the building's operation.

5 Conclusion

The expanded use of computers and the communications revolution of the Internet offer unique opportunities to address the data needs of the whole building life cycle. In this paper we presented a software model for the integration of multiple processes and databases throughout the life cycle of a building, which will allow multiple participants to share information and make decisions. Significant work is needed for the implementation of the described model into a working tool and even more work for testing and validation. Successful implementation will eventually require development of standards for the electronic description of building components and systems, as well as building context parameters. Such efforts have already been underway (http://iaiweb.lbl.gov/). We hope that the thoughts and modelling approaches presented in this paper will contribute towards the realization of the overall vision.

Acknowledgements

This work was supported by the Assistant Secretary for Energy Efficiency and Renewable Energy, Office of Building Technology, State and Community Programs, Office of Building Systems and Office of Building Equipment of the U.S. Department of Energy, under Contract No. DE-AC03-76SF00098.

References

Birdsall, B.E., Buhl, W.F., Ellington, K.L., Erdem, A.E. and Winkelmann F.C. (1990). "Overview of the DOE-2 building energy analysis program, version 2.1D." Lawrence Berkeley Laboratory report LBL-19735, Rev. 1, Berkeley, CA.

Cambell, D.A. (1998). "Architectural construction documents on the Web: VRML as a case study," *Proceedings of ACADIA '98 Conference*, Quebec City, Canada, October 22-25, pp. 267-275.

Clayton, M.J., Johnson, R.E., Song, Y., and Al-Qawasmi, J. (1998). "Delivering facility documentation using intranet technology," *Proceedings of ACADIA '98 Conference*, Quebec City, Canada, October 22-25.

Feustel, H. E. (1992). "Annex 23 multizone airflow modeling – an international effort," Proceedings of the International Symposium on Air Flow in Multizone Structures, Budapest, Hungary.

Jokela, M., Keinänen, A., Lahtela, H., and Lassila K. (1997). "Integrated building simulation tool RIUSKA," *Building Simulation,* Prague, Czech Republic.

Kalay, Y.E. (1997). "P3: An integrated environment to support design collaboration," *Proceedings of ACADIA '97 Conference*, Cincinnati, Ohio, October 3-5.

Kunz, W. and Rittel, H. (1970). "Issues as elements of information systems (IBIS)," Working paper 131, Institute of Urban and Regional Development, CED, UC Berkeley.

Mahdavi, A., Mathew, P., Lee, S., Brahme, R., and Kumar, S. (1996). "On the structure and elements of SEMPER," *Proceedings of ACADIA '96 Conference*, Tucson, Arizona, October 31 – November 2, 1996.

McCall, R., Holmes, S., Voeller, J., and Johnson, E. (1998). "World Wide Presentation and Critique of Design Proposals with Web-PHIDIAS". *Proceedings of ACADIA '98 Conference*, Quebec City, Canada, October 22-25.

Papamichael, K., LaPorta, J., and Chauvet, H. (1997). "Building Design Advisor: automated integration of multiple simulation tools." *Automation in Construction*, Vol. 6, pp. 341-352.

Papamichael, K. and Protzen, J.P. (1993). "The Limits of Intelligence in Design," *Proceedings of the Focus Symposium on Computer-Assisted Building Design Systems*, Fourth International Symposium on System Research, Informatics and Cybernetics, Baden-Baden, Germany.

Piette, M.A. (1996). "Commissioning Tools for Building Life-Cycle Performance Assurance," LBNL Report-38979, presented at the *4th National Conference on Building Commissioning*.

Pohl, J., LaPorta, J., Pohl K.J., and Snyder J. (1992). "AEDOT Prototype (1.1): An Implementation of the ICADS Model." Technical Report CADRU-07-92, CAD Research Unit, Design Institute, School of Architecture and Environmental Design, Cal Poly, San Luis Obispo, CA.

Ward, G. and Shakespeare, R. (1998). *Rendering with Radiance: The Art and Science of Lighting Visualization*. Morgan Kaufman.