Using the Amacont Architecture for Flexible Adaptation of 3D Web Applications

Raimund Dachselt* Michael Hinz† Stefan Pietschmann‡
Department of Computer Science
Dresden University of Technology
Germany

Abstract

Providing personalized content and applications which also address heterogeneous capabilities of multiple device types becomes a significant challenge of today's Web development. By now most of the work in the field of adaptive hypermedia is devoted to Web content such as hypertext, images, video, and animation, but rarely to three-dimensional graphics. Research on the adaptation of 3D graphics is still in its infancy. The generic approach introduced in this paper combines the AMACONT adaptive hypermedia architecture with the component-based 3D document model CONTIGRA to achieve 3D adaptation methods such as parametric, structural, and implicit rule-based adaptation as well as rich hypermedia adaptation across media boundaries. In addition a layered context modeling framework supports not only personalization, but also adaptation to device and other contexts. It is shown that the approach can be generalized to other XML-based 3D document formats, e.g. X3D.

CR Categories: D.2.11 [Software Engineering]: Software Architectures—domain-specific architectures, declarative languages; H.5.1 [Information Interfaces and Presentation]: Multimedia Information Systems—artificial, augmented, and virtual realities; H.5.4 [Information Interfaces and Presentation]: Hypertext/Hypermedia—Architectures, User issues

Keywords: Adaptation, personalization, context modeling, 3D graphics, 3D user interfaces, virtual environments, rich media, AMACONT, CONTIGRA

1 Introduction

Rapid advancements in computer graphics hardware along with the increasing availability of rich media technologies including 3D formats lay the ground for a new type of Web applications enhancing the user's experience. Considering the ever growing number of heterogeneous end devices including traditional PCs, subnotebooks, PDAs, smart phones and others, it becomes apparent that a one-size-fits-all solution will no longer be suitable for current and future Web applications. That is where adaptation regarding user preferences, device capabilities and other contexts turns out to be the appropriate solution. Meanwhile there exists seminal work and a variety of technologies and systems in the field of adaptive hypermedia [Brusilovsky 1996] on personalization of Web and media content. This basically includes hypertext, graphics, images, video, and animation, but rarely 3D graphics, even though a variety of application domains exist. 3D adaptation can be applied to all domains of Web applications including 3D content or even 3D user interfaces. Just to name a few promising areas for applying 3D adaptation: e-commerce (product presentations and 3D shops), e-learning (tutoring and training), product services (maintenance and repair), 3D collaboration (e.g. group communication), entertainment, commercials/advertising, tourism (e.g. mobile 3D city guides) and many more.

However, it can be noticed that research on the adaptation of 3D graphics is still in its infancy. One strategy for applying adaptivity to interactive 3D graphics is to take existing Web3D formats and architectures and extend them with adaptation capabilities, like it was already done by a few researchers (see section 2). The other way, also proposed by Brusilovsky in [2004], is to extend existing adaptation techniques and architectures developed in the field of adaptive hypermedia and enhance or adjust them to allow for 3D adaptation. This is the way we have chosen for this work in order to use the strengths of the established AMACONT adaptation architecture [Hinz and Fiala 2004] with its variety of adaptation techniques and context modeling abilities. Moreover, such an approach also allows at a low cost to treat adaptive 3D graphics like other adaptive media, thus supporting our goal of a seamless integration of adaptive 3D graphics within other adaptable hypermedia.

As already pointed out by [Chittaro and Ranon 2004] it is difficult to integrate languages for representing 3D content (such as VRML or X3D) in existing adaptive hypermedia architectures. We have chosen the component-oriented 3D document format CONTIGRA [Dachselt et al. 2002] as the basis, which promises to be well suited in combination with the AMACONT adaptation architecture. This combination allows for a rich set of adaptation possibilities within a Web3D application as well as across media boundaries. Moreover, it will be shown that other XML-based Web3D formats could also be used within the proposed architecture using the same adaptation techniques.

Another issue not sufficiently covered by current research is the adaptation of 3D content to contexts other than user preferences or user interactions in virtual worlds. Especially in mobile environments we are faced with limited user attention, a high degree of individualization and inherently changing contexts. Therefore, to deliver compelling adaptive content it would be desirable to also adapt 3D graphics to mobile device capabilities, the user's current location, current browser size or installed 3D plug-in and many more. For this purpose we suggest to use a modular context modeling framework instead of a fixed user model.

The paper is structured as follows: The next section reviews existing literature related to adaptation of 3D content. Section 3 introduces the overall adaptation architecture, the embodied context modeling framework as well as the document format used. Section 4 then introduces specific adaptation techniques for 3D Web appli-

*e-mail: raimund.dachselt@inf.tu-dresden.de
†e-mail: michael.hinz@inf.tu-dresden.de
‡e-mail: stefan.pietschmann@inf.tu-dresden.de

cations developed with this architecture. The following section is devoted to important implementation issues, the generalization to other 3D formats, and a discussion of the chosen approach. The paper concludes with a summary of the results and an outline of future work.

2 Related Work

Work on *adaptive hypermedia systems* has been carried out for more than a decade. At the beginning it was mostly applied in e-learning scenarios and basically user-centered, i.e. focused on user modeling and resulting in adaptive content selection and adaptive recommendation based on user interests [Brusilovsky 1996]. Meanwhile research on adaptation is also driven by the mobile and ubiquitous computing community taking into account other contexts such as location, time, computing platform properties, or bandwidth [Billsus et al. 2002; Hinz and Fiala 2004]. At present a huge body of literature is available on adaptive hypermedia mechanisms and systems for 2D Web applications. However, this research was only rarely applied to adaptation of 3D graphics until now.

There exists a number of research projects concentrating on *adapting only the delivery of 3D geometry* to various terminal or network capabilities, e.g. using multi-resolution models or LOD techniques. A recent attempt was for example made by Kim et al. [2004] to adapt three-dimensional content at the rather low level of bitstreams to multiple target devices focusing on shapes and animation data for virtual characters.

Looking at research on personalization of 3D applications, one of the *early research work* was done by Lenzmann and Wachsmuth [1996] on multimodal communication within a multi-agent virtual environment utilizing a reinforcement learning method. An increase of work in the field of 3D adaptation can only be observed in recent years. Usually the approaches originate in *Web3D research introducing adaptive features in certain application scenarios* and 3D areas. A few results are described in the following three paragraphs.

In [Celentano and Nodari 2004] the focus is laid on interaction adaptivity. Utilizing an agent based approach it is attempted to anticipate the users' behaviors by monitoring their interaction patterns. To our knowledge it is a pure client-side approach utilizing VRML sensors and scripts as well as a restricted user model.

In the field of e-learning applications two VRML-based approaches are to be mentioned. In [Estalayo et al. 2004] a client-server architecture is presented for generating VRML applications. Multimedia information associated to scenes are separated from the 3D models and adapted employing a dynamic customization by external parameters. The system also incorporates some features of device modeling on a lower level. Dos Santos and Osrio present in [2004] an intelligent and adaptive virtual environment providing user modeling in an implicit (analysis of the users' navigation and interaction) and explicit (form-based) way. This includes an intelligent agent for navigation. It is yet limited to user adaptation (personalization) and does not address Web-based applications.

Further work has been done in the field of adaptive navigation support for virtual environments by [Hughes et al. 2002]. They suggest several adaptive navigation support technologies, such as direct guidance and adaptive annotation [Brusilovsky 2004]. This is one of the first attempts to combine traditional hypermedia and VE research. Similar techniques were implemented by [Chittaro et al. 2003] based on an approach to automatically derive and personalize guided tours for VRML worlds building upon the AWE3D (Adaptive Web3D) architecture by the same authors [2002].

Whereas the previously mentioned research focus on a certain field of Web3D graphics, the VRML-centered architecture AWE3D was one of the first *general purpose architectures for generating and delivering adaptive 3D content* [Chittaro and Ranon 2002]. It is a modular client-server-architecture with a usage data sensing module. This employs sensors for monitoring user's actions within a client's VRML browser. A usage data recorder, user model database, and personalization module reside on the server side. The VRML world creator module eventually generates 3D content based on PROTOs. Still, limitations of this architecture can be seen in the restriction to VRML content and in the difficulties to integrate AWE3D in existing adaptive hypermedia technologies and tools [Chittaro and Ranon 2004].

Whereas most approaches are based on VRML, Chittaro and Ranon also exploit in [2004] the XML-based successor X3D for the adaptive manipulation of 3D Web content. They use a separate XML-document for defining a list of adaptive content elements being related to elements of the original X3D file via DEF names. The adaptation engine selects the proper content and inserts it into the X3D code. The approach sketched there aims at *integrating 3D content into existing AH architectures*. This is also one of the main goals of the work presented here, since it is the essential basis for adaptation of various Web3D formats and cross media adaptation.

Most of the related approaches mentioned here concentrate on user modeling, taking into account his or her preferences or interaction behavior. Generic adaptation mechanisms considering various device capabilities, current location, browser sizes in terms of layout etc. are rarely or only separately available, if at all. Burigat and Chittaro break new ground with their work [2005] developing GPS-based mobile guides with adaptive VRML models. From that follows that modeling more than just the user context is an important issue and another goal of this work.

3 A Generic System Architecture for 3D Adaptation

In this chapter we present our generic approach to 3D adaptation. It is built on the underlying AMACONT system architecture for the dynamic generation of Web presentations, which is described along with its context modeling framework in the next two subsections. Since it is based on a declarative XML document format [Fiala et al. 2003], a 3D component format providing similar characteristics is lending itself for 3D adaptation. Therefore we chose the declarative 3D component model CONTIGRA as the basis [Dachselt et al. 2002], which is concisely explained in subsection 3.3. Combining these technologies we could achieve several types of 3D adaptation, which are described in detail in the following main section.

3.1 Overview of the Amacont Adaptation Architecture

To achieve various means of 3D adaptation we chose the AMACONT system architecture [Hinz and Fiala 2004] as the basis for this work. It aims at the dynamic generation of Web presentations tailored to users' varying needs, device capabilities and context data such as the users' location. The architecture provides techniques for static and dynamic adaptation of content, layout and structure. Furthermore, mechanisms for modeling dynamically changing context

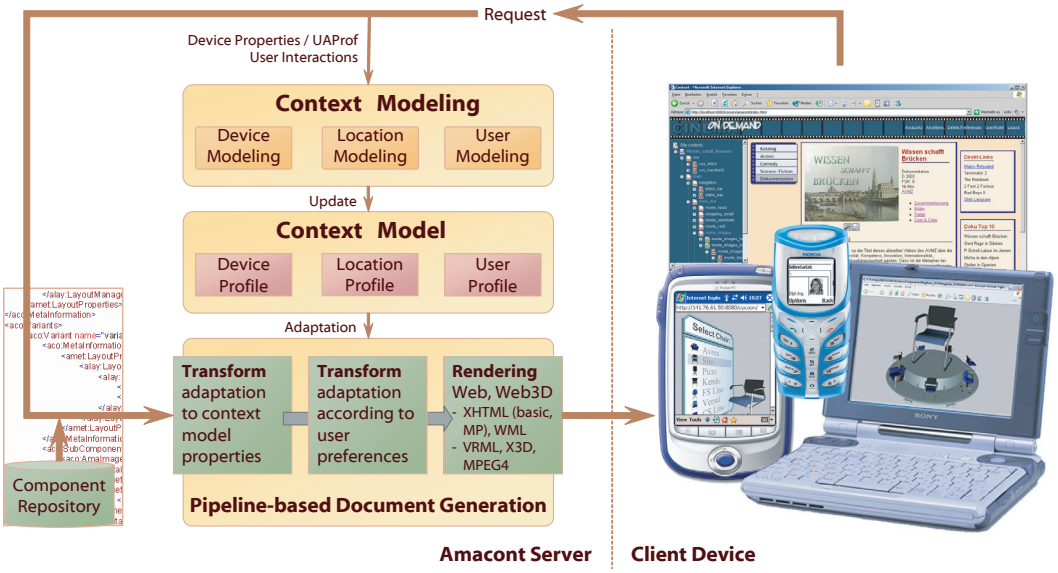

Figure 1: Overall AMACONT system architecture

information like user's device capabilities, preferences, and location are provided in order to guarantee up-to-date context models for the whole content generation process (see section 3.2).

Figure 1 depicts the overall system architecture. Note that in most of the current adaptive hypermedia systems three well established layers can be found: user data acquisition, user model (representation), and adaptation (production) [Brusilovsky 1996; Chittaro and Ranon 2002]. Our system architecture is constructed accordingly, but extending them from merely user-oriented to the context-oriented levels: *Context Modeling, Context Model,* and *Adaptation Pipeline*.

For each user request, a complex document encapsulating all possible variants concerning its content, layout, and structure is retrieved from a component repository. Such documents are described using an XML-based, component-oriented document model [Fiala et al. 2003]. In the following phase of the pipeline-based document generation the XML-documents are transformed in various steps according to the context model containing user, device, and location properties. Thus elements are removed from the big document to eventually contain only the desired, adapted information. The last step of the pipeline carries out the transformation to a specific output format, such as XHTML, WML, X3D, or MPEG-4. The resulting document can then be delivered to the specific target device and rendered accordingly. For the detailed description of the adaptation pipeline see [Hinz and Fiala 2004]. The overall architecture was implemented on the basis of the XML publishing framework Cocoon [Cocoon 2006], which is further explained in subsection 5.1. The next section will provide more details on the context modeling part of the architecture, since it is an essential basis for the adaptation process.

3.2 Modeling Context for the Adaptation Process

For the modeling of context information the AMACONT architecture provides a modular, extensible context modeling framework that offers an integrated approach for gathering, processing and representing context information. Thus it can be effectively used for adapting three-dimensional Web applications. Furthermore, it allows for faster development and deployment of context-aware adaptive Web applications and provides a set of modeling modules (i.e. components). They support developing applications featuring device independence, location dependence, and personalization. The architecture of the framework is divided into three different layers: the sensor layer, the context modeling layer, and the context model layer (see Figure 2).

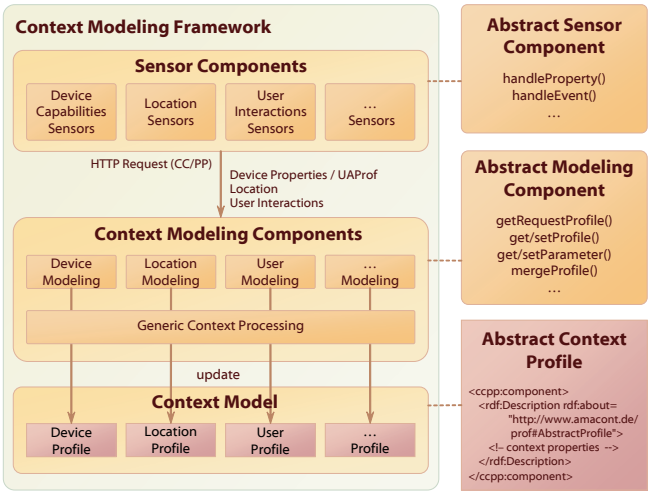

Figure 2: Architecture of the AMACONT context modeling framework

The sensor layer encapsulates the *sensor components* that gather and monitor different properties of the user's context (e.g. user's location, device capabilities, user interactions within a Web application).

This sensor information is processed in the second layer by the *context modeling components* in order to model the user's context. An important component of this layer is the *user modeling* [Hinz and Fiala 2004]: Besides user modeling strategies that require explicit feedback (e.g. filling out questionnaires) an automatic modeling

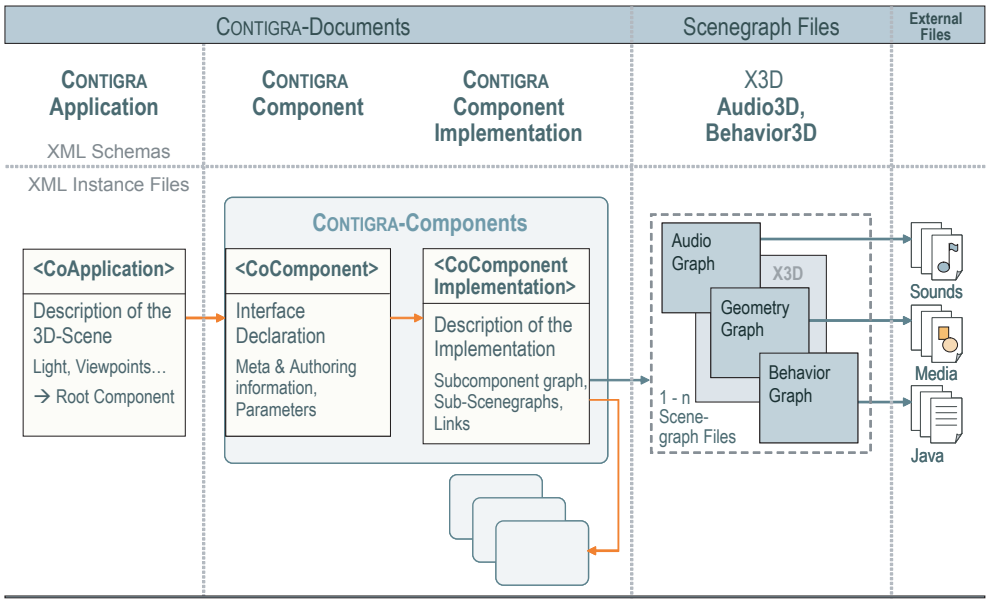

Figure 3: The CONTIGRA markup languages and corresponding XML documents

mechanism is also provided that allows an implicit analysis of user interactions and constitutes the basis for the implicit rule-based media adaptation explained in section 4.4. The layer also comprises *device modeling* components that process sensed device capabilities and store them in a device profile that is based on an extended UAProf specification [UAProf 2001]. In addition to user and device modeling the AMACONT project also provides *location modeling* mechanisms that can be used to develop and deploy location-based services using GPS technology or any third party location server supporting the Mobile Location Protocol [Hinz and Fiala 2005].

In order to be accessible by various Web applications, the resulting context data have to be represented in a sophisticated way. Therefore, the proposed context modeling framework offers in the third layer an extensible *context model*. It stores all gathered and processed data in different context profiles that are associated with various application scenarios. Each profile relies on CC/PP (Composite Capability / Preference Profiles) [Klyne et al. 2003], an RDF grammar for describing device capabilities and user preferences in a standardized way. Taking device capabilities as an example, they are represented by the device profile and can be effectively used for device independent Web applications. The information provided by the device profile are of special interest for adapting 3D graphics: the type of the device, available screen size, screen orientation, color capability, bits per pixel, CPU, memory, and connectivity, but also the MIME-Types for available 3D player plug-ins as well as information on interaction possibilities such as text and voice capabilities.

Already implemented context modeling and sensor components for processing the device and location context are presented in [Hinz and Fiala 2005]. In order to support the addition of further context modeling techniques, the framework provides generic extension mechanisms. For adding new sensor components to the framework it provides a client-server communication manager that enables the registration of various client side sensor components. See section 4.4 for adding sensor components to 3D worlds. It is also possible to develop new modeling components and to extend the context model. In the case of a 3D application the existing device, user, and location modeling components along with their respective profiles can easily be reused.

3.3 The Component-based 3D Document Model Contigra

After having described the generic AMACONT system architecture, we will now have a look at the specific 3D document part being processed by it. Since the AMACONT architecture processes component-oriented XML documents within the adaptation pipeline, we chose the CONTIGRA component format [Dachselt et al. 2002; Dachselt 2004] as the basis for this work. CONTIGRA was designed for the easy construction of Web-enabled, desktop Virtual Reality applications and 3D scenes. The document-centered approach is based on the notion of three-dimensional, high-level components abstracting from the underlying 3D scene graph format, e.g. X3D. A set of multi-layered, declarative markup languages based on XML Schema allows the creation and authoring of structured documents describing 3D component interfaces, their implementation, configuration, and assembly to build interactive 3D Web applications. Figure 3 depicts the different XML markup languages (upper part) used along with their corresponding instance documents (lower part).

An instance document of the grammar *CoApplication* defines an overall 3D application in terms of typical scene parameters (such as lights and viewpoints) and a reference to a 3D root component containing the whole scene. This component - like every other reusable CONTIGRA component - is defined by two XML documents, one for its interface (according to the schema *CoComponent*), the other for its implementation (an instance of *CoComponentImplementation*). The interface document contains configurable high-level parameters defining a component's functionality as well as authoring and other meta information. The implementation document firstly contains a component graph which is a transformation hierarchy containing references to other 3D components. For all additional parts of the scene, which are not yet available as a reusable component it secondly contains a scene graph. This is split into three parts for audio, geometry, and behavior nodes, usually as references to external scene graph files. At present X3D is used for the geometry, the extended grammars Audio3D [Hoffmann et al. 2003] and Behavior3D [Dachselt and Rukzio 2003] for the definition of spatial audio and complex behavior nodes respectively. The third part of

a *CoComponentImplementation* document consists of a separated link section connecting referenced parts (subcomponents and scene graph nodes), conceivable as an extended ROUTE concept.

It is to be mentioned that an authoring tool called ContigraBuilder can be used to author such XML-based 3D applications. Since they are described independently from proprietary 3D toolkits or APIs, they can be translated into target formats such as VRML97, X3D, OpenSG, or Java3D using an internal object model (data binding) or XSLT-Stylesheets.

4 Adaptation Techniques for 3D Web Applications

As already mentioned earlier, we consider the widening of the rather user-centered term personalization to other context information to be an important issue for 3D adaptation, too. The types and dimensions of context-based adaptation in the field of interactive 3D graphics include the following:

- *Personalization (user context adaptation)*. First of all 3D content (also across media boundaries) can be customized, i.e. parts of the scene can be exchanged, removed, added, rearranged, displayed in varying quality (LOD) etc. Secondly the user interface can be adapted in terms of layout (content structure) and interaction. Interaction adaptation applies to navigation, selection, and manipulation in 3D worlds. Interaction techniques and 2D or 3D user interface parts could be chosen according to user preferences or skills.

- *Device context adaptation*. This includes all adaptations to specific, often limited resources (i.e. device characteristics). Especially 3D graphics rendering is very much depending on available memory size, processing power, hardware acceleration, and even display size and image resolution. Available input devices also need to be considered for adaptation in terms of DOF, speed, precision, and multimodal interaction support. User interfaces along with their widgets and interaction techniques need to be adapted accordingly.

- *Extended context adaptation* includes adaptation to location, time, connection, networked users (for group collaboration), and even temperature (which might influence the possible interaction ability of a user). Typical adaptive applications will be location based services in mobile scenarios, such as tourist guides or virtual shopping assistants.

After having described those basic types of 3D adaptation, the following subsections introduce various ways of realizing them technically using the AMACONT architecture. *Structural units* within an XML-based document format are hereby the basic idea. These units can be the original AMACONT document components for traditional 2D Web pages, the CONTIGRA 3D components but also VRML97/X3D-Prototypes or even a set of XML-elements of an arbitrary XML-based hypermedia format. Though we use the term structural units and components synonymously, the term *component* reflects best the semantically and functionally contained nature of these units. These components can be assembled on various levels to build up complex Web applications in 2D, 3D, or even mixed. This assembly should be of a hierarchical nature as we have seen with the CONTIGRA component graph. In addition to that, components usually provide parameters as means of configuring the appearance or functionality of such a component. From that we can conceive two explicit and a derived basic adaptation mechanism, whereas the fourth differs in being an implicit technique:

- *Parametric Adaptation.* Customizing the specific instance of a component in terms of appearance, behavior or functionality in general.

- *Structural Adaptation.* Adapting the hierarchical structure of complex components, e.g. by removing, exchanging, or adding complete components.

- *Rich Media Adaptation.* Whereas parametric and structural adaptation only allows adaptation within one medium or format, this mechanism cares for intra-adaptation of various media formats within a Web application (take for example adaptive 3D-graphics as part of a hypertext application).

- *Implicit Rule-based Media Adaptation.* An implicit form, where user interactions and the semantic description of media objects are used to generate rules by a learning algorithm and utilized for adaptation.

Before explaining these mechanisms in detail within the following subsections, it shall be outlined what is already provided for explicit adaptation by the AMACONT architecture. First of all the XML Schema *AmaAdaptation* allows the definition of variants and adaptation logics. We define variants as different occurrences of structural units or their parameters. Thus variants can be applied to single configuration parameters of a component as well as to complete components or even hierarchies. *Adaptation logics* are used to define adaptation conditions. They are also represented in XML utilizing a reverse Polish notation. Besides the option of doing smaller arithmetical calculation within the statements, access to context model parameters is also granted with specific XML elements. For more details on the AMACONT adaptation logics see [Fiala et al. 2003]. The schematic code in the following listing depicts how variants and their corresponding logic can be defined. The AMACONT pipeline architecture as described in section 3.1 includes a transformation step, where the logical conditions are evaluated at runtime and variants being chosen accordingly. The logic and variant tags are thereafter removed from the document, just leaving the content within the chosen variant tag.

```
<Variants>
  <Logic>
    ...
  </Logic>
  <Variant name="variant1">
    <Content1> ... </Content1>
  </Variant>
  <Variant name="variant2">
    <Content2> ... </Content2>   ⇒   <Content2> ... </Content2>
  </Variant>
  ...
  <Variant name="variantX">
    <ContentX> ... </ContentX>
  </Variant>
</Variants>
```

From that follows the basic idea of an AMACONT-based adaptation, which is importing the *AmaAdaptation* Schema in the chosen 3D document format. Therewith it is possible to use the adaptation mechanisms described in more detail in the following subsections for the CONTIGRA component format.

4.1 Parametric Adaptation

As mentioned above, parametric adaptation is defined as the adaptation of a configurable part of a structural unit, usually of a component's parameter, to context information. Thus it implements the adaptive hypermedia technique of altering information fragments.

This adaptation mechanism is explained using the simple example of a 3D chair component in CONTIGRA containing the following parameter definition *CushionColor* amongst others from the respective *CoComponent* interface document.

```
<Parameter name="CushionColor" dataType="CoAnyURI" …
 description="Color of a chair's cushion" semantics="appearance">
   <cpt:CoAnyURI>"Models/blueCushion.wrl"</cpt:CoAnyURIs>
</Parameter>
```

Please note the high-level nature of the parameter, since not only a texture or a material field is exchanged, but a complete cushion geometry. Integrating the AMACONT *AmaAdaptation* schema now allows defining alternative variants within this or another parameter during authoring time. Thus a dynamic adaptation of component parameters depending on predefined context conditions is made possible during runtime.

The following code fragment illustrates this approach by extending the parameter definition seen above. Note that all additional elements from the included *AmaAdaptation* schema contain the prefix *aada*. These statements define the adaptation behavior for the sample parameter *CushionColor*. The expressions within the `<aada:Logic>` tag determine whether the chair will have a blue or red cushion (through the `<aada:ChooseVariant>` tag). In this example the decision is made with a simple if-then-else statement comparing the favorite color of a user with the constant *blue*. Note that *Favorite Color* is a user characteristics being retrieved from the context model with the help of the `<aada:UserParam>` element. Eventually the variant tag, e.g. `<aada:Variant name="blue_fabric">` contains the actual XML code of the original 3D format, in this case a reference to a VRML file blueCushion.wrl.

```
<Parameter name="CushionColor" dataType="CoAnyURI" …
 description="Color of a chair's cushion" semantics="appearance">
  <aada:Variants>
    <aada:Logic>
      <aada:If>
        <aada:Expr>
          <aada:Term type="=">
            <aada:UserParam>Favorite Color</aada:UserParam>
            <aada:Const>blue</aada:Const>
          </aada:Term>
        </aada:Expr>
        <aada:Then>
          <aada:ChooseVariant>blue_fabric</aada:ChooseVariant>
        </aada:Then>
        <aada:Else>
          <aada:ChooseVariant>red_fabric</aada:ChooseVariant>
        </aada:Else>
      </aada:If>
    </aada:Logic>
    <aada:Variant name="blue_fabric">
      <cpt:CoAnyURI>"Models/blueCushion.wrl"</cpt:CoAnyURIs>
    </aada:Variant>
    <aada:Variant name="red_fabric">
      <cpt:CoAnyURIs>"Models/redCushion.wrl"</cpt:CoAnyURIs>
    </aada:Variant>
  </aada:Variants>
</Parameter>
```

Figure 4 shows a result of the adaptation for a user with the preferred color red, whereas Figure 5 (left) shows the same chairs of the ring menu being adapted dynamically to user preference blue (please compare the figures within the color plate section of the proceedings). Note that this specific example of parametric adaptation belongs to the general adaptation type *personalization* according to user preferences.

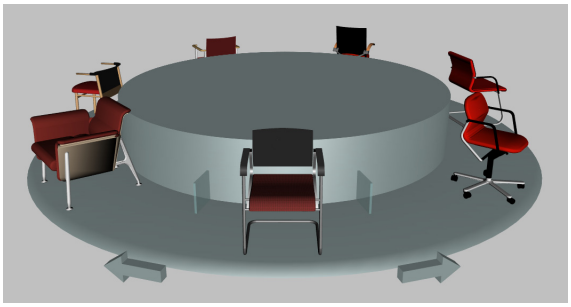

Figure 4: Ring menu with chairs for user-preferred color red

4.2 Structural Adaptation

Besides customizing a component using parametric adaptation it is often necessary to remove a component from the 3D scene (e.g. due to limited screen space), to add others (e.g. additional 3D widgets to compensate for the lower degrees of freedom of a changed input device) or simply to exchange it, for example to adapt to user's preferences or limited system resources. The goal of the structural adaptation can be defined as adapting the hierarchical structure of a 3D scene in terms of insertion, removal, or choice of structural units, all being well-known adaptive hypermedia techniques.

The AMACONT *AmaAdaptation* schema is facilitating structural adaptation in the same way like parametric adaptation, i.e. through the definition of adaptation logics and various variants, this time of components. The following code fragment illustrates it just showing the logic and variants. In this application example the type of the used 3D menu for selecting chairs is exchanged depending on the available browser size. Within the variant tags again resides the actual XML code to be finally used. In this case it is a reference to a CONTIGRA *CoComponent* document describing a component's interface. This means, entire components can be exchanged at runtime while providing the same functionality as before.

```
...
      <aada:Expr>
        <aada:Term type="gt">
          <aada:UserParam>InnerSizeX</aada:UserParam>
          <aada:Const>800</aada:Const>
        </aada:Term>
      </aada:Expr>
...
<aada:Variant name="ringmenu">
  <ComponentInstance DEF="RingMenu"
      fileRef="RingMenu/RingMenu.coc"/>
</aada:Variant>
<aada:Variant name="floatingmenu">
  <ComponentInstance DEF="FloatingMenu"
      fileRef="FloatingMenu/FloatingMenu.coc"/>
</aada:Variant>
...
```

Figure 5 shows on the left an application prototype with a browser size greater than 800 pixels, so that the ring menu is chosen and displayed for selecting chairs. On the right the user reduced the browser size, which resulted in a changed and space-saving floating menu for achieving the same selection task. Note, that this adaptation does not need to be initiated by the user, but might also result from the limited display size of the output device such as a PDA. Whereas this example only illustrates the component exchange during runtime, components might also be added or removed in a similar fashion.

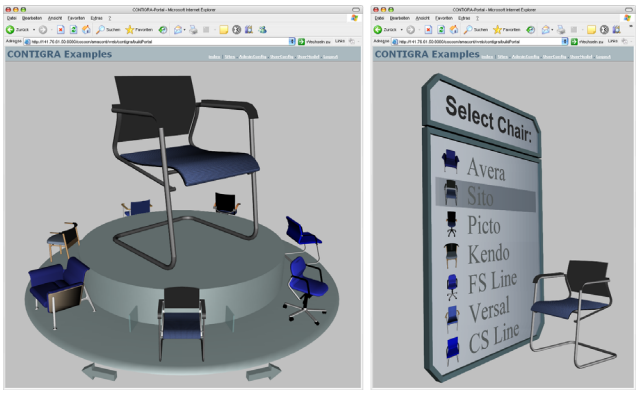

Figure 5: Left: ring menu with chairs for user-preferred color blue. Right: Same application with adapted menu (now floating menu)

This specific example of structural adaptation belongs to several types of 3D adaptation. On one hand it can be seen as an adaptation to the available display size in terms of *device context*, on the other hand as a user interface adaptation in terms of *personalization*, assuming the user adjusting the browser size. Please also note that both structural and parametric adaptation are technical ways of adaptation which can be used for all types of context dependent adaptation.

4.3 Rich Hypermedia Adaptation

The parametric and structural adaptations described above are ways of achieving adaptations within a single document or media type respectively. Given a structured XML-based 3D format all structural units (e.g. CONTIGRA components) can be adapted as a whole or with their parameters. However, current Web applications usually consist of more than one media format. It can also be observed that the majority of standardized Web formats is based on XML grammars. Consequently, a typical rich media Web application could be written in XHTML, include SVG graphics, SMIL animations, X3D content, or even other XML-based media assets. It is therefore desirable not only to define a common layout and allow intra-media-communication (e.g. data exchange), but also to care for a uniform adaptation mechanism across all media involved. We call this seamless adaptation across various Web-based media *inter-media-adaptation*. Furthermore, this mechanism should be the same to the adaptation within a single media, called *intra-media-adaptation*. Besides, inter-media-adaptation also allows for the seamless exchange of different media components. Take for example a 3D application on a PC with enough processing power to render not only a 3D object but also all interface elements in 3D. On a PDA the same application could automatically provide 2D interaction elements instead, thus reducing the graphics processing demands and sparing resources.

Inter-media-adaptation can be technically achieved using the integration capabilities of the AMACONT document model [Fiala et al. 2003]. It provides a declarative, component-based approach for adaptive, dynamic Web documents on the basis of XML-technology. It allows for the definition of various media components (e.g. structured texts, images, sounds, videos, java applets, Flash presentations, and 3D scenes) on various abstraction levels. Thus it is possible to integrate a 3D scene description with intra-media-adaptation into an AMACONT document. Using an additional step in the adaptation pipeline the 3D adaptation is carried out and integrated into the whole application.

An example shall illustrate inter-media-adaptation across media boundaries. Figure 6 depicts a prototype of a simple Web-based application to configure various conference rooms within a congress center. On the left one can choose the type of seating, such as in U shape, for banquets, or parliament seating. Typically for today's Web applications in this field one can see the resulting seating capacity for various room in a table and with an added sample picture on the right. All parts can be realized using the AMACONT document format, which is for example transformed to XHTML.

Now consider a user's computer providing substantial 3D graphics performance. In this case the complete structural unit on the right (i.e. the table with the sample picture) can be dynamically adapted and exchanged as it is shown in Figure 7. Now a complex CONTIGRA component was transformed to an interactive VRML scene, which better visualizes the furnished rooms. It even allows adjustments of rows, columns, and other layout features. Changes are immediately reflected within the scene, the total number of chairs is also displayed accordingly.

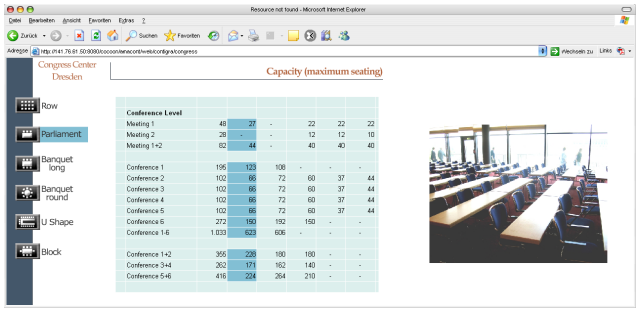

Figure 6: Prototype Website for adjusting the seating capacity of conference rooms, here with XHTML table and picture.

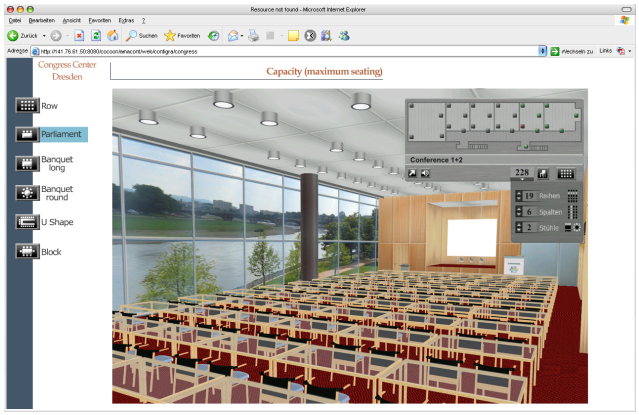

Figure 7: Adapted Website for adjusting the seating capacity of conference rooms, now with interactive VRML scene.

4.4 Implicit Rule-based Media Adaptation

Whereas parametric and structural adaptation (also the derived rich hypermedia adaptation) allows for an explicit determination of 3D adaptation by means of defining variants and adaptation logics within the XML documents, rule-based media adaptation can be considered rather implicit. Even though parameters such as media size are also adapted with this technique, it is not realized by applying explicit selection methods, but by implicitly generated lists

of rules. This adaptation form allows users to interact with media components or other objects contained in a presentation, to implicitly derive user preferences from those interactions and to dynamically update the resulting Web presentation according to those preferences. It is to be mentioned that with all adaptation techniques the adaptation pipeline is used for document generation. In every case an appropriate transformer (see subsection 5.1) is chosen for the adaptation of the current Web presentation, either by directly choosing parameters from the user model within the logic statements (see first three techniques) or by using the list of implicit rules (representing the user model) from the context modeling framework with this technique.

Interactions of a user, e.g. within a virtual world, are sensed by appropriate sensor components. The addition of such a component is facilitated by utilizing the client-component manager of the context modeling framework (see section 3.2). Considering the usage of the VRML EAI interface within a 3D Web browser or VRML/X3D script nodes, one can imagine various ways of sensing and monitoring user behavior in a 3D application. The typical way for VRML worlds will be to use various sensors such as Proximity or Visibility like it was done in the work by [Chittaro and Ranon 2002].

The user events acquired by the sensor components are sent to the server by the client-component manager. A user modeling component then calculates user preference rules based on the incrementally working complementary discrimination learning algorithm (CDL4) [Shen 1996] and updates the user profile of the context model. Based on that profile the requested Web documents can be adapted, respectively. Take for example a mixed-media application, such as a product shop, where the user clicks on the '3D-view' icon of a product or even interacts with the subsequently displayed 3D object. This implies the user having an interest not only for 3D views of objects (the medium) but most likely also for this product category. The following Web pages could already take that into consideration.

To achieve this, the AMACONT document model provides elements for the semantic description of media elements (see [Hinz and Fiala 2004] for details). Imagine for example a 3D model of a cutter with the semantic description `(medium = Ama3DComponent, category = garden_tools)`. If the user then interacts with the 3D model, these descriptions flow into the rules of the preference profile. Interactions could for example be the activation of the 3D view, some activities inside or even the opening of an additional textual description. The rules are updated accordingly representing a user's interest in 3D media within this specific product category. Other media elements with similar semantic descriptions on this or another Web page displayed in the future will be adapted dynamically according to these rules. That means products of the same category will already have an opened 3D view along with a displayed descriptive text.

5 Implementation Issues and Discussion

5.1 Implementation using a transformation pipeline

As already mentioned in subsection 3.1 the AMACONT architecture was implemented with the XML Web publishing framework Cocoon [Cocoon 2006] running on top of the Apache Tomcat servlet container. Cocoon relies on a pipeline processing model, where each arriving server request will be assigned to a pipeline. One or more pipelines are defined in a so-called *sitemap* serving as the configuration of the cocoon framework. Each pipeline usually starts with a *generator* creating SAX (Simple API for XML) events from XML content. They are processed by a series of *transformers* which consume and emit SAX event streams. At the end of a pipeline a *serializer* transforms these streams into binary or character document-streams for final client consumption.

In the current AMACONT implementation two different sitemaps are used. On one hand this is the original AMACONT sitemap containing a pipeline for generating adaptive hypermedia Web applications and additional pipelines for the context modeling process, for handling different users and their context models, and for processing various media resources and streams (see figure 1). On the other hand this main sitemap was extended with a secondary sitemap containing changed document generation pipelines especially for adaptating 3D media based on CONTIGRA documents. This 3D-specific sitemap can be used to generate adaptive stand-alone 3D applications. In order to realize rich hypermedia adaptation combining 3D graphics and other media (see subsection 4.3) the original AMACONT sitemap can be used with CONTIGRA or other 3D document formats in terms of adaptive media components.

The main document generation pipeline of the AMACONT architecture contains several transformers. Three of the most essential are depicted in figure 1. The first, *adaptation according to context model properties*, was realized on the basis of SAX and implements the adaptation techniques parametric adaptation, structural adaptation, and rich hypermedia adaptation. The second transformer *adaptation according to user preferences*, which was implemented using the Document Object Model, evaluates the rules representing the user model. Depending on the current target device and supported document format, the adapted SAX stream is finally processed by a rendering transformer. This transformer uses specific XSLT transformations to actually generate hypermedia documents (e.g. in XHTML and WML) from the generic, adapted component description.

The 3D-specific pipeline works in a similar way. In the last step the SAX event stream (i.e. an adapted CONTIGRA document) is transformed to a specific Web3D format using an XSLT transformation. At the moment transformers to X3D and VRML are implemented, other formats such as MPEG-4 can be added. This is no adaptation-specific issue, since CONTIGRA documents (whether adapted beforehand or not) can be independently transformed to specific 3D target formats. The XSLT transformations developed for this purpose can be easily integrated as rendering transformers at the end of the pipeline.

The 3D adaptation happening within the 3D-specific pipeline takes into account the current context model, evaluates the adaptation logic, chooses the appropriate variant, and finally removes all AMACONT-specific elements from the XML code. The context model is continually updated by the pipeline being responsible for the context modeling process. This pipeline consists of a series of Cocoon actions, which evaluate the acquired sensor component values and model the user's context. The context model itself is also represented in XML within the session context of the server. The adaptation transformers then access these information for the adaptation process.

5.2 Generalization to other 3D Formats

The technical ways of adapting 3D graphics using the CONTIGRA component format were already explained above. In addition to that, these techniques can be generalized to other 3D formats, too. The first way comes with the CONTIGRA format itself, because it can be transformed to other formats as needed (see section 3.3). Since the AMACONT architecture provides an adaptation pipeline

with various transformers, it is easy to first adapt a CONTIGRA document and then to add other transformation steps, e.g. to choose a specific 3D format, which is supported by the end device. At the moment and in all our examples we only use transformers to X3D and VRML. As already mentioned, additional transformers to other 3D formats could be written likewise.

The second way is to not use CONTIGRA at all, but to choose another specific 3D document format. A drawback would be the loss of format independence, which is one of the key advantages of the CONTIGRA component approach. The basic demand for such a format is that it is based on XML, preferable on XML Schema. This is required for integrating the adaptation logic and variants definitions using the AMACONT schema and to facilitate the pipeline-based transformation and adaptation process. The format does not necessarily need to be a component-based document format (such as CONTIGRA), though components are a good notion of self-contained functional units being configurable and exchangeable as a whole for structural adaptation. However, a 3D format being a candidate for 3D adaptation using the proposed architecture should at least provide XML element definitions being suitable for adaptation. Taking for example the Web3D standard X3D it would be less useful to treat transform nodes as adaptable units, whereas a PROTO definition closely resembles the higher-level component idea. It is interesting to note that the AWE3D architecture introduced by Chittaro and Ranon [2002] also uses VRML PROTOs to generate VRML worlds personalized to the user. Still, with VRML or X3D an abstraction level from scene graphs is missing, which is provided by the CONTIGRA approach. This higher level also allows for higher level adaptations, e.g. through adapting complex behaviors being part of a CONTIGRA component.

The first step of integrating the adaptation mechanism to another 3D format would be to import the namespace of the *AmaAdaptation* schema into the own XML schema. This allows using the logic and variants statements for parametric and structural adaptation. Within the own schema one can then define the elements, where adaptivity is allowed and desired, e.g. PROTOs or Routes in X3D. Instance documents might include several variants and adaptation logics as described in subsections 4.1 and 4.2. and can be validated against the own schema and the imported *AmaAdaptation* schema. As a result of the adaptation all AMACONT-specific elements are finally removed from the XML code. This way the generated document can be displayed as usual in an appropriate 3D player or browser plug-in.

5.3 Discussion

The key distinguishing feature of the presented approach is to realize 3D adaptation on the basis of an existing hypermedia adaptation architecture. The AMACONT approach was chosen due to its format independence, the component nature of the internal document format, and due to the rich and advanced context modeling capabilities. All adaptations are performed on the server-side. From that follows, adaptive hypermedia and Web3D applications must request parts of the application from the server. This could be a new three-dimensional room being entered or a product being dynamically added or reloaded in a 3D product store. However, one drawback of the server-side adaptation is the necessary request-generate-send-display cycle being necessary each time an adaptation should be done. Thus real-time adaptation such as automatically adapted content while re-sizing a window is not supported. However, since the AMACONT approach also aims at mobile devices, the chosen client-server adaptation architecture proves to be useful with regard to the limited processing capabilities of mobile devices. Other limitations of the presented approach are the resulting overhead from context modeling and performance drawbacks due to the pipeline-based document generation process.

Beside the smaller 3D adaptation examples presented in section 4 we have implemented a bigger hypermedia application including an interactive 3D view. This application for configuring several congress center rooms with their seating capacities serves as a testbed for various forms of 3D adaptation. Beside the already mentioned rich hypermedia adaptation another example is the adaptation of the scene to the current location of the user, i.e. the *location context*. When the user walks through the real congress center building and activates the application, he or she is automatically presented the virtual room (i.e. the initial scene viewpoint) corresponding to the user's current location. Another example is adaptation according to the *context time*. With it the virtual congress center can be adapted to current time of the day by means of changing the city silhouette texture and changing the lighting conditions. See for example figure 8, where the user accesses the application during the evening in comparison to figure 7 at daytime. At the moment we are experimenting with other forms of 3D adaptation on the basis of the proposed approach.

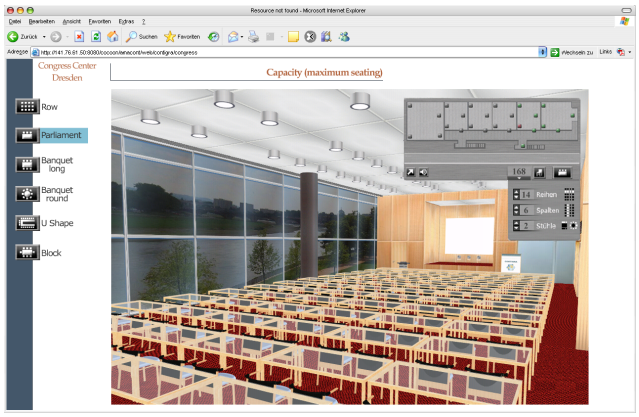

Figure 8: Website of the congress center application with adaptation of the 3D scene according to evening time (compare figure 7).

6 Conclusion and Future Work

In this paper we presented an approach to deploy the adaptive hypermedia architecture AMACONT together with the component-oriented 3D document model CONTIGRA to achieve various types of 3D adaptation within Web pages. We have identified the basic adaptation techniques parametric and structural adaptation as well as the derived form rich hypermedia adaptation. This allows for adaptation across media boundaries, assumed the structural units of a Web page being described in XML. Finally we presented an additional mechanism, the implicit rule-based media adaptation observing user's interactions. Since the architecture also comprises a generic context modeling framework, adaptation is made possible not only to user preferences, but also to device capabilities, user's location and other context information. The realized prototypes encourage us to further develop the proposed approach. We gained the impression that the employment of a general-purpose adaptive hypermedia architectures in the field of 3D graphics results in a considerably increased potential of 3D adaptation.

This needs to be further investigated by developing more sample applications and by extending the proposed adaptation methods. Although we have shown that the approach can be generalized from

CONTIGRA to X3D or other media formats, there is still research do be undertaken in the field of adapting true mixed-media Web applications containing SMIL, MPEG-4, Flash or other media. In addition to that, authoring adaptive media applications and appropriate tool support are important issues to be further investigated. Though separate authoring tools already exist for AMACONT and CONTIGRA applications, they need to be combined and generalized. In addition to that, specialized sensor components could be developed to better monitor and study users' interactions within virtual worlds. Finally it should be tested and evaluated, how users can cope with adaptation in 3D worlds compared to non-adaptive or even traditional applications. We hope to have stimulated more research within the interesting field of 3D adaptation.

We would like to thank the anonymous reviewers for their feedback, questions and suggestions, which helped us to improve this paper.

References

BILLSUS, D., BRUNK, C. A., EVANS, C., GLADISH, B., AND PAZZANI, M. 2002. Adaptive interfaces for ubiquitous web access. *Communications of the ACM 45*, 5, 34–38.

BRUSILOVSKY, P. 1996. Methods and techniques of adaptive hypermedia. *User Modeling and User-Adapted Interaction 6*, 2-3, 87–129.

BRUSILOVSKY, P. 2004. Adaptive Navigation Support: From Adaptive Hypermedia to the Adaptive Web and Beyond. *PsychNology Journal 2*, 1, 7–23.

BURIGAT, S., AND CHITTARO, L. 2005. Location-aware visualization of VRML models in GPS-based mobile guides. In *Web3D '05: Proceedings of the tenth international conference on 3D Web technology*, ACM Press, New York, NY, USA, 57–64.

CELENTANO, A., AND NODARI, M. 2004. Adaptive interaction in Web3D virtual worlds. In *Web3D '04: Proceedings of the ninth international conference on 3D Web technology*, ACM Press, New York, NY, USA, 41–50.

CHITTARO, L., AND RANON, R. 2002. Dynamic generation of personalized VRML content: a general approach and its application to 3D e-commerce. In *Web3D '02: Proceeding of the seventh international conference on 3D Web technology*, ACM Press, New York, NY, USA, 145–154.

CHITTARO, L., AND RANON, R. 2004. Using the X3D Language for Adaptive Manipulation of 3D Web Content. In *Adaptive Hypermedia 2004*, Springer, P. D. Bra and W. Nejdl, Eds., vol. 3137 of *Lecture Notes in Computer Science*, 287–290.

CHITTARO, L., RANON, R., AND IERONUTTI, L. 2003. Guiding visitors of Web3D worlds through automatically generated tours. In *Web3D '03: Proceeding of the eighth international conference on 3D Web technology*, ACM Press, New York, NY, USA, 27–38.

COCOON, 2006. Web development framework 'Apache Cocoon', The Apache Software Foundation (2006). http://cocoon.apache.org.

DACHSELT, R., AND RUKZIO, E. 2003. Behavior3D: an XML-based framework for 3D graphics behavior. In *Web3D '03: Proceeding of the eighth international conference on 3D Web technology*, ACM Press, New York, NY, USA, 101–112.

DACHSELT, R., HINZ, M., AND MEISSNER, K. 2002. Contigra: an XML-based architecture for component-oriented 3D applications. In *Web3D '02: Proceeding of the seventh international conference on 3D Web technology*, ACM Press, New York, NY, USA, 155–163.

DACHSELT, R. 2004. *A Declarative Component Architecture and Widgets for 3D Applications (doctoral thesis, in German: Eine deklarative Komponentenarchitektur und Interaktionsbausteine fuer dreidimensionale multimediale Anwendungen)*. Der Andere Verlag, Toenning, Germany.

DOS SANTOS, C. T., AND OSORIO, F. S. 2004. An intelligent and adaptive virtual environment and its application in distance learning. In *AVI '04: Proceedings of the working conference on Advanced visual interfaces*, ACM Press, New York, NY, USA, 362–365.

ESTALAYO, E., SALGADO, L., MORAN, F., AND CABRERA, J. 2004. Adapting Multimedia Information Association in VRML Scenes for E-Learning Applications. In *Proceedings of the First international Workshop LET-Web3D '04*, 16–22.

FIALA, Z., HINZ, M., MEISSNER, K., AND WEHNER, F. 2003. A Component-based Approach for Adaptive, Dynamic Web Documents. *Journal of Web Engineering 2*, 1-2, 58–73.

HINZ, M., AND FIALA, Z. 2004. AMACONT: A System Architecture for Adaptive Multimedia Web Applications. In *Berliner XML Tage*, 65–74.

HINZ, M., AND FIALA, Z. 2005. Context Modeling for Device- and Location-Aware Mobile Web Applications. In *3rd International Conference on Pervasive Computing (Pervasive 2005), PERMID 2005*.

HOFFMANN, H., DACHSELT, R., AND MEISSNER, K. 2003. An Independent Declarative 3D Audio Format on the Basis of XML. In *Proceedings of the 2003 International Conference on Auditory Display*.

HUGHES, S., BRUSILOVSKY, P., AND LEWIS, M. 2002. Adaptive Navigation Support in 3D E-commerce Activities. In *Proceedings of Workshop on Recommendation and Personalization in eCommerce at the 2nd International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH'2002)*, 132–139.

KIM, H., JOSLIN, C., GIACOMO, T. D., GARCHERY, S., AND MAGNENAT-THALMANN, N. 2004. Adaptation Mechanism for Three Dimensional Content within the MPEG-21 Framework. In *Computer Graphics International 2004 (CGI'04)*, 462–469.

KLYNE, G., REYNOLDS, F., WOODROW, C., OHTO, H., HJELM, J., BUTLER, M., AND TRAN, L. 2003. *Composite Capability/Preference Profiles (*CC/PP*): Structure and Vocabularies*. W3C Working Draft.

LENZMANN, B., AND WACHSMUTH, I. 1996. A user-adaptive interface agency for interaction with a virtual environment. In *IJCAI '95: Proceedings of the Workshop on Adaption and Learning in Multi-Agent Systems*, Springer-Verlag, London, UK, 140–151.

SHEN, W. 1996. An efficient algorithm for incremental learning of decision lists. Tech. Rep. USC-ISI-96-012, Information Sciences Institute, University of Southern California.

UAPROF, 2001. Wireless Application Group: 'User Agent Profile Specification', WAP Forum (2001). http://www.wapforum.org/.