V-FIRE: Virtual Fire in Realistic Environments

**Frederick C. Harris, Michael A. Penick, Grant M. Kelly,
Juan C. Quiroz, Sergiu M. Dascalu, Brian T. Westphal**

Department of Computer Science and Engineering
University of Nevada, Reno
1664 N. Virginia St.
Reno, NV 89557, USA
{fredh, penick, gkelly, quiroz, dascalus, westphal}cse.unr.edu

Abstract *V-FIRE is a 3D fire simulation and visualization software tool that allows users to harness and observe fire evolution and fire-related processes in a controlled virtual environment. This paper presents details of the tool's requirements specification, software architecture, medium and low-level design, and prototype user interface. As the tool is currently in its development stage, a report on the project's current status is also included. In addition, a discussion of development challenges and pointers to future work are provided.*

Keywords virtual environment, fire simulation, requirements specification, software design, prototype user interface.

1 Introduction

Fire is a phenomenon that humans have studied for ages. Yet its dynamics and properties are not fully understood [1]. Fire modeling is a task that has not been achieved to great precision. Most importantly, the only way scientists have been able to study wildfires is by collecting data from actual fire disasters. Yet, this poses many problems. Firstly, a wildfire is a dangerous phenomenon. Due to the fact that fires create their own weather systems, being anywhere near a fire is unsafe. Furthermore, when a wildfire breaks out, it causes a large amount of damage, both to the environment and the community. Thus, the idea of having a fire burning freely just for observation purposes is preposterous. Consequently, the demand for tools which can aid in the studying of fires has increased due to technological advances, especially in computer graphics and modeling. The increasing number of firefighter casualties and the staggering costs of damages due to wildfires [2, 3] show the need for a tool and remedy for this malady.

As such, computer modeling of fires is an effective alternative for scientists to experiment and study the patterns of wildfires. Mathematical models of fires exist, but they do not correctly reflect the true behavior of fire. Fire is such a dynamic system, that a true model representation of it is hard to create. There are many factors that need to be taken into consideration. Most models rely on simplifications and assumptions in order to make the system solvable [1, 4, 5]. Furthermore, the results of such models are data, possibly graphs. The visualization of the model is even more difficult. The behavior of a wildfire can change drastically by a slight variation of the atmospheric pressure or of the speed or direction of the wind.

V-FIRE is a 3D fire simulation and visualization tool. V-FIRE is intended to allow users to harness and observe a fire within a controlled environment. The system has been designed to model a wildfire as realistic as possible with the use of marketable graphics, an efficient physics model, and a mathematically based spreading algorithm. In addition, users will also be able to visualize the interaction of fire with other objects such as smoke, vegetation, and buildings. Moreover, as an empirical tool, V-FIRE is also intended to provide the user with the ability of multiple view points for the main camera, such as aerial and full immersion [6].

The long term goal of V-FIRE is to create real-time, marketable-quality graphics for fire visualization in a Cave Automatic Virtual Environment (CAVE). A CAVE provides a full-immersion experience for its users [7]. Thus, the integration of V-FIRE with a CAVE would create a full 3D simulation in which users would be able to physically interact with a wildfire environment.

Wildfires are complex and dynamic phenomena. As such, the fire simulation with computer graphics poses a great challenge. A fire has a chaotic nature. Consequently, the modeling of fire with computer graphics must introduce a random factor into the algorithms used. Also, the transition from 2D graphics to 3D graphics is a challenging endeavor that needs to be researched further.

The remainder of this paper is structured as follows: Section 2 describes the functional and nonfunctional software requirements for the V-FIRE system, Section 3 presents the use case diagram and the use cases of V-FIRE, Section 4 describes the tool's high-level architecture, Section 5 provides details of its medium and low-level design, Section 6 reports on V-FIRE's current status and points to future work, and Section 7 contains several closing remarks.

2 Requirements Specification

V-FIRE is currently in its implementation stage. Its requirements specification and design phases have been completed, including creation of a low-fidelity user interface prototype [8]. Throughout the tools' development we have used the UML notation [9] to create software models and followed modeling techniques and guidelines from [10]. The functional and non-functional requirements for the V-FIRE system are provided in the next subsections. The requirements define the "reference points" for the remainder of the system's development.

2.1 Functional Requirements

Table I contains functional requirements with priority levels indicated in square brackets.

Table I. V-FIRE Functional Requirements

R01	[1]	V-FIRE shall display 3D fire in the simulation window.
R02	[1]	V-FIRE shall display 3D smoke in the simulation window.
R03	[1]	V-FIRE shall display 3D vegetation in the simulation window.
R04	[1]	V-FIRE shall display 3D buildings in the simulation window.
R05	[1]	V-FIRE shall display interaction between fire, smoke, vegetation, and buildings.
R06	[1]	V-FIRE shall allow the user to start the simulation.
R07	[1]	V-FIRE shall allow the user to stop the simulation at any time.
R08	[1]	V-FIRE shall allow the user to change the vegetation density of the terrain.
R09	[1]	V-FIRE shall allow the user to change the number of buildings on the terrain.
R10	[1]	V-FIRE shall allow the user to change the point of view.
R11	[1]	V-FIRE shall allow the user to view instructions on using the system.
R12	[2]	V-FIRE shall support various point of view presets, including but not limited to: three-quarter view, birds-eye, and ground-level.
R13	[2]	V-FIRE shall support a flying camera.
R14	[2]	V-FIRE shall allow the user to load terrain maps.
R15	[2]	V-FIRE shall allow the user to save terrain maps.
R16	[2]	V-FIRE shall allow the user to initiate a fire by clicking on the map.
R17	[2]	V-FIRE shall allow the user to pause the simulation.
R18	[2]	V-FIRE shall allow the user to fast-forward the simulation.
R19	[2]	V-FIRE shall allow the user to rewind the simulation.
R20	[3]	V-FIRE shall simulate a fire based on a set of input data.

At the beginning of the project an initial fully operational version was set to be completed by the end of April 2005. Given this deadline, level 1 priority requirements were defined as "will be met", level 2 priority requirements as "will most likely be met", and level 3 priority requirements as "will most likely not be met" (however, they will be considered for the next release of the system, planned for Fall 2005).

2.2 Non-Functional Requirements

Table II contains the main non-functional requirements of the V-FIRE software tool.

Table II. V-FIRE Functional Requirements

T01	V-FIRE shall render in real-time.
T02	V-FIRE shall be a cross-platform application.
T03	V-FIRE shall be implemented with Qt, OpenGL, and OpenSG.
T04	V-FIRE shall be multi-threaded environment compatible.
T05	V-FIRE shall have marketable-quality graphics.
T06	V-FIRE shall maintain a simple user interface.
T07	V-FIRE shall use particle based fire and smoke.
T08	V-FIRE shall use dynamic "combustible" models.

3 Use Case Modeling

To gain further insight into V-FIRE's functionality the system' behavior has been "broken up" into use cases. The diagram shown in Fig. 1 outlines the controls that allow the user to interact with the system, as well as the larger scale functionality of the backend. As shown in subsection 3.2, a direct mapping between the system's use cases and its functional requirements was also established.

3.1 Use Case Diagram

The use case diagram shown in Figure 1 outlines the system's functionality and the roles actors play in the V-FIRE system.

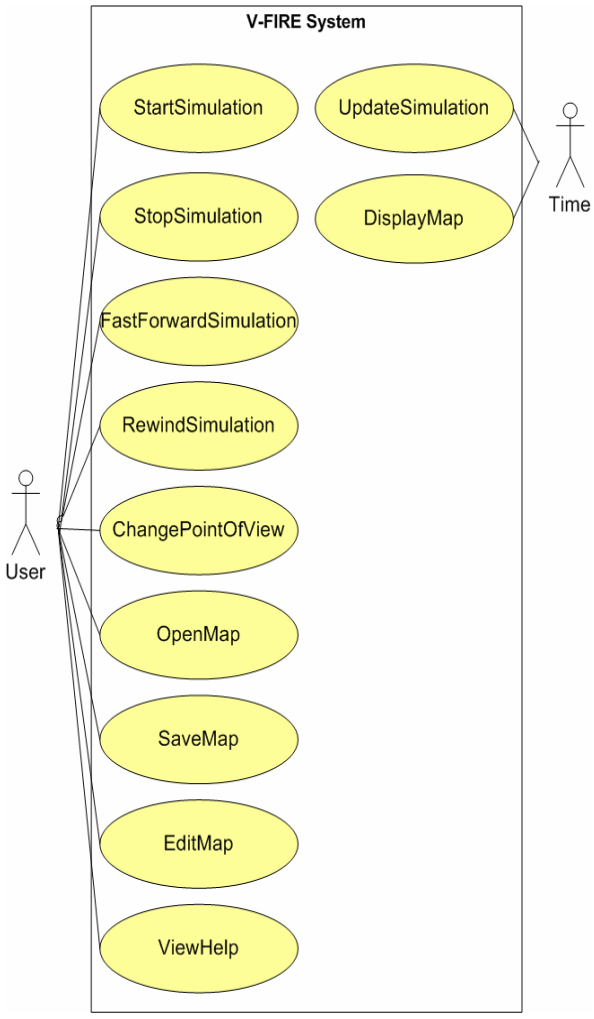

Figure 1 V-FIRE Use Case Diagram

In order to further clarify the functionality, detailed descriptions of each use case are presented next.

UC01 *StartSimulation* The user selects the start location of the fire. The user then pushes the start button to run the simulation at the specified start location.

UC02 *StopSimulation* The user stops a running simulation of the fire by pushing the stop button. The simulation is paused and can be restarted from the current time frame.

UC03 *FastFowardSimulation* The user fast forwards the simulation of the fire by pushing the fast forward button. The simulation can be returned to regular speed by pushing the play button. The user can also stop the simulation while fast forwarding.

UC04 *RewindSimulation* The user rewinds the simulation of the fire by pushing the rewind button. The simulation can be returned regular speed by pushing the play button. The user can also stop the simulation while rewinding.

UC05 *ChangePointOfView* The user can specify the point of view of the camera by selecting one of the presets from the "View" menu. The user has the option for a free flying camera controlled by the keyboard and mouse.

UC06 *OpenMap* The user can load a preexisting map by selecting the "Open Map" option from the "File" menu. The user selects the map to load by selecting from a file selection dialog.

UC07 *SaveMap* The user can save a loaded map by selecting the "SaveMap" option from the "File" menu. The user can also select to rename the map by selecting the "Save As ..." option from the "File" menu.

UC08 *EditMap* The user can edit the currently loaded map by selecting the "Edit Map" option from the "Edit" menu. The editing changes are saved by pushing the "OK" button.

UC09 *ViewHelp* The user can view instructions on how to use the system. The instructions can be accessed by selecting the "Tutorial" option from the "Help" menu.

UC10 *DisplayMap* A map is loaded by default when the user runs V-FIRE. A default number of trees and buildings are included on the default map.

UC11 *UpdateSimulation* V-FIRE displays to the user the interaction of fire, smoke, vegetation, and buildings. The texture of the buildings and vegetation are updated according to damage done by fire. Smoke plumes are displayed proportionately to the fire size.

3.2 Requirements Traceability Matrix

A partial (due to space limitations) requirements traceability matrix is shown in Figure 2, depicting the mapping between the use cases and the functional requirements of the V-FIRE system. The complete requirements traceability matrix can be found at [11].

		Use Case							
		UC01	UC02	UC03	UC04	UC05	UC06	UC07	UC08
Requirement	R06	X							
	R07		X						
	R08								X
	R09								X
	R10					X			
	R11								
	R12					X			
	R13					X			
	R14						X		
	R15							X	
	R16	X							
	R17		X						

Figure 2 Requirements Traceability Matrix for V-FIRE (partial)

4 Architectural Design

The system architecture is composed of several subsystems, as shown in Figure 3. Descriptions of subsystems are as follows:

GUI The *GUI subsystem* contains the classes that interface with Qt, OpenSG, and control user interaction with the system. This subsystem also connects the viewable front-end and the simulation back-end.

Simulation The *Simulation subsystem* contains the backend that controls the simulation. Through a generic interface, a programmer can control the simulation with little knowledge of the other subsystems' implementations. This subsystem is also responsible for the placement and overall density of models on the terrain.

Terrain The *Terrain subsystem* contains classes that describe the topographic features of the visible terrain and provide methods to load a terrain map from a file.

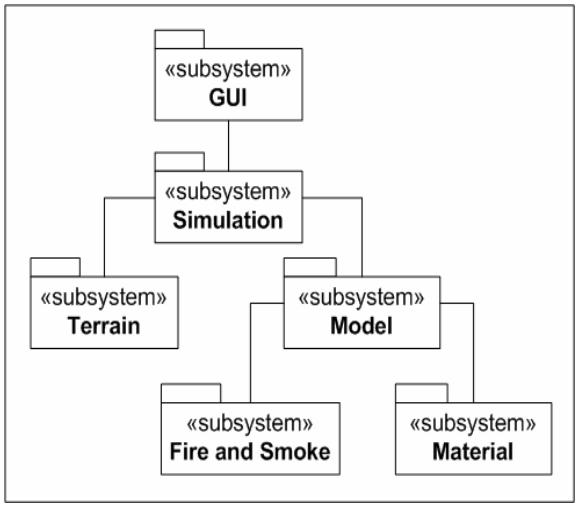

Figure 3 V-FIRE System Architecture

Model The *Model subsystem* contains classes that describe the visible state of 3D models used in the simulation. Such models include vegetation and inhabitable structures. This subsystem is responsible for the loading of models from files and maintaining a model's visible state throughout its life of combustion.

Fire and Smoke The *Fire and Smoke subsystem* contains classes that describe the visible fire and smoke used in the simulation. The state of this subsystem is controlled by logic in the simulation subsystem.

Material The *Material subsystem* contains classes that describe the properties and states of burnable materials in the simulation.

5 Detailed Design

The structure of V-FIRE was designed using an object-oriented approach. The organization of V-FIRE into program units and a sample activity chart are given, respectively, in subsections 5.1 and 5.2.

5.1 Class Diagram

A class diagram of V-FIRE, showing the modularization of the system into object classes is presented in Figure 5. The diagram also includes details of operations, attributes, relationships, multiplicity constraints, and visibility for each class. Complete class and method descriptions can be found at [11].

5.2 System Activity Chart

In order to thoroughly cover the design of V-FIRE, various diagrams were created as part of its software model, including activity charts, state charts, and flow charts. A sample activity chart of V-FIRE is presented in Figure 4.

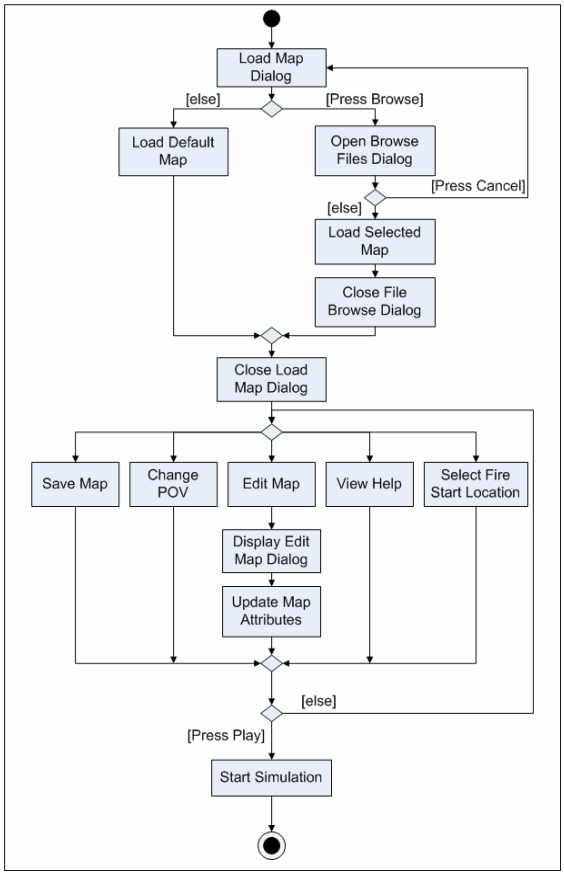

Figure 4 Activity Chart of V-FIRE (shows operations of the system before a fire simulation is started)

6 Current Status and Future Work

The specification, design, and initial implementation phases of the V-FIRE project have been completed. We have laid the groundwork for the visual components of the project, which are functional in the initial prototype. From the implementation work done so far, it became evident to us that the framework created is solid for driving the work ahead.

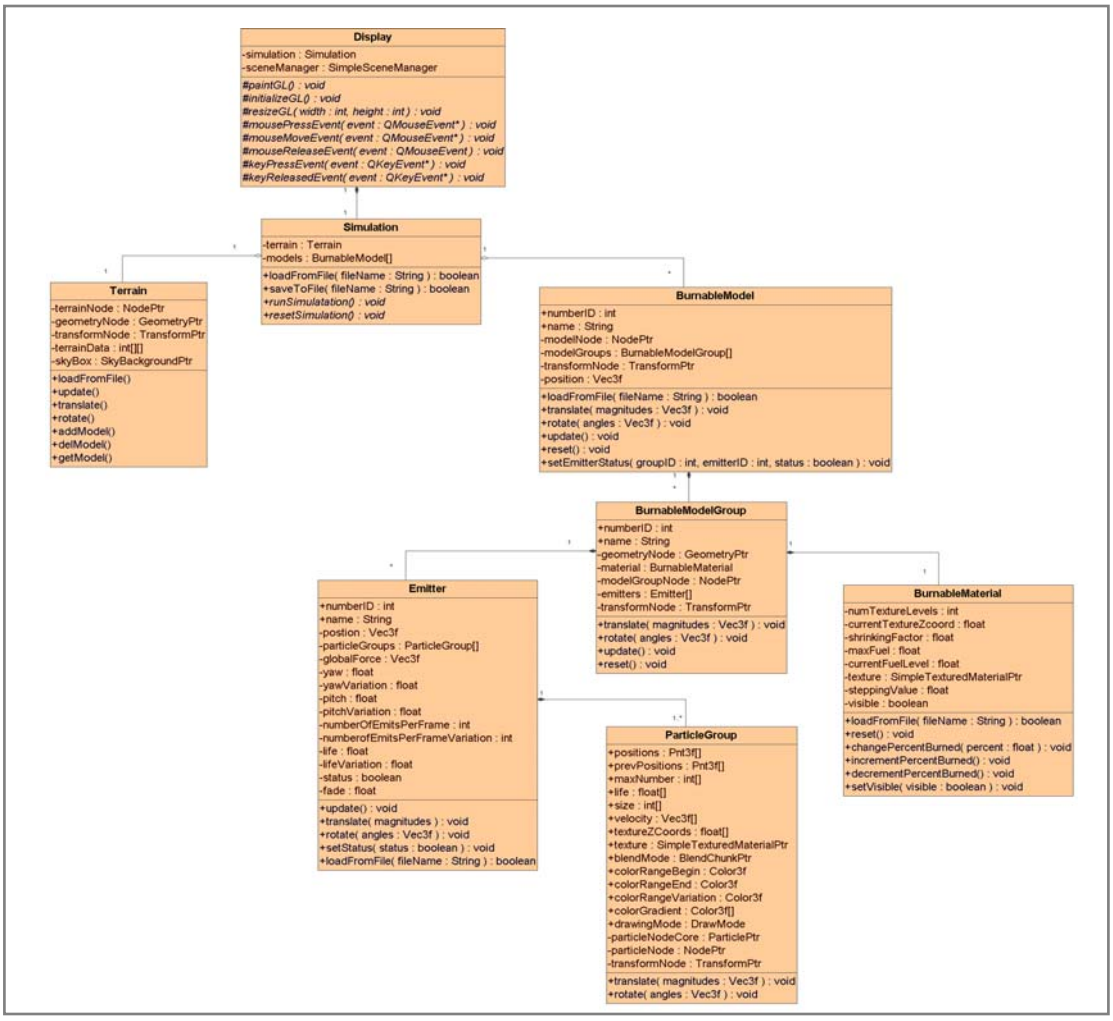

Figure 5. V-FIRE Class Diagram

There are a number of areas of work, however, that are outside the scope of the first phase of this project. V-FIRE is designed to support models and visualizations of fire. The models currently being used, such as those described in [2, 4, 5, 12] do not take all the complexities of fire-environment interaction into account. More advanced models will be created and implemented for future releases of V-FIRE. As processing capabilities increase, more detailed models will be able to run in real-time. Parallel processing techniques can also be used to increase the amount of complexity that can be handled.

As more advanced graphics hardware and rendering techniques and tools become available, they will be incorporated into the system. Tools such as programmable pixel shaders [13], which require the latest graphics hardware, can be used to create more realistic lighting and shadow effects generated by fire and smoke. New graphics engines, such as the Unreal 3 Engine [14], should also be considered as possible platforms for future work.

Lastly, as V-FIRE is used in research environments and integrated with other tools, it is likely that requests for different types of system interactions will be made. A comprehensive physics engine could be added to aid in creating realism within the environment. This would be especially important in applications such as firefighter or evacuation training software.

7 Conclusions

V-FIRE is an application designed to help researchers visualize models of fire in realistic environments. The system provides a safe context for learning how wildfires, accidental fires, and arson affect objects in the real world. Although this system will eventually be used for fully immersive 3D modeling and event re-creation, its current implementation with single monitor support provides the structure on top of which the software can continue to evolve. To offer some insight into the "look and feel" of the environment, Figure 6 shows a screenshot of the V-FIRE application's main window and Figure 7 presents a configuration editor for environment parameters.

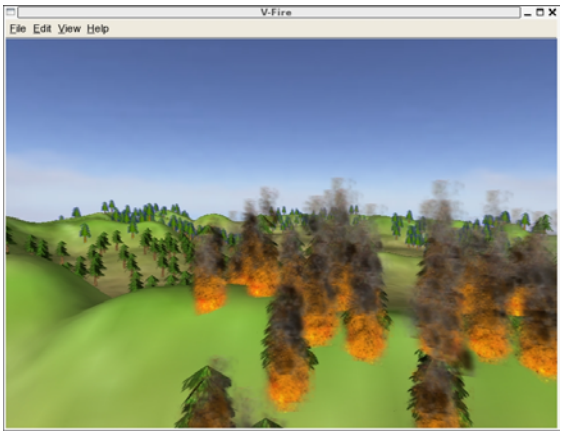

Figure 6 V-FIRE Main Interface

The specification and design processes undertaken by the project team have helped to create a flexible architecture that can be expanded without retooling the core elements of the system. Specifically, each subsystem is encapsulated to support more advanced fire modeling and visualizations in future work. Having a flexible and well-encapsulated architecture for this type of research is essential, as technical advances that can be applied to fire modeling and visualization are frequent.

References

[1] Drysdale, D., *An Introduction to Fire Dynamics,* Wiley & Sons, 2001.

[2] McCormick, P.S. & J.P. Anrens. Visualization of Wildfire Simulations. *IEEE Computer Graphics and Applications*, vol. 18, no. 2, 1998, pp. 17-19.

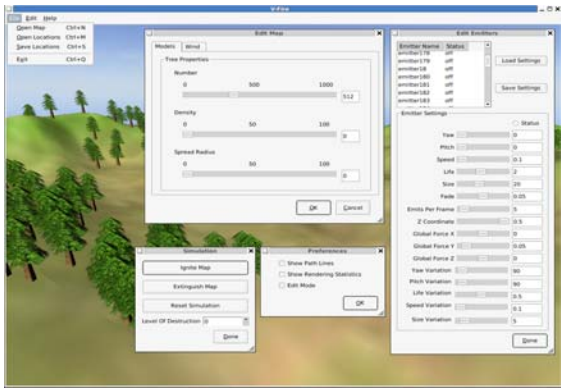

Figure 7 Environment editing and options

[3] Takeuchi, S. & S. Yamada. Monitoring of Forest Fire Damage by Using JERS-1 InSar, *Procs. 2002 IEEE Geosience and Remote Sensing Symposium*, vol. 6, pp. 3290-3292.

[4] Nguyen, D.Q., Fedwik, R., & H.W. Jansen. Physically Based Modeling and Animation of Fire. *Proceedings of the 29th ACM Intl. Conference on Computer Graphics and Interactive Techniques*, 2002, pp. 721-728.

[5] Wei, X., Li, W., Mueller, K. & A. Kaufman. Simulating Fire with Texture Splats. *Proceedings of the 2002 ACM Conference on Visualization,* pp. 227-235.

[6] Randima, F. *GPU Gems: Programming Techniques, Tips, and Trips for Real-Time Graphics.* Addison-Wesley, 2004.

[7] Creagh, H., CAVE Automatic Virtual Environments, *Proc. of the 2003 IEEE Conf. on Electrical Insulation and El. Manufacturing & Coil W. Technology*, pp. 499-504.

[8] Preece, J., Rogers, Y., & H. Sharp. *Interaction Design: Beyond Human-Computer Interaction,* Wiley & Sons, 2002.

[9] OMG's *UML Resource Page*, accessed March 19, 2005 at http://www.omg.org/uml

[10] Arlow, J. & I. Neustadt. *UML and the Unified Process: Practical Object-Oriented Analysis & Design*, Addison-Wesley, 2002.

[11] V-FIRE Project, accessed March 19, 2005 at http://www.cse.unr.edu/~gkelly/v-fire

[12] Ahrens, J., McCormick, P., Bossert, J., Reisner, J., & J. Winterkamp. Case Study: Wildfire Visualization. *Proceedings of IEEE Visualization'97*, pp. 451-454.

[13] nVIDIA website, *Pixel Shaders*, accessed March 18, 2005 at http://www.nvidia.com /object/ feature_pixelshader.html

[14] Epic Games, acc. March, 18, 2005 at www. epicgames.com/UnrealEngineNews.html