A Hybrid Visual Environment for Models and Objects

Paul A. Fishwick

Department of Computer & Information Science & Engineering
University of Florida
Gainesville, Florida 32611, U.S.A.

July 13, 1999

ABSTRACT

Models and objects that are modeled are usually kept in different places when we consider most modern simulation software packages. Software that permits the user to view 3D objects may also permit a viewing of the dynamic models for the objects, but these views are usually separate. The object can be rotated, translated and navigated while the model is represented in a 2D fashion using text or 2D iconic graphics. We present an approached based on the Virtual Reality Modeling Language (VRML), where the object and model reside in the same space. A browsing capability is built to allow the user to search for models "within" objects. Aside from the visual benefits derived from this integrated approach, this methodology also suggests that models are really not very different from objects. Any object can serve to model another object and when these objects are made "web friendly," it becomes feasible to use VRML to create distributed models whose components can reside anywhere over the web.

1 THE NATURE OF MODELING

One physical object captures some information about another object. If we think about our plastic toys, metal trains and even our sophisticated scale-based engineering models, we see a common thread: to build one object that says something about another—usually larger and more expensive—object. Let's call these objects the *source object* and the *target object*. Similar object definitions can be found in the literature of metaphors (Lakoff 1987) and semiotics (Noth 1990). The source object *models* the target, and so, modeling represents a relation between objects. Often, the source object is termed *the model* of the target. We have been discussing scale models identified by the source and target having roughly proportional geometries. Scale-based models often suffer from the problem where changing the scale of a thing affects more than just the geometry. It also affects the fundamental laws applied at each scale. For example, the hydrodynamics of the scaled ocean model may be different than for the real ocean. Nevertheless, we can attempt to adjust for the scaling problems and proceed to understand the larger universe through a smaller, more manipulable, version.

Later on in our education, we learned that modeling has other many other forms. The mathematical model represents variables and symbols that describe or model an object. Learning may begin with algebraic equations such as $d = \frac{1}{2}at^2 + v_o t + d_0$ where d, v and a represent distance, velocity and acceleration, and where d_0 and v_0 represent initial conditions (i.e., at time zero) for starting distance and initial velocity. These models are shown to be more elegantly derived from Newton's laws, yielding ordinary differential equations of the form $f = ma$. How do these mathematical, equational models relate to the ones we first learned as children?

To answer this question, let's first consider what is being modeled. The equations capture attributes of an object that is undergoing change in space (i.e., distance), velocity and acceleration. However, none of the geometrical proportions of the target are captured in the source since the structure of the equations is invariant to the physical changes in the target. A ball can change shape during impact with the ground, but the equations do not change their *shape*. If a ball represents the target, where is the source? The source is the medium in which the equations are presented. This may, at first, seem odd but it really is no different than the toy train model versus the actual train. The paper, phosphor or blackboard—along with the medium for the drawing, excitation or marking—has to exist if the equations are to exist. In a Platonic sense, we might like to think of the equations as existing in a separate, virtual, non-physical space. While one can argue their virtual existence, this representation-less and non-physical form is impractical. Without a physical represen-

tation, the equation cannot be communicated from one human to another. The fundamental purpose of representation and modeling is communication. Verbal representations (differential air pressure) are as physical as those involving printing or the exciting of a phosphor via an electron beam.

2 RUBE

Since 1989 at the University of Florida, we have constructed a number of modeling and simulation packages documented in (Fishwick 1992; Cubert and Fishwick 1998) In late 1998, we started designing Rube, named in dedication to Rube Goldberg (Marzio 1973), who produced many fanciful cartoon machines, all of which can be considered *models* of behavior. The procedure for creating models is defined as follows. *Step 1*: The user begins with an object that is to be modeled. For JPL, this can be the Cassini spacecraft with all of its main systems: propulsion, guidance, science instrumentation, power, and telecommunication. If the object is part of a larger scenario, this scenario can be defined as the top-most root object; *Step 2*: *scene* and interactions are sketched in a story board fashion, as if creating a movie or animation. A scene is where all objects, including those modeling others, are defined within the VRML file. VRML stands for Virtual Reality Modeling Language (Carey and Bell 1997), which represents the standard 3D language for the web. The Rube model browser is made available so that users can "fly though" an object to view its models without necessarily cluttering the scene with all objects. However, having some subset of the total set of models surfaced within a scene is also convenient for aesthetic reasons. The modeler may choose to build several scenes with models surfaced, or choose to view objects only through the model browser that hides all models as fields of VRML object nodes; *Step 3*: The shape and structure of all Cassini components are modeled in any modeling package that has an export facility to VRML. Most packages, such as Kinetix 3DStudioMax and Autodesk AutoCAD have this capability. Moreover, packages such as CosmoWorlds and VRCreator can be used to directly create and debug VRML content; *Step 4*: VRML PROTO (i.e., prototype) nodes are created for each object and component. This step allows one to create *semantic attachments* so that we can define one object to be a behavioral model of another (using a *behavior* field) or to say that the Titan probe is part of the spacecraft (using a *contains* field), but a sibling of the orbiter. Without prototypes, the VRML file structure lacks semantic relations and one relies on simple grouping nodes, which are not sufficient for clearly defining how objects relate to one another; *Step 5*: Models are created for Cassini. While multiple types of models exist, we have focused on dynamic models of components, and the expression of these components in 3D. Even textually-based models that must be visualized as mathematical expressions can be expressed using the VRML text node. Models are objects in the scene that are no different structurally from pieces of Cassini—they have shape and structure. The only difference is that when an object is "modeling" another, one interprets the object's structure in a particular way, using a dynamic model template for guidance; *Step 6*: Several dynamic model templates exist. For Newell's Teapot (in Sec. 3), we used three: FBM, FSM, EQN. These acronyms are defined as follows: FSM = Finite State Machine; FBM = Functional Block Model; EQN = Equation Set. Equations can be algebraic, ordinary differential, or partial differential; *Step 7*: The creative modeling act is to choose a dynamic model template for some behavior for Cassini and then to pick objects that will convey the meaning of the template within the scenario. This part is a highly artistic enterprise since literally any object can be used. In VRML, one instantiates an object as a *model* by defining it: `DEF Parthenon-Complex FSM {...}`. In other words, a collection of Parthenon-type rooms are interconnected in such a way that each Parthenon-Room maps to a state of the FSM. Portals from one room to another become transitions, and state-to-state transitions become avatar movements navigating the complex; *Step 8*: There are three distinct types of roles played modelers in Rube. At the lowest level, there is the person creating the *model templates* (FSM,FBM,EQN,PETRI-NET). Each dynamic model template reflects an underlying system-theoretic model (Fishwick 1995). At the mid-level, the person uses an existing model template to create a *metaphor*. A Parthenon-Complex as described before is an example of an architectural metaphor. At the highest level, a person is given a set of metaphors and can choose objects from the web to create a model. These levels allow modelers to work at the levels where they are comfortable. Reusability is created since one focuses on the level of interest; *Step 9*: The simulation proceeds by the modeler creating threads of control that pass events from one VRML node to another. This can be done in one of two ways: 1) using VRML Routes, or 2) using exposed fields that are accessed from other nodes. Method 1 is familiar to VRML authors and also has the advantage that routes that extend from one model component to an adjacent component (i.e., from one state to another or from one function to another) have a topolog-

ical counterpart to the way we visualize information and control flow. The route defines the topology and data flow semantics for the simulation. Method 2 is similar to what we find in traditional object-oriented programming languages where information from one object is made available to another through an assignment statement that references outside objects and classes. In method 1, a thread that begins at the root node proceeds downward through each object that is role-playing the behavior of another. The routing thread activates Java or Javascript Script nodes that are embedded in the structures that act as models or model components for the behaviors; *Step 10*: Pre- and Post-processing is performed on the VRML file to check it for proper syntax and to aid the modeler. Pre-processing tools include wrappers (that create a single VRML file from several), decimators (that reduce the polygon count in a VRML file), and VRML parsers. The model browser mentioned earlier is a post-production tool, allowing the user to browse all physical objects to locate objects that model them. In the near future, we will extend the parser used by the browser to help semi-automate the building of script nodes.

Rube treats all models in the same way. For a clarification of this remark, consider the traditional use of the word "Modeling" as used in everyday terms. A model is something that contains attributes of a target object, which it is modeling. Whereas, equation and 2D graph-based models could be viewed as being fundamentally different from a commonsense model, Rube views them in exactly the same context: everything is an object with physical extent and modeling is a relation among objects. This unification is theoretically pleasing since it unifies what it means to "model" regardless of model type.

3 NEWELL'S TEAPOT

In the early days of computer graphics (c. 1974-75), Martin Newell rendered a unique set of Bézier surface spline patches for an ordinary teapot, which currently resides in the Computer Museum in Boston. The teapot was modeled by Jim Blinn and then rendered by Martin Newell and Ed Catmull at the University of Utah in 1974. While at this late date, the teapot may seem quaint, it has been used over the years as an icon of sorts, and more importantly as a benchmark for all variety of new techniques in rendering and modeling in computer graphics. The Teapot was recently an official emblem of the 25th anniversary of the ACM Special Interest Interest Group on Computer Graphics (SIGGRAPH).

One of our goals for Rube was to recognize that the Teapot could be used to generate another potential benchmark—one that captured the entire teapot, its contents and its models. The default teapot has no behavior and has no contents; it is an elegant piece of geometry but it requires more if we are to construct a fully *digital teapot* that captures a more complete set of knowledge. In its current state, the teapot is analogous to a building façade on a Hollywood film studio backlot; it has the shape but the whole entity is missing. In VRML, using the methodology previously defined, we built TeaWorld in Fig. 1. We have added extra props so that the teapot can be visualized, along with its behavioral model, in a reasonable contextual setting. The world is rendered in Fig. 1 using a web browser. *World* is the top-most root of the scene graph. It contains a *Clock*, *Boiling_System*, and other objects such as the desk, chairs, floor and walls. The key fields in Fig. 2 are VRML nodes of the relevant field so that the *contains* field is refers to multiple nodes for its value. This is accomplished using the VRML *MFNode* type. The hierarchical VRML scene graph for Fig. 1 is illustrated in Fig. 2. The

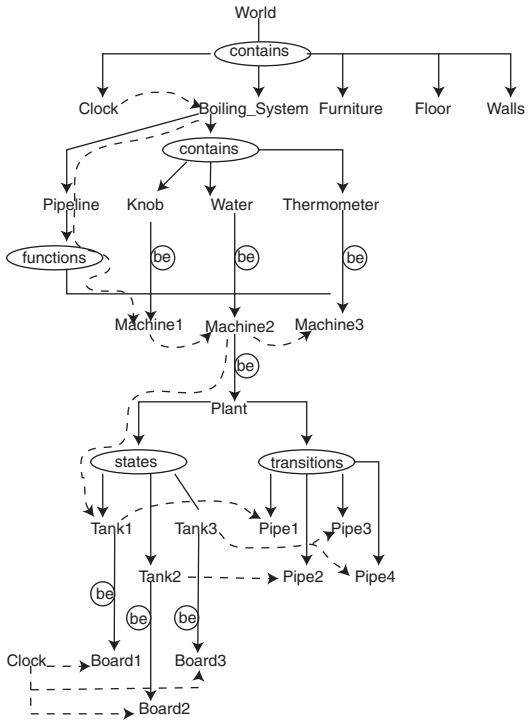

Figure 2: VRML Scene Graph for the Teapot and its models.

scene contains walls, a desk, chair and a floor for context. On the desk to the left is the teapot which is filled with water. The knob controlling whether the teapot heating element (not modeled) is on or

Figure 1: Office scene with Newell Teapot, dynamic model and props.

off is located in front of the teapot. To the right of the teapot, there is a pipeline with three machines, each of which appears in Fig. 1 as a semi-transparent cube. Each of these machines reflects the functional behavior of its encapsulating object: *Machine1* for *Knob*, *Machine2* for *Water* and *Machine3* for *Thermometer*. The *Thermometer* is a digital one that is positioned in *Machine3*, and is initialized to an arbitrary ambient temperature of 0° C. Inside *Machine2*, we find a more detailed description of the behavior of the water as it changes its temperature as a result of the knob turning. The plant inside *Machine2* consists of *Tank1*, *Tank2*, *Tank3*, and four pipes that move information from one tank to the next. Inside of each tank, we find a blackboard on which is drawn a differential equation that defines the change in water temperature for that particular state. The following modeling relationships are used: *Pipeline* is a Functional Block Model (FBM), with three functions (i.e., machines); *Machine* is a function (i.e., semi-transparent cube) within an FBM; *Plant* is a Finite State Machine (FSM) inside of Machine 2; *Tank* is a state within a FSM, and represented by a red sphere; *Pipe* is a transition within a FSM, and represented by a green pipe with a conical point denoting direction of control flow; and *Board* is a differential equation, represented as white text. The following metaphors are defined in this example. The three cubes represent a sequence of machines that create a pipeline. One could have easily chosen a factory floor sequence of numerically controlled machines from the web and then used this in TeaWorld to capture the information flow. Inside the second machine, we find a plant, not unlike a petroleum plant with tanks and pipes.

The *Pipeline* and its components represent physical objects that can be acquired from the web. For our example, we show simple objects but they have been given meaningful real-world application-oriented names to enforce the view that one object models another and that we can use the web for searching and using objects for radically different purposes than their proposed original function. The overriding concern with this exercise is to permit the modeler the freedom to choose *any* object to model *any* behavior. The challenge is to choose a set of objects that provide metaphors that are meaningful to the modeler. In many cases, it is essential that more than one individual understand the metaphorical mappings and so consensus must be reached during the process. Such consensus occurs routinely in science and in modeling when new modeling paradigms evolve. The purpose of Rube is not to dictate one model type over another, but to allow the modelers freedom in creating their own model types. In this sense, Rube can be considered a meta-level modeling methodology.

The simulation of the VRML scene shown in Fig. 2 proceeds using the dashed line thread that begins with the *Clock*. The clock has an internal time sensor that controls the VRML time. The thread corresponds closely with the routing structure built for this model. It starts at *Clock* and proceeds downward through all behavioral models. Within each behav-

ioral model, routes exist to match the topology of the model. Therefore, *Machine1* sends information to *Machine2*, which accesses a lower level of abstraction and sends its output to *Machine3*, completing the semantics for the FBM. The FSM level contains routes from each state to its outgoing transitions.

Fig. 3 shows a closeup view of the pipeline, that represents the dynamics of the water, beginning with the effect of the turning of the knob and ending with the thermometer that reads the water temperature.

Figs. 4,5 and 6 show the pipeline during simulation when the knob is turned on and off at random times by the user. The default state is the cold state. When the knob is turned to the on position, the system moves into the heating state. When the knob is turned again back to an off position, the system moves into the cooling state and will stay there until the water reaches ambient room temperature at which time the system (through an internal state transition) returns to the cold state. Temperature change is indicated by the color of *Water* and *Machine3*, in addition to the reading on the *Thermometer* inside of *Machine3*. The material properties of *Machine1* change depending on the state of the knob. When turned off, *Machine1* is semi-transparent. When turned on, it turns opaque. Inside *Machine2*, the current state of the water is reflected by the level of intensity of each *Plant*. The current state has an increased intensity, resulting in a bright red sphere.

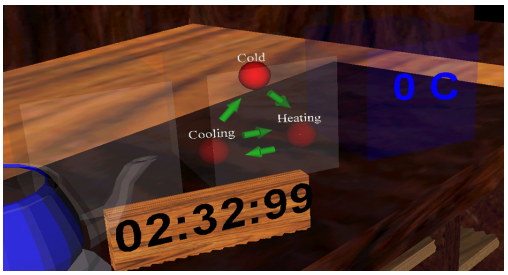

Figure 3: Pipeline closeup.

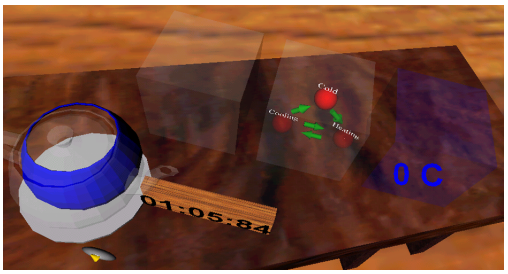

Figure 4: Cold state.

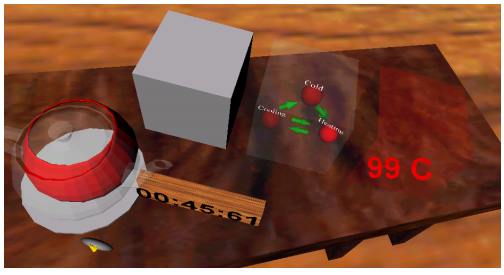

Figure 5: Heating state.

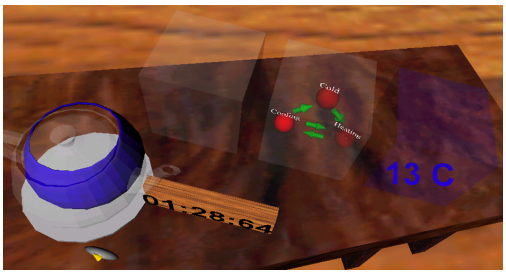

Figure 6: Cooling state.

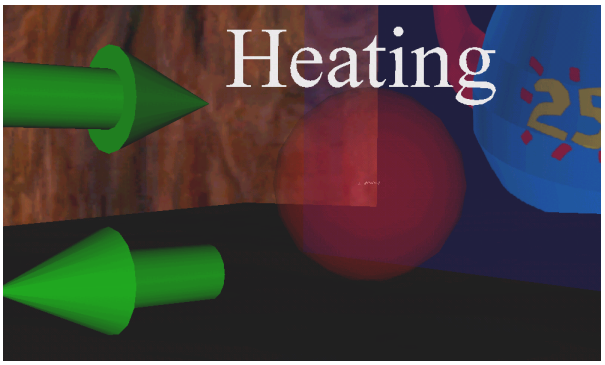

Figure 7: Outside the Heating phase.

The dynamics of temperature is indicated at two levels. At the highest level of the plant, we have a three state FSM. Within each state, we have a differential equation. The equation is based on Newton's Law of Cooling and results in a first order exponential decay and rise that responds to the control input from the knob. The visual display of temperature change confirms this underlying dynamics since the user finds the temperature changing ever more slowly when heating to 100°C or cooling back to the ambient temperature. Fig. 7 displays a closeup of the heating phase from the outside, and Fig. 8 is a view from inside the red sphere modeling the phase.

Given the Newell Teapot scene, there are some key issues which we should ask ourselves: *Is it a visualization*? The work in Rube provides visualiza-

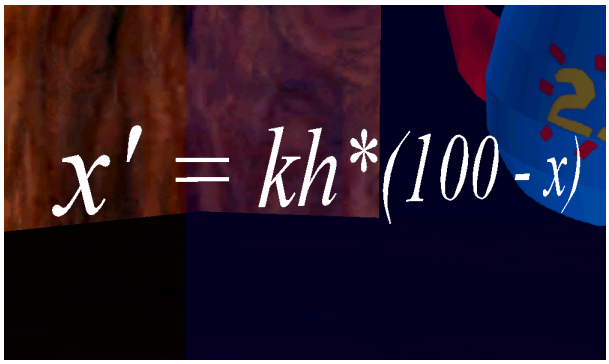

Figure 8: Inside the Heating phase.

tion, but models such as Cassini and Newell's Teapot demonstrate active modeling environments whose existence serves an engineering purpose and not only a post-project visualization purpose for outside visitors. This sort of modeling environment is needed from the very start of a mission—as an integral piece of the puzzle known as model design; *Is it economical?* Is this a lot of work just to create an FSM? Why go through the bother of creating the Parthenon, the complex and the avatar? All of these items are reused and so can be easily grabbed from the web. The concept of reuse is paramount to the Rube approach where the metaphor can be freely chosen and implemented. Without the web, Rube would not be possible. 3D object placement can be just as economical as 2D object placement, but object repositories are required not only for Cassini and Titan, but also for objects that serve to model the dynamic attributes of other objects (i.e., the Parthenon). Another economical aspect centers on the issue of computational speed for these models. Would creating a simulation in a more typical computer language would be more efficient? The structure of objects and their models within a VRML scene can be translated or compiled into native machine code as easily as source code; the 3D model structure becomes the "source code;" *What is the advantage?* If we consider psychological factors, the 3D metaphor has significant advantages. First, 3D spatially-specific areas serve to improve our memory of the models (i.e., mnemonics). Second, graphical user interfaces (GUIs) have shown that a human's interaction with the computer is dramatically improved when the right metaphors are made available. Rube provides the environment for building metaphors. One should always be wary of mixed metaphors. We leave the ultimate decision to the user group as to which metaphors are effective. A Darwinian-style of evolution will likely determine which metaphors are useful and which are not. Aesthetics plays an important role here as well. If a modeler uses aesthetically appealing models and metaphors, the modeler will enjoy the work. It is a misconception to imagine that only the general populous will benefit from fully interactive 3D models. The engineers and scientist need this sort of immersion as well so that they can understand better what they are doing, and so that collaboration is made possible; *Is this art or science?* The role of the Fine Arts in science needs strengthening. With fully immersive models, we find that we are in need of workers with hybrid engineering/art backgrounds. It is no longer sufficient to always think "in the abstract" about modeling. Effective modeling requires meaningful human interaction with 3D objects. So far, the thin veneer of a scale model has made its way into our engineering practices, but when the skin is peeled back, we find highly abstract codes and text. If the internals are to be made comprehensible (by anyone, most importantly the engineer), they must be surfaced into 3D using the powerful capabilities of metaphors (Lakoff and Johnson 1980; Lakoff 1987). This doesn't mean that we will not have a low level code-base. Two-dimensional metaphors and code constructs can be mixed within the 3D worlds, just as we find them in our everyday environments with the embedding of signs. At the University of Florida, we have started a *Digital Arts and Sciences* Program with the aim to produce engineers with a more integrated background. This background will help in the production of new workers with creative modeling backgrounds.

4 ART OF MODELING

It is sometimes difficult to differentiate models used for the creation of pieces of art from those used with scientific purposes in mind. Models used for science are predicated on the notion that the modeling relation is unambiguously specified and made openly available to other scientists. Modeling communities generally form and evolve while stressing their metaphors. In a very general sense, natural languages have a similar evolution. The purpose of art, on the other hand, is to permit some ambiguity with the hopes of causing the viewer or listener to reflect upon the modeled world. Some of the components in worlds such as Fig. 1 could be considered non-essential modeling elements that serve to confuse the scientist. However, these elements may contribute to a more pleasing immersive environment. Should they be removed or should we add additional elements to please the eye of the beholder? In Rube, we have the freedom to go in both directions, and it isn't clear which inclusions or eliminations are appropriate since it is entirely up to the modeler or a larger modeling

community. One can build an entirely two dimensional world on a blackboard using box and text objects, although this would not be in the spirit of creating immersive worlds that allow perusal of objects and their models.

It may be that a select number of modelers may find the TeaWorld room exciting and pleasing, and so is this pleasure counterproductive to the scientist or should the scientist be concerned only with the bare essentials necessary for unambiguous representation and communication? Visual models do not represent *syntactic sugar* (a term common in the Computer Science community). Instead, these models and their metaphors are essential for human understanding and comprehension. If this comprehension is complemented with a feeling of excitement about modeling, this can only be for the better. Taken to the extreme, a purely artistic piece may be one that is so couched in metaphor that the roles played by objects isn't clear. We can, therefore, imagine a kind of continuum from a completely unambiguous representation and one where the roles are not published. Between these two extremes, there is a lot of breathing space. Science can be seen as a form of consensual art where everyone tells each other what one object *means*. Agreement ensues within a community and then there is a mass convergence towards one metaphor in favor of another.

We are not proposing a modification to the VRML standard although we have found that poor authoring support currently exists in VRML editors for PROTO node creation and editing. We are suggesting a different and more more general mindset for VMRL—that it be used not only for representing the shape of objects, but all modeling information about objects. VRML should be about the complete digital object representation and not only the representation of geometry with low-level script behaviors to support animation. Fortunately, VRML contains an adequate number of features that makes this new mindset possible even though it may not be practiced on a wide scale. While a VRML file serves as the digital object, a model compiler is also required for the proper interpretation of VRML objects as models.

5 SUMMARY

There is no unified modeling methodology, nor should there be one. Instead, modelers should be free to use and construct their own worlds that have special meaning to an individual or group. With Rube, we hope to foster that creativity without limiting a user to one or more specific metaphors. Rube has a strong tie to the World Wide Web (WWW). The web has introduced a remarkable transformation in every area of business, industry, science and engineering. It offers a way of sharing and presenting multimedia information to a world-wide set of interactive participants. Therefore any technology tied to the web's development is likely to change modeling and simulation. The tremendous interest in Java for doing simulation has taken a firm hold within the simulation field. Apart from being a good programming language, its future is intrinsically bound to the coding and interaction within a browser. VRML, and its X3D successor, represent the future of 3D immersive environments on the web. We feel that by building a modeling environment in VRML and by couching this environment within standard VRML content, that we will create a "trojan horse" for simulation modeling that allows modelers to create, share and reuse VRML files.

Our modeling approach takes a substantial departure from existing approaches in that the modeling environment and the object environment are merged seamlessly into a single environment. There isn't a difference between a circle and a house, or a sphere and a teapot. Furthermore, objects can take on any role, liberating the modeler to choose whatever metaphor that can be agreed upon by a certain community. There is no single syntax or structure for modeling. Modeling is both an art and a science; the realization that all objects can play roles takes us back to childhood. We are building Rube in the hope that by making all objects virtual that we can return to free-form modeling of every kind. Modeling in 3D can be cumbersome and can take considerable patience due to the inherent user-interface problems when working in 3D using a 2D screen interface. A short term solution to this problem is to develop a model package that is geared specifically to using one or more metaphors, making the insertion of, say, the Parthenon complex rooms a drag and drop operation. Currently, a general purpose modeling package must be used carefully position all objects in their respective locations. A longer term solution can be found in the community of virtual interfaces. A good immersive interface will make 3D object positioning and connections a much easier task than it is today.

We will continue our research by adding to Rube and extending it to be robust. In particular, we plan on looking more closely into the problem of taking legacy code and making it available within the VRML model. This is probably best accomplished through TCP/IP and a network approach where the Java/Javascript communicates to the legacy code as a separate entity. We plan on extending the VRML

parser, currently used to create the model browser, so that it can parse a 3D scene and generate the Java required for the VRML file to execute its simulation. Presently, the user must create all Script nodes. The model browser will be extended to permit various modes of locating models within objects. A "fly through" mode will take a VRML file, with all object and model prototypes, and place the models physically inside each object that it references. This new generated VRML file is then browsed in the usual fashion. Multiple scenes can be automatically generated.

ACKNOWLEDGMENTS

I would like to thank the students on the Rube Project: Robert Cubert, Andrew Reddish, and John Hopkins. I would like to thank the following agencies that have contributed towards our study of modeling and simulation: (1) Jet Propulsion Laboratory under contract 961427 *An Assessment and Design Recommendation for Object-Oriented Physical System Modeling at JPL* (John Peterson, Stephen Wall and Bill McLaughlin); (2) Rome Laboratory, Griffiss Air Force Base under contract F30602-98-C-0269 *A Web-Based Model Repository for Reusing and Sharing Physical Object Components* (Al Sisti and Steve Farr); and (3) Department of the Interior under grant 14-45-0009-1544-154 *Modeling Approaches & Empirical Studies Supporting ATLSS for the Everglades* (Don DeAngelis and Ronnie Best). We are grateful for their continued financial support.

REFERENCES

Carey, R., and G. Bell. 1997. *The Annotated VRML 2.0 Reference Manual*. Addison-Wesley.

Cubert, R. M., and P. A. Fishwick. 1998. MOOSE: An Object-Oriented Multimodeling and Simulation Application Framework*Simulation* 70(6), 379–395.

Fishwick, P. A. 1992. Simpack: Getting Started with Simulation Programming in C and C++. In *1992 Winter Simulation Conference*, Arlington, VA, 154–162.

Fishwick, P. A. 1995. *Simulation Model Design and Execution: Building Digital Worlds*. Prentice Hall.

Lakoff, G. 1987. *Women, Fire and Dangerous Things: what categories reveal about the mind*. University of Chicago Press.

Lakoff, G., and M. Johnson. 1980. *Metaphors We Live By*. University of Chicago Press.

Marzio, P. C. 1973. *Rube Goldberg, His Life and Work*. New York: Harper and Row.

Noth, W. 1990. *Handbook of Semiotics*. Indiana University Press.

AUTHOR BIOGRAPHY

PAUL FISHWICK is Professor of Computer and Information Science and Engineering at the University of Florida. He received the PhD in Computer and Information Science from the University of Pennsylvania in 1986. His research interests are in computer simulation, modeling, and animation, and he is a Fellow of the Society for Computer Simulation (SCS). Dr. Fishwick will serve as General Chair for WSC00 in Orlando, Florida. He has authored one textbook, co-edited three books and published over 100 technical papers.