Chapter 1.5
Construction of Collaborative Virtual Environments[1]

Anthony Steed
University College London, UK

Emmanuel Frécon
Swedish Institute of Computer Science, Sweden

ABSTRACT

In this chapter, we give an overview of some of the issues that face programmers and designers when building collaborative virtual environments (CVEs). We do this by highlighting three aspects of CVE system software: the environment model (data structures, behaviour description) that the system provides, the data-sharing mechanism (how the environment model is shared), and the implementation framework (the structure of a typical client or platform in terms of the services it provides to the user). When a CVE system is designed, choices have to be made for each of these aspects, and this then constrains how the designers and programmers go about constructing the CVE worlds themselves. We present the main body of the overview by using examples that highlight many of the important differences between CVE systems. We will also relate our discussion to the common topics of network topology and awareness management.

INTRODUCTION

A collaborative virtual environment (CVE) is a computer generated, three-dimensional space within which a geographically distributed set of users can interact in real-time. Many different types of CVEs can be found in use today, from online computer games to military simulations. The content and behaviour of different CVEs are very wide-ranging because of the different demands of the application. The primary requirement of an online computer game might be rapid response so that the game is fluid and enjoyable. The primary requirement of a military simulation might be verifiability and consistency so that the simulation can be studied for tactical purposes.

These different applications thus require very different systems to support the CVE and they use quite different types of programming model and description languages.

In this chapter we look at the structure of CVE systems and how the structure constrains and informs the role of the designers and programmers of CVEs. A CVE system is a software suite upon which a number of different CVEs can be built. A particular CVE system, such as a military simulation built around the DIS system (see p. 251), is best suited to support one particular class of CVE. Thus although games can, and have, been built on DIS, it is not the most common CVE system for such applications.

Building a CVE system presents many interesting challenges. A CVE system needs to present audio, video, and potentially haptic data to participants. A CVE system needs to support interactive and reactive capabilities so that the CVE can respond to the participant and present an interesting world. And whilst doing both input and output in real-time, the CVE system also needs to distribute all the data to multiple users at different sites so that they can collaborate.

The types of data involved in describing a CVE and the requirement to be both real-time and distributed means that CVE systems are a field of study in their own right, and they are somewhat distinct from other types of distributed system. For example, CVE systems typically generate high volumes of small data packets that need to be delivered at very low latency with a mixture of reliability requirements. Faced with potentially overwhelming amounts of data to maintain, CVE systems focus on only managing data concerning the local surroundings of each participant. In practice this means reducing the complexity of data within the system, and taking advantage of the limitation of display systems and human perceptual capabilities so that only important, perceivable data are generated.

In this chapter we will look at how typical CVE systems are built. We emphasise commonalities and differences between different CVE systems, and how these impact upon the role of the designers and programmers of the actual CVEs. We will describe the differences between CVE systems by focusing on three aspects of the design of a CVE system: the environment model for describing the CVE, the data-sharing mechanism to support distribution of CVEs described in that model, and the implementation framework of the CVE system. We introduce each of these aspects in turn in the following section. We then give an introduction to some of the main real-world issues to consider when developing a CVE system. The following three sections then discuss environment model, data-sharing mechanism, and implementation framework in more detail. Each of these sections will present two contrasting systems that illustrate some of the main issues. Each section also presents the authors' view of some of the main research challenges in that area. We then devote a section to discussion of scalability, the current challenge for CVEs, and finish by discussing some requirements and prospects for the next generation of CVE systems.

STRUCTURE OF A CVE SYSTEM

In this section we elaborate on what we mean by the three aspects of a CVE system: environment model, data-sharing mechanism, and implementation framework. These three aspects can be summarised from a programmer/designer's point of view as: What data structures and programs do I need to write to build my CVE? How does data get shared between the users and what do I need to do to maintain the CVE? What application services are there within the system and how do I use them? Identifying these three aspects will allow us to contrast various CVE systems and isolate conventions that are used, but which may not always be explicit when the systems are described.