TAP – The Architectural Playground

C++ framework for scalable distributed collaborative architectural virtual environments

SEICHTER Hartmut, DONATH Dirk, PETZOLD Frank

Bauhaus Universität Weimar, Chair Computer Science in Architecture
http://www.uni-weimar.de/iar

http://www.technotecture.net/projects/tap

Architecture is built information (Schmitt, 1999). Architects have the task of restructuring and translating information into buildable designs. The beginning of the design process where the briefing is transformed into an idea is a crucial phase in the design process. It is where the architect makes decisions which influence the rest of the design development process (Vries et al., 1998). It is at this stage where most information is unstructured but has to be integrated into a broad context. This is where TAP is positioned – to support the architect in finding solutions through the creation of spatially structured information sets without impairing thereby the creative development. We want to enrich the inspiration of an architect with a new kind of information design. A further aspect is workflow in a distributed process where the architect's work becomes one aspect of a decentralised working patterns. The software supports collaborative work with models, sketches and text messages within an uniform surface. The representations of the various media are connected and combined with each other and the user is free to combine them according to his or her needs.

***Keywords**: collaborative distributed virtual environment, API, cross-platform, C++, rapid architectural prototyping, sketching*

Preface

The idea for TAP was born out of experience with the existing VE–applications at the chair for Computer Science in Architecture Prof. Dr. D. Donath. Virtual Reality in the context of architecture has been dominated by 3D-simulations with walk-through and interaction-capabilities with the virtual model. The predecessors of TAP, most notably voxDesign, planeDesign and at least VRAM (Regenbrecht et al., 2000) were designed to act as real-time 3D-simulations. A variety of approaches to assisting the architect during the early design phase already exist (LaViola et al., 1998a, 1998b) however, it became clear during the course of the VeDS (Schnabel et al., 2001) that a series of new requirements are necessary for existing VE software. There is not only a need for appropriate interaction techniques within the VE, but also a need to communicate about those designs using a software tool (Garner, 2000; Do, 2000). Having to switch between a variety of different and incompatible communication tools hampers communication more than enabling communication, the problem becoming one more of conversion than of communication. The result is a loss of descriptive versatility: Design ideas requiring multiple media cannot be elaborated. A software is required which can provide the architect with many media forms under one consistent interface with which to describe his or her ideas. A further aspect is the need for a more generic software solution and less specifity, so that prototypes do not need to be rebuilt again from scratch when the application area changes. TAP is therefore designed to be an API and not only a specific application.

Goals

The main goal of TAP is to provide a fine-grain component-based software development kit to match the needs of the architectural design process. The results of a user survey undertaken in advance showed that it would be better to dispense with the wide variety of tools currently necessary in a typical design process and to replace it with a complete design environment which can be tailored to the specific needs of the user without needing special knowledge of programming or user-interface. TAP is therefore not only a desktop-VE with extended multimedia capabilities but also able to provide fully-immersive VE as well. TAP therefore represents a compromise between high integration and high adaptability of a software tool to the needs of the design process. The communication of architecture and the design process is not a matter of only one medium. It is the intelligent combination of different media of various categories that enriches the design process. As such architectural design can be seen as a specialised metastructure for media. It is this metastructure which TAP aims to describe and adapt using software.

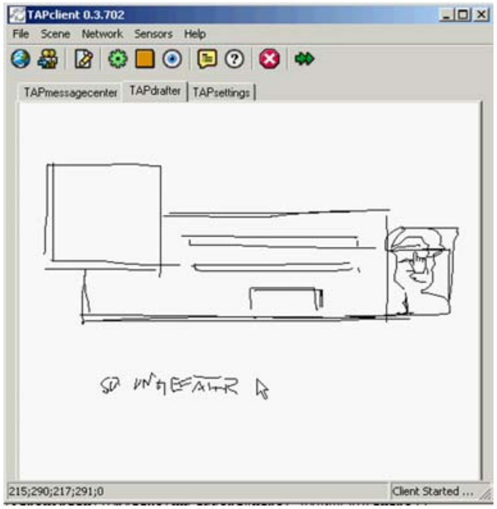
The software implementation concept

A primary aspect was the use of existing technologies and their adaptation to the requirements of architectural design. Because TAP uses a wide variety of third-party software, a main concern was interoperability and exchangeability which is made possible by wrapping all API's in an extra internal layer and then connecting the modules with one another at the level of this layer.

Architecture, the process of architectural design and communication in conjunction with architectural design can be expressed to a large degree in object-oriented structures, and for this reason a lot of modules are designed using UML tools which can easily be implemented in C++. Furthermore, most of the API's are written in C++ making the choice of implementation language clear. All third-party API's were encapsulated in internal abstraction layers to enable their recombination in a wide variety of prototypes based upon single source code modules.

The design of media for architectural needs

In an examination of today's tools it is clear that the transparency in the usability of different media within a single software environment is of vital importance. The architect as principal user of TAP wishes to integrate and interlink different media without becoming overwhelmed by technical aspects or different interfaces. Because TAP is a domain-specific framework, most of this can be automated. In TAP a medium is represented by a generalised object (i.e. non media-specific) which in turn is able to store specific information about the type, preferences and networking capabilities. If this object is to be output to a particular window, the internal rendering-engine decides how the information will be shown and the networking-engine decides which information will be provided by the server. The internal workings in


TAPclient 0.3.702
File Scene Network Sensors Help
TAPmessagecenter TAPdrafter TAPsettings
215;290;217;291;0
Client Started ...

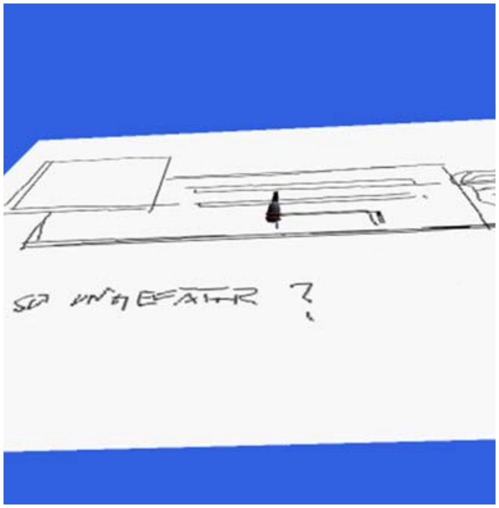


Figure 1. A draft on desktopside …

Figure 2.. … appears inside VE

the use of media objects becomes transparent to the user. Controls and presets can constrain the rendering capabilities with regard to user administration or media representation.

Scripting your own TAPapplication

The compilation time for the API has become an important factor during the development process of TAP. Even on high-performance systems the complexity and the high-level abstraction hierarchy is such that the compilation time has become an issue. To solve this, TAP introduces a scripting layer which makes it easier to verify conceptional approaches within the software without recompiling it. We have chosen to embed ECMAScript which is provided by the Mozilla project Spidermonkey (www.mozilla.org/js/spidermonkey: May 2002). ECMAScript/JavaScript is a well-known scripting language introduced in server-side and client-side dynamic web applications with a comparable object-oriented script code which can be ported to the C++ API if run-time performance becomes an issue. As a result the user is provided with a means of customisation and TAP can be integrated into existing applications.

Distributed networking made easy

Decentralised working patterns in a design group require a single location where all participants' contributions can be presented, stored and discussed. For this reason, TAP provides a client-server infrastructure. The TAPserver can be seen as the conference table where all decisions are discussed or the whiteboard where details are sketched. The server is integrated transparently within TAP with only a few settings required. Communication is via a very simple meta-protocol based on TCP/IP. All calls are wrapped by a very lean text-based command structure. The network layer is also enclosed within a wrapper in TAP and could easily be changed to another network protocol if necessary. For participants with full-immersive VE, TAP handles data using a special priority-scheduler. The actions of an immersive working designer can be distributed to all participants of a design session. Sharing data (models, text documents, pictures, sounds and metadata) can be done in two ways. One way is to use web sharing i.e. the server distributes links to the rest of the group to resources you have found or made on the web or by server sharing i.e. where you upload data to the server.

Prototypes

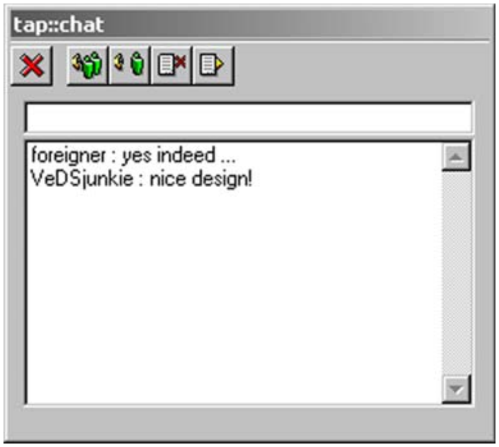
TAP is conceived as a playground in which to develop and examine the performance of a variety of prototypes. Prototypes are used to verify particular issues in the research of VE and architecture. TAP is therefore a tool to build tools and the following prototypes serve to demonstrate and examine some of the capabilities and features to be expected in a fully-blown system.

TAPserver

This is the central hub for all net-enabled TAP applications. Here all kinds of media are stored and it can also be used as a relay station for CSCD. The TAPserver can be complied and used on several platforms including Windows, Linux, IRIX and MacOS X. The provision of user policies is envisaged.

TAPclient

This is the full-featured Version of the TAP client application. The TAPclient includes an instant-messaging module, a digital scrapbook and a 3D Simulation module as well as several drivers for Tracking-input devices. The TAPclient can act as a desktop application as well as a fully-immersed VE and can be smoothly switched between those modes. Furthermore TAPclient provides a small IDE for editing and starting scripts within the application. The TAPclient includes all core technologies of TAP within one software. When in a VE, incoming text messages show up as bubbles within the VE, when in desktop mode, the same messages would appear in the chat window. Similarly drawings from the scrapbook appear as a portable panel in the VE or as a separate window in the desktop environment. By way of example, desktop-users can drag screenshots from their windowed-view of another user's immersive session and make annotations on it, returning these then to the immersed user's environment via the network. This is just one example of a variety of communication possibilities which will be examined in more detail in future research.



foreigner : yes indeed ...
VeDSjunkie : nice design!

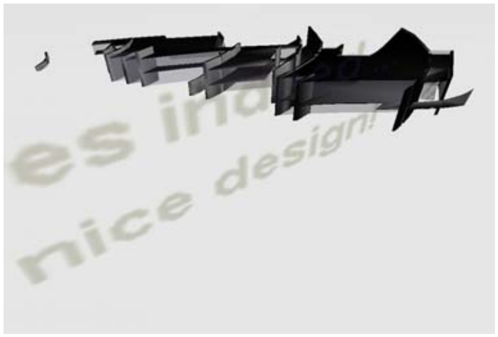


Figure 3. instant messaging on desktop

Figure 4. same messages inside immersive VE

TAP lean

TAPlean is a stripped or 'lean' version of the client which can be used in desktop environments or immersive environments. Unlike the client TAPlean contains no network capabilities, scrapbook or scripting capabilities and is designed instead as a partial module as a means of testing rendering and interaction techniques in the digital architectural design process. Like the other prototypes it is constructed using the same set of basic source-code modules.

Future perspectives

At the moment the representation of media objects within TAP is restricted to text, pictures and models. Future versions will see these extended to include audio, video and other kinds of media which may be useful in describing and communicating the architectural design. The next development step will be to track the process of an architectural design with all kinds of media and annotations – to provide a new kind of brainstorm container for the early stages of a design.

The software is in an early development stage and constantly undergoing adaptation and improvement. The concepts of TAP will be tested using on-stage experiments which we hope will provide us with more detailed feedback about the API in TAP.

Acknowledgements

We like to thank the wxWindows team (www.wxwindows.org) the critical input from MSc. CAAD Thorsten M. Lömker, Prof. Thomas Kvan and Marc Aurel Schnabel and the igroup (www.igroup.org).

References:

Do, E. Y.-L. (2000): Sketch That Scene For Me, Proceedings of eCAADe2000, Weimar, 265-268.

Garner, S. (2000): Is Sketching Still Relevant in Virtual Design Studios?, Proceedings of DCNet, Sydney.

LaViola, J., Jr., Holden, L. S., Forsberg, A. S., Bhuphaibool, D. S. and Zeleznik, R. C. (1998a): Collaborative Conceptual Modeling Using the SKETCH Framework, Proceedings of IASTED International Conference on Computer Graphics and Imaging, 154-158.

Regenbrecht, H., Donath, D., Kruijff, E., Beetz, J., Grether, K. and Seichter, H. (2000): VRAM – a virtual reality aided modeller, Proceedings of eCAADe 2000, Weimar, 235-237.

Schmitt, G. (1999) Information Architecture – basis of CAAD and its future, Birkhäuser, Basel, Switzerland.

Schnabel, M. A., Kvan, T., Kruijff, E. and Donath, D. (2001): The First Virtual Environment Design Studio,Proceedings of eCAADe2001, Helsinki, 394-400.

Vries, B. d., Achten, H. and Jessurun, J. (1998): What offers Virtual Reality to the Designer?, Proceedings of Conference on Integrated Design & Process Technology, Berlin, Germany.

LaViola, J., Jr., Holden, L. S., Forsberg, A. S., Bhuphaibool, D. S. and Zeleznik, R. C. (1998b): Collaborative Conceptual Modeling Using the SKETCH Framework,Proceedings of IASTED International Conference on Computer Graphics and Imaging, 154-158.