Flexible Application Design Using VR Juggler

Allen Bierbaum, Christopher Just, Patrick Hartling, Carolina Cruz-Neira
Virtual Reality Applications Center
Iowa State University
{allenb, cjust, patrick, carolina}@vrac.iastate.edu

Introduction

Designers of virtual reality (VR) applications need tools to effectively develop applications that meet the constraints of virtual environments. Many current tools have limited flexibility and require in-depth knowledge of the underlying VR system. We created VR Juggler, an open source *virtual platform* for VR application development, to provide a standard environment for application development, testing, and execution. VR Juggler's design allows for features such as dynamic reconfiguration, input abstraction, and performance monitoring [2]. This sketch describes these features and how they benefit VR developers.

Design Approach

VR Juggler utilizes an object-oriented software design [5] that uses standard interfaces to encapsulate VR components based on their functionality, thereby isolating applications from the underlying hardware details. For example, a tracking device uses a positional device interface to hide all hardware details. An application only cares about the type of data it sends and receives; the library handles all the details of accessing the actual hardware.

VR Juggler's component abstraction supports dynamic reconfiguration of applications at run time. Users are able to add, swap, or modify any component without affecting running applications. This reduces downtime because an application does not have to restart to make changes to a single component.

VR Juggler also provides a framework for application development. It supplies a well-defined interface that a developer realizes to create an application object. This interface assists development by providing a contract that guarantees when each part of the interface is executed and what the system's state will be at that time.

An added benefit of the application interface is that an application is merely another system component. An application object may be added, removed, or exchanged at run-time. VR Juggler also allows multiple applications to run concurrently.

VR Juggler scales across multiple platforms and VR systems. Developers can create applications in a simulator using the resources available at their desktops, without needing access to the actual VR system until the application is near completion.

Implementation

At VR Juggler's center is a portable microkernel[3], a minimal core that coordinates the entire system (Figure 1). The microkernel builds on an operating system abstraction layer that hides platform specific primitives such as threads and semaphores behind common interfaces to provide portability.

Managers supply most of VR Juggler's services; they are components that encapsulate common services and plug in to the microkernel. They can be added and removed at runtime. Examples are the Input Manager that regulates input devices and the Draw Managers that handles rendering tasks for specific graphics APIs.

Applications are components that the microkernel and Draw Manager execute using the standard application interface defined by VR Juggler.

Conclusion and Future Work

VR Juggler provides a flexible platform for VR application development. Developers benefit from its dynamic reconfiguration capabilities, extensibility, portability, input abstractions and componentization.

VR Juggler is an open source project and currently runs on many platforms. Current work includes using Bamboo[4] as a component framework, and extensions to support clusters. Check [1] for project status.

References

1 http://www.vrjuggler.org/

2 A. Bierbaum and C. Just ``Software Tools for Application Development," *ACM SIGGRAPH 98 Course #14: Applied Virtual Reality*. ACM SIGGRAPH 98 Conference, Orlando, July 1998. pp. 3.1-3.45

3 F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad and M. Stal ``Pattern-Oriented Software Architecture: A System of Patterns." 1996.

4 K. Watsen and M. Zyda "Bamboo - Supporting Dynamic Protocols For Virtual Environments," *Proceedings of the IMAGE 98 conference*. 1998

5 R. Stuart *The Design of Virtual Environments*. McGraw-Hill, 1996.

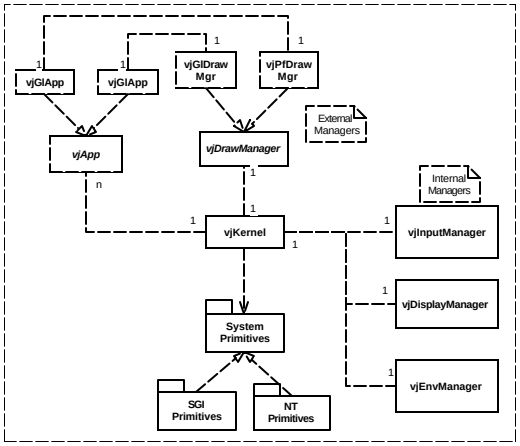

Figure 1: Architecture overview