AN AGENT APPROACH TO DATA SHARING IN VIRTUAL WORLDS AND CAD

Mary Lou Maher, Pak-San Liew and John S. Gero
Key Centre of Design Computing and Cognition, University of Sydney
mary@arch.usyd.edu.au

SUMMARY

This paper describes an agent approach to sharing and synchronising building model data among CAD and virtual world systems. Virtual worlds facilitate a level of communication and collaboration not readily available in conventional CAD systems. The integration of virtual worlds and CAD systems using a common data model can make a significant impact on synchronous collaboration and real time multi-user multi-disciplinary modification of building data. By using agents, the integration of virtual worlds and conventional CAD systems can go beyond that of passive data transfer. Data within a central database is monitored and the relevant applications using the data are notified automatically of any changes through sensors and effectors embedded within agents that define their interface to the database. We use an object-oriented EDM database as the central repository of data and Active Worlds as the virtual environment that is coordinated with the shared data within the database. An interface agent is being developed to connect the virtual world to the database to allow active data access and modification. This agent approach can be extended to the integration of other applications and data models.

INTRODUCTION

The construction industry has long recognised the need for specialised software for specific needs, even though this variation in software has limited the ability for information to be seamlessly transferred from one discipline or one software application to another. The development of standards such as the Industry Foundation Classes (IFC) provides an opportunity to facilitate data transfer by specifying building data as objects with attributes that include geometric and non-geometric information. Although this development has been slow, the positive aspects of the approach include the object centred representation of building data and the inclusion of non-geometric data.

In addition to the need for data sharing, there is a need for data visualisation that supports communication and collaboration. With recent developments in multimedia and the WWW, we are seeing project management and document sharing have an impact on design and construction processes. These developments have made it easier to transfer files and to interact with the visualisation data. However, they have not yet made an impact on synchronous collaboration and real time multi-user modification of building data.

In this paper we present two concepts that will facilitate data sharing in the building industry: virtual worlds and agents. Virtual worlds support real time multi-user access to 3D models in which each user can independently walk around and through the model while talking to other users and making changes to the shared model. Agents are autonomous programs that can reason about their environment and make changes to the environment. While virtual worlds introduce yet another specialised application with their own data models, agents allow us to build on and go beyond the development of standards to develop an approach to data sharing that is proactive.

IFC AND EDM DATABASE

The Industry Foundation Classes (IAI, 2000) provide a standard object-oriented representation for life-cycle information in the AEC industry. This standardization allows the exchange and sharing of information across different applications based on a common object model. To facilitate data sharing, the information created by different applications according to IFC is normally stored within a central database where it is accessed and modified as required. The Express Data Manager Developer

(EPM, 2002) database is an object-oriented database that provides direct support for modelling, application development and database management of data defined according to the IFCs. It uses EXPRESS as a data description language to define the representation of a model. The EDM database supports model conversion and merging through the definition of mappings between schema and objects and it allows the definition of "business objects" for creating specific views of a model. With the definition of rules, an EDM database also allows objects to have automatic behaviours. This paper uses an EDM database as a central database for the integration of CAD models and virtual environment models using IFCs as the common model.

VIRTUAL WORLDS

A virtual world is a multi-user networked 3D virtual environment. Most virtual worlds have been developed for the entertainment industry, but we are beginning to see virtual worlds for educational and professional uses. 3D virtual worlds support communication and collaboration in a place-like context. The virtual worlds that we are considering are object-oriented systems that associate a 3D model and a behaviour with each element of the world. Examples of such worlds include: Active Worlds (http://www.activeworlds.com) and VirTools (http://www.virtools.com). For the remainder of this paper, we will describe our agent approach using an adaptation of the Active Worlds platform.

In addition to the virtual world providing a shared environment, the people in the world are represented as avatars that can walk around the world and make modifications to the world. For example, new 3D objects are added to the world by copying an existing object, moving it as required, and editing a dialog box to configure it, as shown in Figure 1. The dialog box allows the world builder to specify a 3D model and a script that describes the behaviour of the object. The 3D models can be taken from a standard library provided by Active Worlds, or can be generated in a 3D modelling package and added to the library.

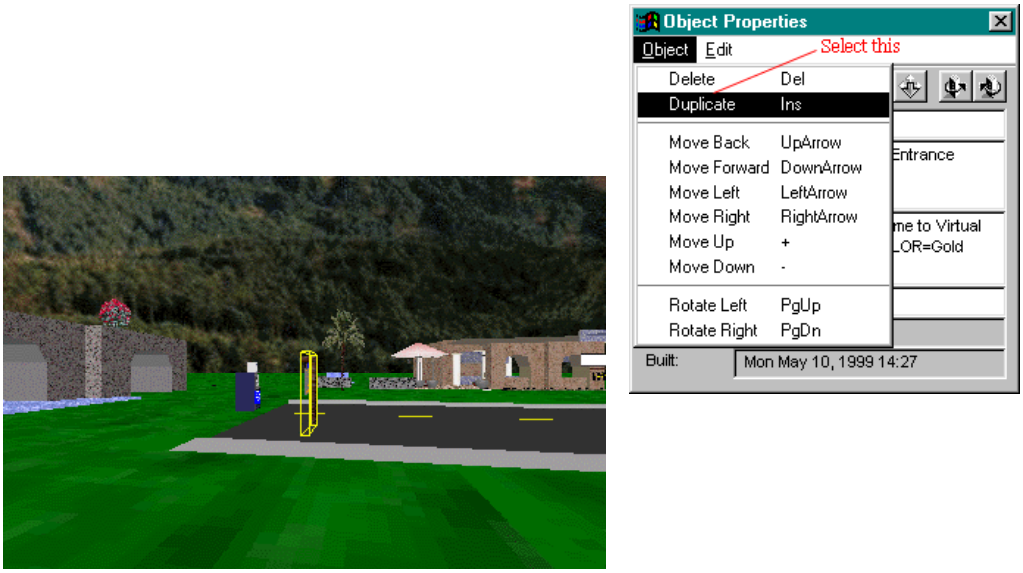

Figure 1. Inserting or modifying an object in a virtual world

Benefits of using virtual worlds for sharing a visualization of the building model is the ease of the use of the interface and the inherent capability as a multi-user environment. CAD systems have complex interfaces that support the development and maintenance of the complex geometric models and views of the building. Virtual worlds are intended as multi-user collaborative systems and therefore provide a simpler interface that allows communication and limited modifications to the geometry of the model. As a client-server application, the data about the world is maintained on the server and the object data is transferred as separate small files to each client as needed. All rendering is done on the client side so that walkthroughs need not be defined ahead of time. Since the object data is transferred in small object files, there is no need to wait for large files to download.

AN AGENT PARADIGM

Agent-based computing started in the 1970s, and recently the concept of agents has become important for internet applications, drawing ideas from Artificial Intelligence and Artificial Life. There is no universal definition for the term agent. However in the context of computer science, agents as intentional systems operate independently and rationally, seeking to achieve goals by interacting with their environment (Wooldridge and Jennings 1995). This idea of an agent is illustrated in Figure 2.

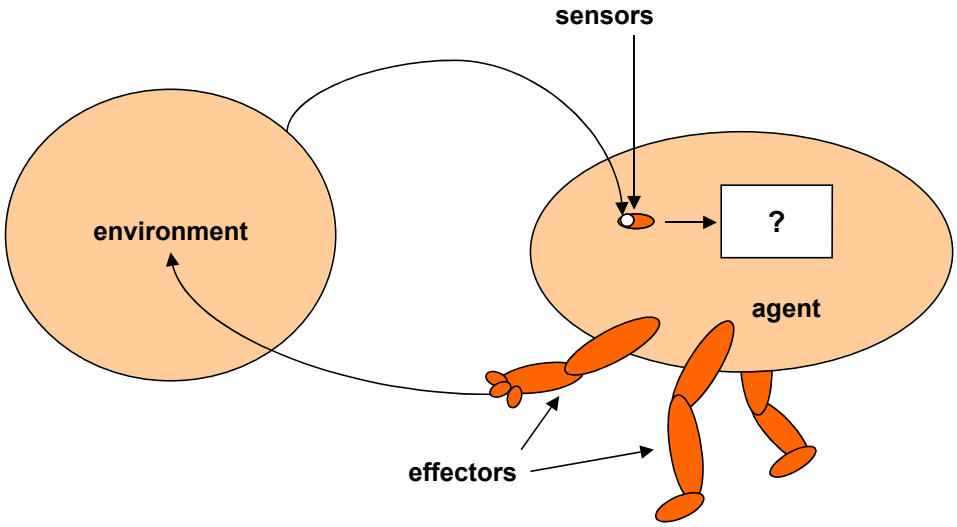

Figure 2. An agent interacts with its environment through its sensors and effectors

An agent has the ability to operate usefully by itself, however the increasing interconnection and networking of computers is making this situation rare. Typically, the agent interacts with other agents (Huhns and Stephens 1999). Hence the concept of multi-agent system is introduced with the applications of distributed artificial intelligence.

Object-oriented programming is one of the major types of programming methods. In object-oriented systems, objects are defined as computational entities that encapsulate some states, are able to perform actions, or methods on this state, and communicate by message passing. There are similarities between agents and objects, but there are also significant differences (Wooldridge 1999):

- Agents embody a stronger notion of autonomy than objects, and in particular, they decide for themselves whether or not to perform an action on request from another agent.
- Agents are capable of flexible (reflexive, reactive, reflective/proactive and social) behaviors, and the standard object model has nothing to say about such types of behaviors.
- A multi-agent system is inherently multi-thread, in that each agent is assumed to have at least one thread of control.

The intelligence of agents also reflects on its direct interaction with multi-agent environments. (Huhns and Stephens 1999) summarize the characteristics of multi-agent environments:

- Multi-agent environments provide an infrastructure specifying communication and interaction protocols.
- Multi-agent environments are typically open and have no centralized designers.
- Multi-agent environments contain agents that are autonomous and distributed, and may be self-interested or cooperative.

We have developed a multi-agent system as the core of a 3D multi-user virtual world. Each object in the world is an agent in a multi-agent system. The agent model provides a common vocabulary for describing, representing, and implementing agent knowledge and communication. Our common agent model has sensors, effectors, and five kinds of reasoning: sensation, perception, conception, hypothesizer, and action. This agent model is illustrated in Figure 3.

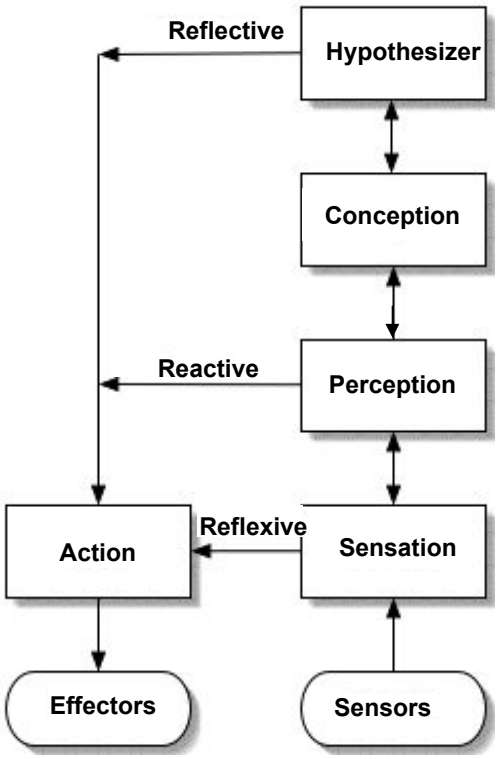

Figure 3. An agent model with 5 processes (after Maher and Gero 2002)

The components of the agent model are described below.

- Sensors recognize two kinds of events: sense_data in which the agent identifies relevant data by monitoring the virtual world and the common EDM data model, and receive_data in which the agent receives a message from another agent. An example of sense-data for a virtual world agent is a change in an object in either the collaborative virtual world, or the EDM database.
- Sensation transforms raw input from the Sensors into structures more appropriate for agent reasoning and learning.
- Perception is a process that finds grounded patterns of invariance in the agent's representation of the sense data. For example, a concrete wall agent may know about patterns in the aspect ratios of the dimensions of the wall. Perception is goal driven by concepts that the agents knows about and data driven by the sense-data.
- Conception associates concepts with sense data or patterns of sense data. Concepts are abstractions of experience that confer a predictive ability for new situations. The concept of a load bearing wall, for example, is a representation of the behaviours and functions of the wall agent, and its meaning is its predictions of possible interaction with other building elements.
- Hypothesizer identifies mismatches between the current and desired situation, which goals are relevant to the current state of the building models and reasons about which goal should be achieved in order to reduce or eliminate that mismatch. It identifies possible actions which when executed will change the building model to meet those goals.
- Action reasons about which sequence of operations on the building model, when executed, can achieve a specific goal.
- Effectors are the means by which actions are achieved. Two types of effectors are: Change_data in which the agent causes a direct change to the virtual world or EDM database, and SendMsg_data in which the agent sends a message to another agent to respond by changing the world.

This agent model is derived from recent developments in cognitively-based design agents, where design is considered as a situated act (Gero 1998). The agents are developed to interact with the design and the design knowledge (Smith and Gero 2001; Saunders and Gero 2001). The agent

approach to virtual worlds provides for new kinds of interaction among the elements of the virtual world representation and between individuals and project teams with the components of the virtual world that makes both the virtual environment and interactions with it dynamic.

Agents can function in three modes based on their internal processes: reflexive, reactive, and reflective. Each mode requires increasingly sophisticated reasoning, where reflexive is the simplest. These modes are indicated in Figure 3 by labels on the paths through specific agent processes. A simpler reasoning involves fewer agent processes.

- Reflexive mode: here the agent responds to sense data from the environment with a pre-programmed response – a reflex without any reasoning. In this mode the agent behaves as if it embodies no intelligence. Only pre-programmed inputs can be responded to directly. Actions are a direct consequence of sense data. This mode is equivalent to the kinds of behaviors that are available in current virtual worlds.
- Reactive mode: here the agent exhibits the capacity to carry out reasoning that involves both the sense data, the perception processes that manipulate and operate on that sense data and knowledge about processes. In this mode the agent behaves as if it embodies a limited form of intelligence. Such agent behavior manifests itself as reasoning carried out within a fixed set of goals. It allows an agent to change the world to work towards achieving those goals once a change in the world is sensed. Actions are a consequence not only of sense data but also how that data is perceived by the agent. The agent's perception will vary as a consequence of its experience.
- Reflective mode: here the agent partially controls its sensors to determine its sense data depending on its current goals and beliefs; it also partially controls its perception processes again depending on its current goals and beliefs; its concepts may change as a consequence of its experiences. The concepts it has form the basis of its capacity to "reflect", ie not simply to react but to hypothesize possible desired external states and propose alternate actions that will achieve those desired states through its effectors. The reflective mode allows an agent to re-orient the direction of interest by using different goals at different times in different situations (Gero and Kannengiesser 2002).

In the following section we show how this agent model can be used for data sharing between CAD systems and virtual worlds.

DATA SHARING THORUGH THE USE OF AGENTS

A central theme of data sharing is the access to shared data contained within a central database. One common approach to accessing this data is through the use of a facade design pattern (Shalloway and Trott 2001) described in (Gamma et al. 1995).

A facade provides a high-level interface that encapsulates many other subsystem interfaces to the database. This differs from normal database through the Application Programming Interface (API) provided by the database, Figure 4. The access to the database via a facade provides a layered structure where users of the database need not have any knowledge of the underlying API of the database. This layered structure also helps to isolate changes as the versions of the database or its associated API gets upgraded of modified.

The facade approach to database access is not designed for active data access. A user has to manually access the data that reside passively within the database via the functionalities provided by the facade when he/she has knowledge about a modification made to the data. An additional functionality has to be built on top of the facade to facilitate active data access. This functionality needs to be encapsulated within a coherent framework that includes the facade and not simply be added as a patch.

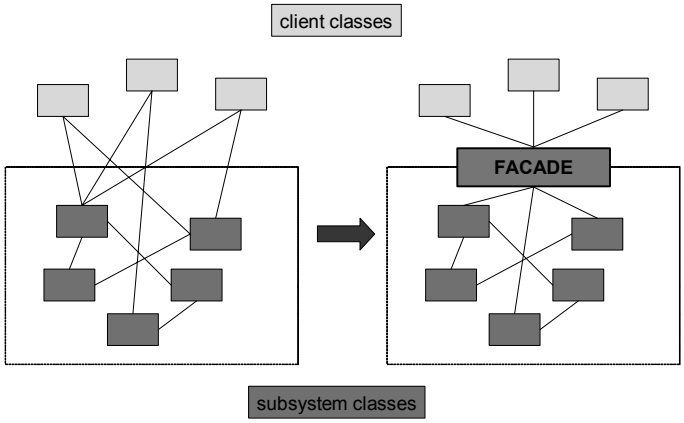

Figure 4. Database access via API and via a façade

Figure 5 illustrates a framework for data sharing based on the use of the facade design pattern and the agent paradigm. The interface to a virtual world that facilitates active data access is described. The interface agent encapsulates the connection between the virtual world and the object database where shared data are stored. There is an interface agent for each object type in the EDM object database and the Virtual World. For example, "walls" agent coordinates the data about walls in the EDM and Active Worlds models. Although we illustrate this interface agent for coordinating the EDM and Active Worlds, other applications can be included. Each agent is composed of a reasoning component with a set of sensors and effectors related to the database and another set of sensors and effectors related to a specific application, that is, the virtual world. The reasoning component provides the knowledge for perception, conception, hypothesizing and action.

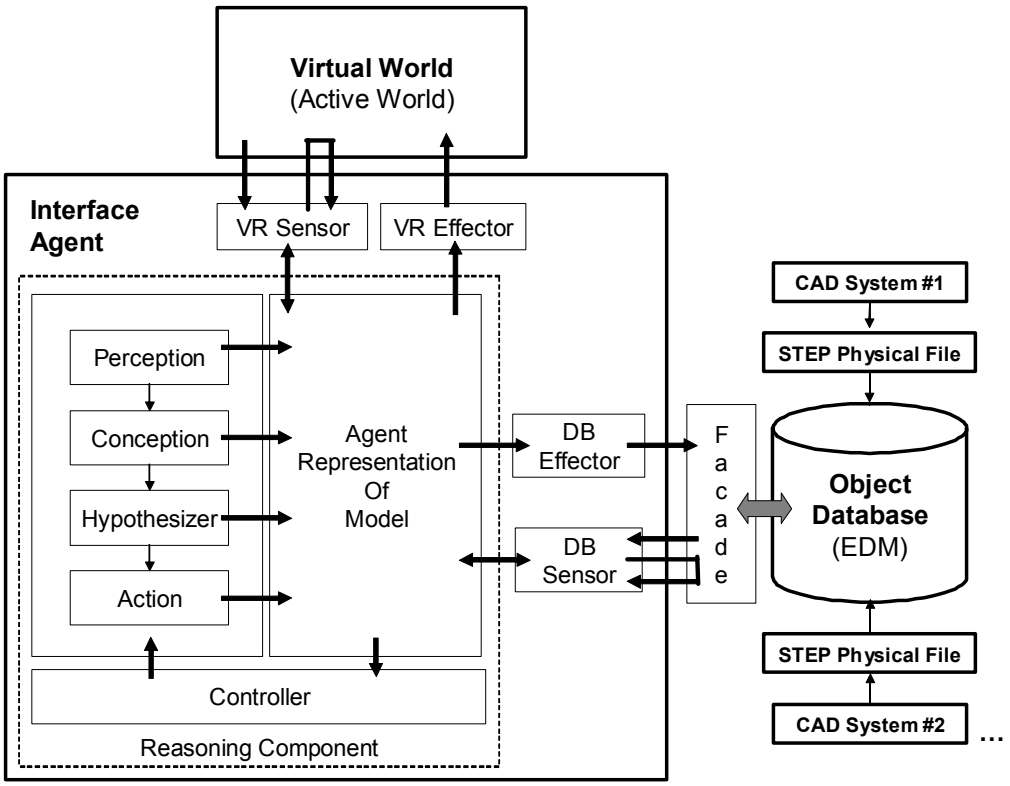

Figure 5. Framework for an agent-centric approach to data sharing

The database sensor (DB Sensor) has two functions: it senses the database for any modifications happening within it and it looks out for specific data within the database as directed by the model of reasoning. As new data is added or old data is modified, the database will trigger the sensor by pushing data into the agent. This data is structured by perception into a form suitable for reasoning

within system. When the sensor is triggered by the reasoning model to look for specific data within the database, it pulls the required data from the database.

The virtual world sensor (VR Sensor) also has two functions: it senses the virtual world for any modifications happening within it and it looks out for specific data within the world as directed by the model of reasoning. As new entities are added or old entities are modified, the virtual world will trigger the sensor by pushing data into the agent. This data is structured by perception into a form suitable for reasoning within the system. When the sensor is triggered by the reasoning component, it looks for specific data within the world and pulls the data into the agent for processing.

The reasoning component deals with the consistency of the data within a specific application and data shared across different applications. In the case of the virtual environment, any modifications and creation of relevant data in the virtual world or within the database are propagated to each other.

OPERATION OF THE INTERFACE AGENT

To illustrate the agent framework for data sharing, consider the behaviour of the interface agent (a "walls" agent) that keeps tracks of walls of a specific type in the virtual world. This walls agent creates a "wall" agent for each wall in the EDM when the system is first initiated. A wall agent is also created by this interface (walls) agent when a new wall is created in the virtual world. Each wall agent created in the virtual world has a reflex behaviour of notifying its corresponding interface (walls) agent so that the EDM database can be updated with the new information regarding the new wall.

Since the virtual world is intended as the collaborative modelling environment, the (wall) agent monitors conversation and object modifications in the virtual world. This agent is able to reason about this sense data to make modifications to the object in the virtual world and its corresponding object in the EDM. This (wall) agent can able reason about the modifications and interact with the users if suggested modifications violate constraints or are inconsistent with other modifications. In general, the reasoning processes for the wall agent are:

- Sensation: Sense data about the initial wall data from the EDM is placed in the interface (walls) agent's memory and changes to the wall in the virtual world are put into the wall agent's memory. Sense data continues to flow into the wall agent's memory about the conversations of the users in the virtual world, about the location and changes in objects near the wall, and about direct changes to the wall from the users in the virtual world.
- Perception: The sense data is organized in the interface (walls) agent's memory by the perception process, looking for patterns such as the shape of the wall, the aspect ratios of the wall, the people in the virtual world that are near or interacting with the wall.
- Conception: The concepts that the wall agent can know about include: connections to other objects in the world such as other walls, or doors or windows, etc; and people that are responsible for the design of the wall vs. strangers that can not make permanent changes to the wall.
- Hypothesiser: The hypothesiser reasons about the goals of the interface (walls) agent. One goal is to maintain a consistent representation among the EDM and virtual world. Other goals related to the wall agent include responding to queries or changes initiated in the conversation of the users of the virtual world; satisfying constraints on connections between the wall and other objects in the world; and making sure that the functions of the wall are achieved by the current data associated with the wall.
- Action: This reasoning model knows about the effectors for changing and adding data to the EDM and the virtual world. These changes are directed in reflex rules when specific sense data is pushed into the wall or interface (walls) agents' memory, or directed by recognized patterns for reactive behaviours, and by the identification of goals for reflective behaviour. The action rules take three forms:
    - Reflex: If sense data then effect changes in EDM or virtual world
    - Reactive: If a pattern is recognized then effect changes in EDM or virtual world
    - Reflective: If goal is active then effect changes in EDM or virtual world

The following reflexive behaviours are being implemented for a wall agent:

- If position of wall is changed in the virtual world then get change parameters and update the EDM database;

- If orientation of wall is changed in the virtual world then get change parameters and update EDM database;
- If wall is removed from the virtual world then delete wall from the EDM database;
- If wall is cloned in the virtual world then get information of new wall and create a new wall agent.

The agent approach allows a generalised reasoning component for a type of object to be defined, but then allows the agent's memory of the use and operations on that object to be reasoned about so that each object is unique and maintains its own history of changes and reasoning. Although the implementation of the agent reasoning component has only progressed to the reflexive behaviours, the framework for the collaborative visualization and the agent models for objects in a building model has been established.

CONCLUSIONS AND FUTURE WORK

The agent approach to data sharing described in this paper facilitates synchronous collaboration and real time multi-user multi-disciplinary modification of shared data through a coherent framework that integrates the functionalities of different sensors, effectors and reasoning components. This framework serves as a foundation upon which subsequent reasoning components are added for more elaborative reasoning. Future work includes the development of the reasoning component to allow behaviours that are beyond the reflexive mode described here. Constraints contained within the EDM database will drive the reactive behaviour by indicating any invalid data modifications. The hypothesizer will propose the appropriate actions on top of indicating invalid modifications when the agent operates in a reflective mode. For example, a user may change the material of a particular wall in the virtual environment and this change is updated into the central database. Before the data is changed, the constraints that are applicable are applied to see if any violations are created by the alterations. Any violations are flagged as an error and the user is notified when the agent reacts reactively. Additional suggestions are given when the agent reasons about other possibilities that may be applied in the current situation when the reflective component is added.

ACKNOWLEDGEMENTS

This research is funded by the Collaborative Research Centre for Construction Innovation (CRC-CI) in Australia. The authors acknowledge Lan Ding and Robin Drogemuller for their contributions to the project and Greg Smith for his implementation of the agent model using the Active Worlds SDK[1] and a Java Native Interface, and Jess[2] as the production system language for the agent rules.

REFERENCES

EPM Technology. (2002). *EDMAssist4*.5, vols 1-5. Norway: EPM Technology.

Gamma, E., Helm, R., Johnson, R. and Vlissides, J. (1995). *Design patterns*, Addison-Wesley, MA.

Gero, J.S. (1998). Conceptual designing as a sequence of situated acts, in Smith I. (ed.), *Artificial Intelligence in Structural Engineering*, Springer, Berlin, pp. 165-177.

Gero, J.S. and Kannengiesser, U. (2002). The situated Function-Behavior-Structure framework, in Gero J.S. (ed.), *Artificial Intelligence in Design'02*, Kluwer, Dordrecht, pp. 89-104.

Huhns, M.N. and Stephen, L. (1999). *Multiagent* Systems and Society of Agents, in Weiss, G. (ed.), *Multiagent Systems, A Modern Approach to Distributed Artificial Intelligence*, MIT Press, MA, pp.79-120.

IAI. (2000). *IFC Specifications*, www.iai-international.org.

Maher, M.L. and Gero, J.S. (2002). Agent models of virtual worlds. *ACADIA 2002: Thresholds*, CA: California State Polytechnic University, Pomona, pp. 127-138.

Saunders, R and Gero, J.S. (2001). A curious design agent, in Gero, J.S., Chase, S. and Rosenman, M. (eds), *CAADRIA'01*, Key Centre of Design Computing and Cognition, University of Sydney, pp. 345-350.

[1] http://www.activeworlds.com/sdk
[2] http://herzberg.ca.sandia.gov/jess/

Shalloway, A. and Trott, J.R. (2001). *Design patterns explained: A new perspective on object-oriented design*, Addison-Wesley.

Smith, G. and Gero, J.S. (2001). Situated design interpretation using a configuration of actor capabilities, in Gero, J.S., Chase, S. and Rosenman, M. (eds), *CAADRIA'01*, Key Centre of Design Computing and Cognition, University of Sydney, 2001, pp. 15-24.

Wooldridge, M. (1999). Intelligent agents, in Weiss, G. (ed.), *Multiagent Systems, A Modern Approach to Distributed Artificial Intelligence*, MIT Press, MA, pp.27-78.

Wooldridge, M. and Jennings, N.R. (1995). Intelligent agents: Theory and practice, in *Knowledge Engineering Review* **10**(2): 115–152.