research directions– *architectural patterns*

Facing the task of developing a multimedia information system, there are many options. Currently, the web seems to be the dominant infrastructure upon which to build a multimedia system. Now, assuming that we chose the web as our vehicle, how should we approach building such a system or, in other words, what architectural patterns can we deploy to build an actual multimedia information system? As you undoubtly know, the web is a document system that makes a clear distinction between *servers* that deliver documents and *clients* that display documents. See [OO], section 12.1. At the server-side you are free to do almost anything, as long as the document is delivered in the proper format. At the client-side, we have a generic document viewer that is suitable for HTML with images and sound. Dependent on the actual browser, a number of other formats may be allowed. However, in general, extensions with additional formats are realized by so-called *plugins* that are loaded by the browser to enable a particular format, such as *shockwave*, *flash* or *VRML*. Nowadays, there is an overwhelming number of formats including, apart from the formats mentioned, audio and video formats as well as a number of XML-based formats as for example SMIL and SVG. For each of these formats the user (client) has to download a plugin. An alternative to plugins (at the client-side) is provided by Java *applets*. For Java applets the user does not need to download any code, since the Java platform takes care of downloading the necessary classes. However, since applets may be of arbitrary complexity, downloading the classes needed by an application may take prohibitively long.

The actual situation at the client-side may be even more complex. In many cases a media format does not only require a plugin, but also an applet. The plugin and applet can communicate with eachother through a mechanism (introduced by Netscape under the name LiveConnect) which allows for exchanging messages using the built-in DOM (Document Object Model) of the browser. In addition, the plugin and applet may be controlled through Javascript (or VBscript). A little dazzling at first perhaps, but usually not to difficult to deal with in practice.

Despite the fact that the web provides a general infrasructure for both (multimedia) servers and clients, it might be worthwhile to explore other options, at the client-side as well as the server-side. In the following, we will look briefly at:

- the Java Media Framework, and
- the DLP+X3D platform

as examples of, respectively, a framework for creating dedicated multimedia applications at the client-side and a framework for developing intelligent multimedia systems, with client-side (rich media 3D) components as well as additional server-side (agent) components.

Java Media Framework The Java platform offers rich means to create (distributed) systems. Also included are powerful GUI libraries (in particular, Swing), 3D libraries (Java3D) and libraries that allow the use and manipulation of images,

audio and video (the Java Media Framework). Or, in the words of the SUN web site:

http://java.sun.com/products/java-media

> *The JavaTM Media APIs meet the increasing demand for multimedia in the enterprise by providing a unified, non-proprietary, platform-neutral solution. This set of APIs supports the integration of audio and video clips, animated presentations, 2D fonts, graphics, and images, as well as speech input/output and 3D models. By providing standard players and integrating these supporting technologies, the Java Media APIs enable developers to produce and distribute compelling, media-rich content.*

However, although Java was once introduced as the *dial tone of the Internet* (see [OO], section 6.3), due to security restrictions on applets it is not always possible to deploy media-rich applets, without taking recourse to the Java plugin to circumvent these restrictions.

DLP+X3D In our DLP+X3D platform, that is introduced in section **??** and described in more detail in appendix **??**, we adopted a different approach by assuming the availability of a generic X3D/VRML plugin with a Java-based External Authoring Interface (EAI). In addition, we deploy a high-level ditributed logic programming language (DLP) to control the content and behavior of the plugin. Moreover, DLP may also be used for creating dedicated (intelligent) servers to allow for multi-user applications.

The DLP language is Java-based and is loaded using an applet. (The DLP jar file is of medium size, about 800 K, and does not require the download of any additional code.) Dua, again, to the security restrictions on applets, additional DLP servers must reside on the site from where the applet was downloaded.

Our plugin, which is currently the *blaxxun* VRML plugin, allows for incorporating a fairly large number of rich media formats, including (real) audio and (real) video., thus allowing for an integrated presentation environment where rich media can be displayed in 3D space in a unified manner. A disadvantage of such a unified presentation format, however, is that additional authoring effort is required to realize the integration of the various formats.