# A survey of plasticity in 3D user interfaces

Jérémy Lacoche, Thierry Duval, Bruno Arnaldi, Eric Maisel, Jérôme Royan

# A survey of plasticity in 3D user interfaces

Jérémy Lacoche* IRT b<>com
Thierry Duval † Université de Rennes 1 IRT b<>com
Bruno Arnaldi† INSA de Rennes IRT b<>com
Eric Maisel ‡ ENIB IRT b<>com
Jérôme Royan* IRT b<>com

**ABSTRACT**

Plasticity of 3D user interfaces refers to their capabilities to automatically fit to a set of hardware and environmental constraints. This area of research has already been deeply explored in the domain of traditional 2D user interfaces. Besides, during the last decade, interest in 3D user interfaces has grown. Designers find with 3D user interfaces new ways to promote and interact with data, such as e-commerce websites, scientific data visualization, etc. Because of the wide variety of Virtual Reality (VR) and Augmented Reality (AR) applications in terms of hardware, data and target users, there is a real interest in solutions for automatic adaption in order to improve the user experience in any context while reducing the development costs. An adaptation is performed in reaction to different criteria defining a system such as the targeted hardware platform, the user's context and the structure and the semantic of the manipulated data. This adaptation can then impact the system in different ways, especially content presentation, interaction techniques modifications and eventually the current distribution of the system across a set of available devices. We present the state of the art about plastic 3D user interfaces. Moreover, we present well known methods in the field of 2D user interfaces that could become relevant for 3D user interfaces. With this survey, we show that current solutions do not meet all plasticity requirements. That is why we propose an action plan to meet these requirements.

**Index Terms:** A.1 [Introductory and Survey]: ;— [H.5.1]: Information interfaces and presentation—Multimedia Information SystemsArtificial, augmented, and virtual realities; H.5.2 [Information interfaces and presentation]: User Interfaces—Graphical user interfaces (GUI)

## 1 INTRODUCTION

In the last years, interest in 3D user interfaces has grown. 3D user interfaces differ from traditional graphical user interfaces (GUI) by including a third dimension to present content and by using a larger range of interaction devices than the traditional set: mouse, keyboard and touch-screen. Indeed, the 3D interaction devices set is even larger. It includes devices like sensors, force feedback devices, head-mounted displays (HMD) or gamepads. Designers find with 3D user interfaces new ways to promote and interact with data, such as e-commerce websites, scientific data visualization, etc. Using 3D can improve attractiveness, interactivity and usability of these applications. This new trend is possible thanks to the improvement of graphics performance of devices like PCs or smartphones and also thanks to the generalization of VR and AR devices like the Microsoft® Kinect™, the Oculus Rift, the Google™Glass, etc.

In parallel, ubiquitous computing and continuity of access to information are widespread uses in everyday life. The acronym ATAWAD that means an access to information and services at AnyTime, AnyWhere and on Any Device is becoming a main interest in a context of mobility. Regarding the wide range of interaction devices for 3D user interfaces, we understand even better the real need to provide users with plastic interfaces.

During the development of 3D user interfaces, designers and developers have to handle a lot of input and output devices [3], a lot of interaction techniques [3] [25], a lot of possible kinds of target users [32], and a lot of ways to present content [14]. Developing manually a version of an application for each possible configuration is not a very flexible way toward adapting it to various features. Better solutions propose to adapt them automatically. In this case, it is referred to as plastic or adaptive 3D user interfaces. The goal is to preserve usability in any condition while minimizing development and maintenance costs [55]. Moreover, relevant adaptations can have a substantial impact on the interest of the user in an interactive system. Indeed, plasticity can bring a 3D user interface from an impersonal and maybe not optimally usable shape to a personalized one that will meet user's expectations.

Dealing with plasticity creates new challenges like hardware and users modeling. To continue, designing an intelligent adaptation engine that will find and perform the most suited modification of a system is another challenging step. Managing hardware is a really complex stage regarding ubiquitous computing and the growing number of new devices. Moreover, another challenge relates to the right time and the way to react to these new configurations of the system. As a simple example, if a user with a profile that contains a field "favorite color" navigates on an e-commerce website, we may only show him items of this color [13]. If the application allows adaptation at runtime, it is substantial to choose the good time for adaptations in order to avoid changing the application too often. The risk of too many modifications is to get the opposite by disturbing the user experience.

This paper provides a literature review that takes into account research work on plasticity and adaptive solutions for 3D user interfaces. We also consider the most relevant studies in plasticity for 2D user interfaces and adaptive hypermedia [5] that are more mature fields of research. This paper is structured as follows: in section 2.1 we recall the main definitions concerning plasticity and adaptive 3D user interfaces. Then, we identify adaptation sources, adaptation targets and adaptation time in section 2.2, 2.3 and 2.4 respectively. Next, in section 3, we list existing solutions for automatic adaptation of 3D user interfaces. To finish, in section 4 we discuss the limitations of current methods before concluding in section 5 and talking about our future work in section 6.

## 2 CONTEXT

In this section we detail the actual state of the plasticity concept. Therefore this section provides definitions of plasticity and related fields of research. Then plasticity issues are classified into three parts, adaptation sources, adaptation targets and the adaptation temporal dimension.

### 2.1 Definitions

Thevenin and Coutaz were the first to introduce the notion of plasticity [55]. This property of an interactive system refers to its capacity to withstand variations of both the system physical characteristics and the environment while preserving its usability. Plasticity

*{jeremy.lacoche,jerome.royan}@b-com.com
†{thierry.duval, bruno.arnaldi}@irisa.fr
‡e-mail:maisel@enib.fr

was already a well known problem in the field of software engineering. It included requirements like parametrization, customizability, configurability, user-adaptivity, reusability and portability. In the field of real-time interactive systems (RIS), the plasticity definition is adapted to cover the interaction problematic. The definition of Thevenin and Coutaz means that code interoperability is a necessary condition but is not sufficient for an interactive system to be considered as plastic. This notion of interoperability means that the system can be run on all considered platforms but does not ensure a good usability. That is why usability continuity has to be guaranteed too. Performances and possibilities have to be at least constant. One goal of plasticity is to specify once to serve multiple sources in order to minimize development and maintenance costs. This is done by performing adaptations on the system. These adaptations can be chosen by users from a predefined set of parameters: this is system adaptability. They can be chosen automatically by the software: this is system adaptivity. In the reference paper, plasticity is modeled on 3 axes which will be more developed in the three next subsections:

- **The adaptation sources:** point to the entities for which adaptation is intended. Three examples are given for these entities, users, environment and hardware characteristics.
- **The adaptation targets:** concern modifications applied by the system and the software components involved in adaptation. For example, it can refer to the application content or the interaction techniques.
- **The temporal dimension of adaptation:** the adaptation mechanism is static if it occurs between sessions and is dynamic if it can also occur at runtime.

Lindt et al. propose a complementary classification of adaptive user interfaces based on the temporal dimension and the adaptation controller (user or system) [39]. This classification is shown in table 1. The figure 1 summarizes the four dimensions of plasticity. Then, in the same paper, an adaptive user interface is defined as

| adaptation time (when) \ adaptation controller (who) | user | system |
|---|---|---|
| Compile time | configurable user interfaces | auto-configurable user interfaces |
| Runtime | adaptable user interfaces | adaptive user interfaces |

Table 1: Classification of user interfaces based on adaptation controller and adaptation time from [39]

any interface than can be classified in one of the four possible categories. Therefore we can say that an adaptive user interface can be considered as plastic as long as the adaptation is done in order to guarantee a constant user experience or to improve it while considering the current context. The definition should not just be limited to usability continuity but should also be extended to usability improvement.

Other fields of research are also concerned by these issues and can then be considered close to plasticity and adaptive user interfaces even if they do not mention it as such. First we can mention intelligent user interfaces which change their behavior to adapt to a person or a task [47]. As an example an autonomous agent may provide users with advice if it notices his difficulties to interact with the system. This kind of agent can be compared to the field of Intelligent Tutoring Systems (ITS). ITS aim to provide users with personalized instructions and feedbacks in a learning process. PEGASE [6] is an example of use of ITS in Virtual Reality. Next, we

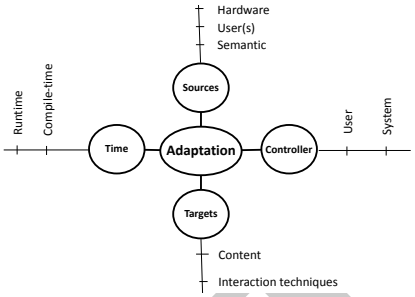


Figure 1: The four dimensions of the plasticity property

have adaptive hypermedia [4] that focuses on the adaptation of hypertext and hypermedia (including 3D [13]) systems to the user and to the environment [5] and could be therefore considered as plasticity. By building a user model according to his goals, preferences and knowledge, the system can adapt itself to his needs. Environment adaptation appeared in adaptive hypermedia with ubiquitous computing. The environment can refer to the user location or to its hardware platform. Therefore, we can compare this kind of adaptation with context aware applications [49] [10] which are systems that examine the computing environment (the context) and can react to its changes. As reported by Chen et al. [10], there are no unified definition for context but according to Dey et al. [17] context means any information that can be used to characterize the situation of something that is relevant to the interaction between a user and an application (person, place, object). As well as for plastic user interfaces, adaptive hypermedia and context aware applications are systems that examine adaptation sources to find the best adaptation targets in order to maximize the user experience.

In the same way as adaptive Hypermedia, in the web context, responsive web design appeared recently to help developers during websites creation in order to ensure their perfect usability and readability in a large range of platforms [40]. With CSS3 media queries, a website is able to adapt automatically its layout to the rendering device. The aim is to create and manage one version of a website to serve all the possible platforms such as desktop PCs, smartphones, tablets or connected TVs.

These definitions suit both 2D or 3D applications, nevertheless, the range of adaptation sources and targets increases significantly in the case of 3D applications. In fact, there is a wide difference between these two categories of user interfaces. While 2D user interfaces are mostly based on Windows, Icons, Menus, Pointers (WIMP) interaction devices like mouses and touch-screens, 3D user interfaces can use a larger range of interactions techniques, input and output devices and ways to present content. We can deduce that the combinatorial complexity is even larger for 3D user interfaces. Indeed, developing a code for all possibilities creates a large number of possibilities detailed in the following equation :

$$\begin{aligned} number\ of\ possible\ codes = {} & number\ of\ hardware\ platforms \\ & \times number\ of\ content\ presentations \\ & \times number\ of\ interactions\ techniques \end{aligned}$$

Plasticity aims to solve this combinatorial complexity.

### 2.2 Adaptation sources

As explained in the previous section adaptation sources refers to entities for which adaptation is intended. These entities must be relevant to the interaction between the system and the users to be taken into account. Thus, according to the context definition given in section 2.1, we can generalize by talking about context. An adaptation source refers to something that represents context for the application. First, these properties must to be modeled in the system. Secondly, they must be available to the adaptation mechanism. According to the literature we decomposed these sources into three

parts. Adaptation to hardware in section 2.2.1, adaptation to users in section 2.2.2 and adaptation to the semantic in section 2.2.3.

#### 2.2.1 Adaptation to hardware

There is a wide variety of platforms on which an application can be run and this list grows substantially when we talk about 3D. All platforms do not offer the same capabilities. For instance, without talking about code interoperability, it is obvious that we cannot run the same version of an application in an immersive cube than on a tablet for virtual reality, or on a tablet with camera than on see-through glasses for augmented reality. In both cases, setups are too different. Typically a platform is one or more computing units connected with input and output devices. This composition may change during runtime, some devices may be disconnected while others may be added. It is also important to notice that some devices like force feedback devices can be used as input as well as output. As an example of an output device, we can mention image display devices which can be characterized by their size, resolution, shape or refresh rate. The size property can be used to dynamically adapt the layout of an application in 2D [55] or in 3D [13]. Notice that output devices include any devices that return sensori-motor feedbacks (haptic devices, sound systems, etc). On the other hand, input devices allow the user to interact with the application in different ways. In this category we identify devices such as touch-screen, mouse, keyboard as well as 3D trackers, gaze trackers, speech recognizers, electroencephalogram (EEG) for brain computer interfaces, etc, which can be characterized by their quality, range, degree of freedom (DOF) or frequency. One critical point is to seamlessly adapt the application interaction techniques to these devices. Indeed, as detailed by Lindt in [38], exchanging an input device by another can result in an impractical interaction technique. Regarding the computing units of a system, there are other properties that can represent hardware constraints for an application, such as the bandwidth of the network or the computing resources available. For example a transfer of large amount of data coupled with a slow connection can affect the user experience. The network and the terminal capabilities can be taken into account to send an adaptable version of a 3D content using level of detail (LOD), such as a 3D agent [31], or a 3D model of a city [48]. More details about LOD representations will be given in section 2.3.2.

#### 2.2.2 Adaptation to the user or to the users (in case of collaboration)

In that case, we talk about personalization. Before being able to adapt the application to a user, we have to extract or learn from him a profile that contains his main characteristics such as his location, preferences, age, skill or habits what is called user modeling. A lot of research work have been published concerning user modeling [32]. According to Kobsa [33] a user model is defined as a set of information and assumptions on an individual user or on user groups that are used in an adaptation process. Sometimes we do not need a lot of information to perform adaptation. In [53], Sparacino represents a museum visitor just by one criteria among three (greedy, busy, selective). This value is then used to propose him the most suited visit program on a multimedia guide. In the 2D case, Motti et al. [42] compare different studies on interaction techniques with touch-screen for elderly people and demonstrate that age of users must be a source of adaptation. For example, elderly people will perform better if targets on a touchs-screen and spaces between them are bigger. Chittaro et al. [12] and Dos Santos et al. [18] present e-commerce applications where the personalization of the 3D worlds are made according to the detected user preferences. By tracking the user action, his profile with his preferences is edited and the visibility of his favorite products is increased. In some cases, it is also interesting to know more about the environment in which the user is located. We can consider various properties like the time, the lighting of the room, if the user is alone or not, and many other things that could be relevant for the application. The use of this kind of properties is cited by Chen et al. in a call forwarding process [10]. A user's phone is ringing in his office which is empty, consequently the system detects his location (position property) and automatically forwards the call to the nearby phone. In a second time, the system notices that the user is in a meeting, therefore the software forwards the call directly to the user's voice mailbox. Concerning 3D interactive systems, properties of the user environment can also be useful, e.g in an augmented reality application in which the lighting property of the real scene can be used to get more consistency between the real and the virtual world. This is done by Kanbara et al. in [30] where the evaluation of the real scene lighting is made with a mirror ball technique. The distribution of light sources in the real scene is estimated with the projected area of the mirror ball in the final image. This distribution is then used for the virtual object rendering.

#### 2.2.3 Adaptation to the semantic

In an application, the user will interact with data which have meanings represented by their semantic. There are two categories of semantic modeling according to Chevaillier et al. [11]. The first one is content-oriented semantic modeling which provides with a representation of an application content and that is commonly based on ontologies. The second one is system-oriented semantic modeling which aims to give meta-access to the features of the application entities. In most cases, application designers and developers use the content-oriented semantic even if it has not been specified to the system. For example, Walczak uses X-VRML for designing two different applications manipulating each one a particular kind of data, furniture for the first one and art artifacts for the other [9]. The designer chooses the way he will organize the two 3D scenes by using his semantic knowledge about data. That is why he chooses the museum representation for the artifacts and the shop for the furnitures. Therefore, in applications where semantic is specified to the system, it can be used as a source of automatic adaptation. For example, Hatala et al. use ontologies to match sounds from a database with museum artifacts and user preferences [26]. In the same way, Bilasco et al. uses semantic for object retrieval [1]. The reference paper describes rules that specify adaptations to 3D objects with a particular semantic. One of the given example consists in deteriorating geometries of "building" type objects. MASCARET [11] uses UML in order to model the semantic of 3D objects, procedures and behaviors in order to automatically generate interactive 3D worlds with learning scenarios. This knowledge database can then be used as a source of adaptation by 3D autonomous agents. The solution introduced by Wiebusch and Latoschik [57] presents concepts for decoupling in real-time interactive systems that supports hardware and component adaptations. The system uses a semantic description of components encoded as OWL-based models. This knowledge representation ensures reusability and portability and could be used for semantic adaptation.

### 2.3 Adaptation targets

Basically an adaptation mechanism takes account of sources to produce adaptation on some targets. These targets are part of the system and are modified in order to satisfy the adaptation mechanism request. The aim is to change the system in order to respect the plasticity property developed in section 2.1. We have classified the targets found in the literature into three groups. Interaction techniques modification in section 2.3.1, content adaptation in section 2.3.2 and redistribution in section 2.3.3.

#### 2.3.1 Interaction techniques modification

For 3D interactive systems, three kinds of interaction techniques are proposed by Hand [25]: objects manipulation, navigation and ap-

plication control. In each category, there are a lot of possible techniques that may perform differently depending on the context. In some circumstances it is possible that some interaction techniques cannot be used because of this context. In the paper already cited in section 2.2.1, Lindt [38] shows that selecting an object with the ray technique is possible with a hand tracker but not with a speech recognizer. If this last device is the only one available, the application will have to use another selection technique like speech commands. The interaction techniques selection depending on available devices is one of the aims of Grapple [22], by matching interactions techniques to device categories. Intrinsic parameters of interaction techniques can also be adapted. For example ICON [19] proposes to filter the movement of a cursor in order to stabilize it for people suffering from Parkinson's disease. Similarly, Schon et al. [50] change navigational keys sensitivity according to the user performance. If the user is getting lost in the 3D world this sensitivity is damped in order to improve his performance. While this method aims at facilitating navigation in 3D worlds, Celentano et al. propose to improve interactions with 3D objects [8]. Interaction techniques are modeled with finite state machines (FSM), then user activity is monitored in order to recognize interaction patterns. The final aim is to anticipate what the user wants to do and therefore shorten his interaction. The temporal Augmented Transition Network (tATN) introduced by Latoschik [34] goes further FSM by associating transitions with events, guarding constraints and temporal information. tATN are included into the framework presented in [35]. This framework enables designers to easily create multimodal interactions with context adaptation and hardware platform portability capabilities. Octavia et al. [43] consider both types of interaction techniques adaptation, parameters modification and techniques replacement and theses adaptations are made according to the user model.

#### 2.3.2 Content adaptation

An interactive system lets a user interact with some content that can be adapted. This content can be modified according to many ways like intrinsic parameters variations, quality improvement or degradation and content structure modifications. Global properties of the content are also concerned such as the sound volume delivered by the system or the brightness of a screen. Changing intrinsic parameters of some objects is one of the possibilities offered by Dachselt et al. [13], as already detailed in section 1, chair's color is set to the user's favorite color in one of the given examples. In the same way, in the e-commerce application [12] already cited in section 2.2.2, the size of user's favorite products is changed in order to get more visible. Visibility of important information is also the issue of Julier et al. [29] who aim at finding the correct opacity for each information display according to its consistency in the current context. Content quality can also be adapted such as an image resolution or the number of vertices and polygons of a 3D mesh. In such a case, we talk about level of detail (LOD). The aim of LOD is to improve or decrease the complexity of a content according to a metric. Creating these different levels of detail for a 3D content can be possible thanks to decimation algorithm such as the method proposed by Schroeder et al. [51]. This algorithm aims to reduce the number of triangles of a mesh while preserving its topology. Progressive meshes by Hoppe [28] are another way for dealing with LOD. It offers a continuous and lossless representation of a 3D mesh that handle different issues such as the smooth choice of LOD, progressive transmission and mesh compression. The metric taken into account in the LOD choice can be one of the adaptation sources detailed in section 2.2. This is the case for the two examples cited in section 2.2.1 where the metric is the hardware capabilities for adapting a 3D agent [31] and a 3D city model [48].

To finish, content structure modification is also considered in the literature. Adding or removing content can be performed like in [2] which fulfills virtual museum rooms only with 3D objects that match the user's preferences. It is also possible in 2D as in FlexClock [23] which displays a graphic clock and a calendar only if the window is large enough. Content structure also refers to the layout of the scene. For instance ENTER [24] can update a scene layout with functions like "Swap" "Move" or "Rotate" called on 3D objects by an adaptation engine which takes into account the user-profile. Layout adaptation is further addressed in [36] which proposes an automatic layout framework. By defining layout policies, a designer can manage a scene by taking into account different constraints like the user interactions.

#### 2.3.3 Redistribution

Redistribution consists in migrating the whole or a part of an application across different devices [7]. This may happen when the hardware configuration of the system changes during runtime. For example, CamNote [15] proposes a redistributable slides viewer application. In CamNote, if a PDA is available, a remote controller that allows navigation within the presentation can migrate to it. In the case of 3D user interfaces, redistribution as an adaptation target hasn't been well explored but we can find on the consumer market some solutions that already use it. This is the case in video games with the migration possibility offered by the Nintendo® Wii U™[1]. The migration is not automatically done but chosen by the user in the case that he does not or cannot play on his television anymore. In that case, the game totally migrates on the gamepad and the user can continue his game as he would have done on his television.

### 2.4 Temporal dimension

The last adaptation's dimension is the temporal one. Some solutions allow dynamicity, which means that adaptation can occur at runtime while others are just static. Static means that adaptation can just be done between sessions. Current input devices abstraction layers like VRPN [54] and OpenTracker [46] are only static as they do not allow to replace a device at runtime. Likewise X-VRML [9] or style-sheets from [21] provide only a solution for statically adapting the layout of 3D scenes. With these two solution designers can edit templates that will be chosen and parametrized before the session and that will not change until the next session. Conversely, applications like FlexClock as a 2D example [23], or solutions like the one proposed by Chittaro et al. [12] and AMACONT [13] in the case of 3D application, can automatically adapt themselves at runtime. Indeed, for instance, FlexClock and AMACONT allow a modification of the content presentation if the window size changes at runtime. Dynamic adaptation could become a main advantage for application as it improves the continuity of interaction when the system is modified in real time. For example, the apparition or the disappearance of an hardware component [15], a modification of the network quality [20] or a variation of the user-profile [12]. Nevertheless, adaptations at runtime have to be carefully performed because too much change can lead to inconsistency and thus disturb the user. This precaution is taken in the layout framework, presented by Lee et al. [36], by not considering, in the space allocation algorithm, 3D objects that may change too often.

## 3 Solutions

This section provides the solutions found in the literature that deal with the plasticity property. Solutions are classified into three groups according to the adaptation sources they handle : hardware adaptation in section 3.1, user adaptation in section 3.2 and semantic adaptation in section 3.3. In each group, we try to gather the solutions into subcategories according to the implementation of the adaptation mechanism.

[1] http://www.nintendo.com/wiiu/features/ Date of access : january 2014

### 3.1 Hardware adaptation

Concerning hardware adaptation, device abstraction layers allow designers to handle a wide variety of input and output devices. A device abstraction layer classifies devices into different classes based on what data they can provide such as 3-DOF or 6-DOF trackers. This classification is made in order to hide the complexity of concrete devices. Moreover, this approach allows an application to accept the replacement of a device by another one of the same type. OpenTracker [46] and ICON [19] already cited in section 2.3.1, are two abstraction layers based on dataflow architectures which can combine and filter input devices in order to connect them to different actions of an application. ICON has the advantage over OpenTracker to be dynamic, as the reconfiguration can be done at runtime. ICON also provides with a graphical user interface in order to configure the system. Unfortunately, it does not handle natively any VR devices. Conversely, OpenTracker has been designed to handle trackers but the configuration has to be made before execution with a XML file. Grapple [22] also offers a mechanism for static adaptation by classifying input devices into categories and by providing an abstract representation for each category. A developer can then match interaction techniques to device category, then at runtime these techniques can work with any device of the associated categories. VRPN is another static solution [54] which proposes a client-server architecture with a network-transparent interface to a wide variety of peripherals like trackers, buttons or force feedback devices. In the same way DEVAL [44] introduces a generic device abstraction layer that defines device classes and structures them hierarchically. DEVAL supports dynamicity with devices addition and replacement at runtime. Viargo [56] is another dynamic solution. Input devices are abstracted by device units like in VRPN which provide events to interaction metaphor components. Then, these components process the events in order to update the state system of the application. If a device is exchanged at runtime, the interaction metaphor is not disturbed while the new device events are compatible with it. To finish with abstraction layers, MiddleVR [2] offers to handle transparently a lot of input devices including those from VRPN. Moreover, unlike other solutions, MiddleVR can also deal with output devices and clustering. For instance, it is possible to configure its own screen setup. This setup contains parameters such as the number of screens, how they are placed, stereoscopic properties and size and resolution of each screen. Moreover, this setup also includes association between computers and screens. These configurations features allow a developer to deal with a wide variety of display devices, from simple monitors to immersive cubes and workbenches. This configuration is done with a graphical interface and can be changed at any time during execution.

The problem with abstraction layers is that devices are just defined as input and output data while it would be interesting to also know their physical properties in the real physical workspace or their internal properties like refresh rate or accuracy. The device model proposed by Lindt [39] extends DEVAL [44] to add this kind of meta-data. Three kinds of meta-data can be added to a device instance, first static devices properties that do not change over the time like the weight of a HMD or the position of a device in the real world. Next, configurable devices properties that depend on the device setup like the smoothing factor of a tracking device. To finish, runtime properties which include performances and device states. This device model is integrated into a more complex architecture in the CATHI framework (Context-Adaptive Three-Dimensional Interaction) designed for the creation of 3D adaptive user interfaces. With a rule-based system and a rating mechanism, the model is able to build the most suited 3D user interfaces represented by a set of interconnected components. The framework supports user and auto-configuration, dynamicity and can impact the main adaptation targets that we cited in section 2.3.

To be able to deal with a lot of devices is a kind of hardware adaptation but when the hardware is known, a system can also deduce some other modifications to perform on itself. Using rule-based systems is one of the solutions proposed in the literature. AMACONT [27] offers this kind of solutions like in [13] which shows an example of output devices adaptation with a choice of a 3D menu type according to the screen size as detailed in section 2.4. In that case, a rule set the screen size threshold that will lead to the modification of the 3D menu.

In the field of 2D user interfaces, some solutions are based on model driven engineering, which has not been well explored for 3D. It consists in modeling and decomposing a user interface with a high level language. This kind of framework is used in [52] or in [55] which represent hardware with a context or platform model. This platform model is then processed by the system in order to choose adapted interactors and an adapted way to display content. A similar approach is used in CamNote [15], already cited in section 2.3.3, to detect if a PDA is available in order to migrate on it a remote controller for the application.

### 3.2 User adaptation

As explained in the second section, an interactive system can be personalized according to a user and to the context in which he is located. User modeling is the entry point for this adaptation mechanism. We can find in the literature different methods that take into account this adaptation source.

First, similarly to hardware adaptation we can cite rule-based solutions. In that case, a designer will associate different adaptations rules to an application. These rules specify reactions to a particular state of the user model. For example, with AMACONT [13], a designer can write XML rules which have access to the user profile. The example cited in the paper, already explained in section 2.3.2, is a rule that changes the color of all chairs in a 3D scene by the favorite color of the current user in the context of e-commerce. Chittaro et al. also use rules that make requests on the user profile to change the content of a virtual shop [12]. This is an example of rule proposed in the paper :

**IF** seen(X)=0 **AND** NumberOfVisits>3 **THEN**
goal(IncreaseExposureLevel(X))

This rule asks the system to increase the visibility of a product X in a 3D world if the current visitor came more than 3 times and if he has never seen the product. This is achieved by generating a VRML world fulfilled with VRML "PROTOs" instantiated with the parameters required by the personalization choices. Increasing visibility of an object could have an impact on its scale in the VRML model for example. The solution proposed by Dennemont et al. [16] also offers a rule-based system for adaptation to the user and his environment. A reasoning engine processes the rules and the application knowledge, represented with conceptual graphs, in order to find adaptations. The given scenario tries to enhance user's interests with content and interactions adaptations. Nevertheless, in its current state, the delay of the engine can just allow punctual helps.

Next, rating systems are also a possible solution for user adaptation. Bonis et al. in [2] use a tree to represent the user model where nodes are objects or objects categories and edges represent relations between them. Each node has an associated score which is the user interest for the object or the category. Scores are updated by an algorithm that takes into account user interactions in the 3D world. User's favorite products will get better scores. Finally, the main virtual room is fulfilled with objects with the best scores and some doors give access to rooms with the remaining objects of the

[2] http://www.imin-vr.com/middlevr-for-unity/ Date of access : january 2014

database. This methods can handle more than one user by using a clustering algorithm in order to create groups of users with similar preferences.

As detailed in section 3.1 the model proposed by Lindt [39] uses a combination between a rating mechanism and a rule-based system in order to perform adaptation to hardware. Moreover, this solution also includes adaptation to the user and his associated context. For example, a user can write his own adaptation rules that will be taken into account during the adaptation process. However, even if the framework could be extended with such a component, in its current state no user model with user monitoring is integrated.

Then some solutions for user adaptation are based on mutli-modules architecture where each module has its own task in the adaptation process and communicates with the others such as ENTER [24] with its multi-agent architecture. User modeling is done with three agents, a login agent for initialization, a listener agent to monitor user activities and an analysis agent to extract user preferences. Thus, the presentation agent is informed by the analysis agent of user preferences and displays an adapted world. The conceptual framework of Octavia et al.[43] is quite similar to ENTER. Indeed, it connects some modules with different roles and takes into account the user and the current task. However, the goal is different as the adaptation target is not the 3D world but interaction techniques. Like in ENTER, the framework is made up of a set of intercommunicating modules : the monitoring module to analyze user activities ; a knowledge base that stores the past adaptations and interactions ; a user model ; an adaptation engine which determines with the help of the previous three modules how to adapt interactions techniques. It can go from modifying an intrinsic parameter of an interaction technique such as sensitivity to the replacement thanks to a more adapted technique. Also, the AdapTIVE architecture of Dos Santos et al. [18] is made of a set of modules such as the user model manager that monitors users and updates users models. The content manager is responsible of content addition, removal and classification as well as the environment generator that creates the suited 3D world according to content and user models. The given example is a virtual shop where 3D rooms contain products. Rooms are ordered according to the user interest, the one that matches at best becoming more accessible.

Then, using machine learning is another way for user adaptation. Sto(ry)chastics [53] is based on this approach and uses a Bayesian network to estimate the user models and to perform content selection. Concerning machine learning in 3D user interfaces, Lenzmann et al. propose to use reinforcement learning in order to perform user interface customization, according to user preferences, for manipulating 3D objects in a virtual environment [37].

### 3.3 Semantic adaptation

An interactive system, especially a 3D user interface, is designed to allow a user to navigate into and interact with a set of data. These data have a semantic that is taken into account at the design step by a developer or a designer. The example cited in the section 2.3 was the use of 3D templates in order to take this semantic into account. Some solutions go further by modeling this semantic and then use it to adapt a user interface accordingly. However, semantic adaptation has not been as well explored as hardware and user adaptation.

As for user adaptation, there are some rating system based approaches. The solution proposed by Julier et al. [29], previously cited in section 2.3.2, aims to perform information filtering in an augmented reality application in order to only show the most important information to the user. Authors propose to model the object's semantic in the context of the application's scenario. An object is modeled with objective properties such as shape, type or location and subjective properties represented by an importance vector. This vector contains a score of importance from 0 (irrelevant) to 1 (highly relevant) for each task of the scenario. This scoring is done by domain experts before runtime. At runtime depending on the user position, his current task and the set of importance vector, for each object the rating system updates a score from 0 to 1 that will be the object transparency on the screen in order to make objects fade in and fade out according to their importance for the current context. In the Web3D context, AVE - Adaptive Visualization Environment [9] also uses a rating system. In a document retrieval process, the aim is to choose automatically a good visualization metaphor for the search result according to its semantic properties. The interface selection process first determines a set of interfaces suited for the data structure. Then, for each of these interfaces a ranking coefficient is computed according to the semantic properties of the search result. At the end, the interface with the minimum coefficient is selected. If two or more interfaces get the same score, the choice is given to the user.

Rule-based systems are also present for semantic adaptation. Bilasco et al. [1], detailed in section 2.2.3, offer to choose the adapted LOD for 3D objects according to their semantic importance in the current context. An object with a low importance in the context of the application can then be replaced by its bounding box or can be excluded on devices with low capacities.

Another solution to deal with semantic is proposed by Moreno et al. in a fire fighting simulation system [41]. Authors propose to integrate a semantic layer, composed of a knowledge base and a reasoner, in the system architecture. The knowledge base compiles emergency procedures and concept, geographic data and user categorization into a set of ontologies. Besides, the reasoner sends signals to the virtual reality simulator in order to produce modifications. An example given in the paper is semantic highlighting which consists in changing visual parameters of some objects to make them more distinguishable. For instance, the semantic layer can give the instruction to highlight a burning building.

## 4 Discussion

In this survey, we aim to show that a lot of solutions dealing with the plasticity property for 3D user interfaces exist in the literature. However, all these solutions do not offer the same possibilities in terms of adaptation targets, adaptation sources and dynamicity.

We have seen that some solutions can take into account different adaptation sources, this is the case for AMACONT [13] or the model proposed by Lindt [39] that handle hardware adaptation as well as user adaptation. However, AMACONT does not include an abstraction layer for input devices and has been designed for Web3D. Regarding the other solutions, the vast majority is only concerned with one adaptation source. Moreover, our survey shows that taking data semantics and collaboration as adaptation sources is not as deeply explored as taking hardware and individual users as sources.

Regarding adaptation targets, similarly as for adaptation sources, most solutions handle only one of the targets. Once again, AMACONT [13] and the model proposed by Lindt [39] are two counterexamples because they can adapt content as well as interaction techniques but they do not handle redistribution. Indeed, redistribution is the less explored target in the literature.

As detailed, a system is considered as static if adaptations need restarting the system or worth recompiling it. A system is considered as dynamic if adaptations can occur at runtime. In the literature we can find both types but choosing a static only solution may be a disadvantage for an application. Indeed, every system can vary at runtime. For instance, devices may be replaced, the network may slow down, users may arrive and leave. Theses variations may force to reconsider the set of adaptations chosen at initialization and may make the application inconsistent with the context.

We explained that adaptations can be determined sometimes by the system, sometimes by the user and sometimes it can be a mix between both, what should be possible in a perfect scenario. How-

ever, if automatic adaptations may happen, according to Lindt [39], the user must feel that he has the control of the system, see what is going on, be able to reverse adaptations, and the adaptations must preserve the consistency of the 3D user interface.

AMACONT [13] and the CATHI framework [39] seem to be the most complete solutions of the literature. Both support multiple adaptation sources, multiple adaptation targets, dynamicity and user and auto-configuration. However, AMACONT is especially designed for Web3D, and CATHI has a strong dependence to the VR/AR framework MORGAN [45] for 3D rendering and tracking device handling. Moreover, none of them take into consideration issues like collaboration, redistribution or semantic adaptation. CATHI aims to provide an extensible framework, therefore it could be improve to handle these issues.

This survey also explores the field of 2D user interfaces. In this field of research, model based driven engineering seem to be the most suited technique to build plastic user interfaces. However, model based driven engineering lacks authoring tools for developing applications. Thus, today, solutions based on this concept are not widely used by developers. Moreover, in the field of 3D, model based driven engineering has not been deeply explored. Some solutions exist to generate 3D applications based on models but they do not take into account any adaptation concept.

With these previous affirmations, we can highlight the fact that in the field of adaptive or plastic 3D user interfaces there is a lack of a unified solution that could deal easily with all adaptation sources, all adaptation targets, dynamicity, user configuration and auto-configuration. Providing developers and designers with this kind of solution could really help them in the plastic 3D user interfaces development process. This is true only if an authoring tool to ease the development of such applications is available.

## 5 Conclusion

The need for plasticity is becoming increasingly important. Thus, in this paper, we have presented an overview of current techniques for designing plastic 3D user interfaces. We have shown that current methods offer adaptation to different constraints such as hardware, user and semantic in order to maximize the usability of applications. This goal can be reached with different kinds of adaptation which are: content modification, redistribution or interaction techniques revision. Adaptations are either static or dynamic. They can be performed by human requests from a predefined set of parameters and we talk of adaptability, or automatically by the system, in this case we talk of adaptivity.

However we have seen that no solution meets all the requirements of plasticity. We need a unified solution that takes into account all adaptation sources, all adaptation targets, that supports dynamicity and adaptivity and that allow user and auto-configuration. Even if model-based user engineering aims to provide a generic solution for plasticity in 2D, it is still too badly accepted by developers and is not really well explored for 3D applications. Moreover, collaboration, which could also be an adaptation source, has not been well explored yet. It is interesting to wonder how to adapt a shared application by users with different sets of constraints: for example, modifying the content differently for multiple users could alter their collaboration.

## 6 Future work

These shortcomings in current solutions will drive our future work. We will focus on finding generic and scalable methods to deal with all adaptation issues. Our aim is to meet all the requirements of the plasticity property including those not well explored today.

First, as CVE is an important part of 3D user interfaces, it is substantial to consider collaboration in our future work. Thanks to the wide variety of interaction devices a lot of collaboration scenarios are possible. In all possible scenarios, it is important to handle awareness issues. Users have to be aware of the others interaction possibilities and of how they see the content. This awareness is important, especially when all users applications are not adapted in the same way.

Secondly, modeling the content oriented and the system oriented semantic in an application has to be considered. This semantic could be really useful to create dynamic scenes adapted for the content. For example, in a virtual museum context, it would be really interesting to create a museum template that adapts itself to the artifacts it contains. It would allow to provide a futuristic version of the museum for science fiction objects and a castle version for Middle Ages artifacts.

To finish, we aim to propose a flexible solution for developers but also easy to use for designers. This solution will be designer friendly if an authoring tool is available. Indeed, we think that providing an authoring tool is the only way to bring such a solution on the consumer market.

To achieve it, our road map for our future work will begin by finding the best way to model hardware, user and semantic. Then we will define an adaptation model which will take into account all our issues. To finish, we will provide developers with an authoring tool for creating plastic 3D user interfaces.

## References


[1] I. M. Bilasco, M. Villanova-Oliver, J. Gensel, and H. Martin. Semantic-based rules for 3D scene adaptation. In *Proceedings of the twelfth international conference on 3D web technology*, pages 97–100, 2007.

[2] B. Bonis, J. Stamos, S. Vosinakis, I. Andreou, and T. Panayiotopoulos. A platform for virtual museums with personalized content. 42:139–159, 2009.

[3] D. A. Bowman, E. Kruijff, J. J. LaViola, and I. Poupyrev. *3D User Interfaces: Theory and Practice*. Addison Wesley Longman Publishing Co., Inc., 2004.

[4] P. Brusilovsky. Methods and techniques of adaptive hypermedia. *User modeling and user-adapted interaction*, 6(2-3):87–129, 1996.

[5] P. Brusilovsky. Adaptive hypermedia. 11:87–110, 2001.

[6] C. Buche, C. Bossard, R. Querrec, and P. Chevaillier. PEGASE: A generic and adaptable intelligent system for virtual reality learning environments. *International Journal of Virtual Reality*, 9(2):73–85, Sept. 2010.

[7] G. Calvary, J. Coutaz, O. Dâassi, L. Balme, and A. Demeure. *Towards a new generation of widgets for supporting software plasticity: the" comet"*, pages 306–324. Springer, 2005.

[8] A. Celentano, M. Nodari, and F. Pittarello. Adaptive interaction in Web3D virtual worlds. In *Proceedings of the Ninth International Conference on 3D Web Technology*, Web3D '04, pages 41–50. ACM, 2004.

[9] W. Cellary and K. Walczak. *Interactive 3D MultiMedia Content*. Springer, 2012.

[10] G. Chen, D. Kotz, and D. Kotz. A survey of context-aware mobile computing research, 2000.

[11] P. Chevaillier, T.-H. Trinh, M. Barange, P. De Loor, F. Devillers, J. Soler, and R. Querrec. Semantic modeling of virtual environments using MASCARET. In *Software Engineering and Architectures for Realtime Interactive Systems (SEARIS), 2012 5th Workshop on*, pages 1–8, 2012.

[12] L. Chittaro, R. Ranon, I. S. Artificial, and V. Realities. Dynamic generation of personalized VRML content: a general approach and its application to 3D e-commerce. In *In Proceedings of Web3D 2002: 7th International Conference on 3D Web*, pages 145–154. Press, 2002.

[13] R. Dachselt, M. Hinz, and S. Pietschmann. Using the AMACONT architecture for flexible adaptation of 3D web applications. In *Proceedings of the Eleventh International Conference on 3D Web Technology*, Web3D '06, pages 75–84. ACM, 2006.

[14] R. Dachselt and A. Hübner. Virtual environments: Three-dimensional menus: A survey and taxonomy. *Comput. Graph.*, 31(1):53–65, Jan. 2007.

[15] A. Demeure, L. Balme, and G. Calvary. CamNote: A plastic slides viewer. Plastic Services for Mobile Devices (PSMD), Workshop hel in conjunction with Interact'05, Rome, 12 Septembre 2005., 2005.

[16] Y. Dennemont, G. Bouyer, S. Otmane, and M. Mallem. 3D interaction assistance in virtual reality: a semantic reasoning engine for context-awareness. In *Context-Aware Systems and Applications*, page 30–40. Springer, 2013.

[17] A. K. Dey. Understanding and using context. 5:4–7, 2001.

[18] C. T. dos Santos and F. S. Osorio. Adaptive: An intelligent virtual environment and its application in e-commerce. In *Computer Software and Applications Conference, 2004. COMPSAC 2004. Proceedings of the 28th Annual International*, pages 468–473. IEEE, 2004.

[19] P. Dragicevic and J.-D. Fekete. *Input device selection and interaction configuration with ICON*, pages 543–558. Springer, 2001.

[20] T. Duval and C. e. Zammar. A migration mechanism to manage network troubles while interacting within collaborative virtual environments. In *Proceedings of the 2006 ACM international conference on Virtual reality continuum and its applications*, pages 417–420. ACM, 2006.

[21] N. Esnault, J. Royan, R. Cozot, and C. Bouville. A flexible framework to personalize 3d web users experience. In *Proceedings of the 15th International Conference on Web 3D Technology*, Web3D '10, pages 35–44, New York, NY, USA, 2010. ACM.

[22] M. Green and J. Lo. The grappl 3D interaction technique library. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*, VRST '04, pages 16–23. ACM, 2004.

[23] D. Grolaux, P. V. Roy, and J. Vanderdonckt. Qtk: An integrated model-based approach to designing executable user interfaces. In *DEPT. OF COMPUTER SCIENCE, UNIV. OF GLASGOW*, pages 77–91. Springer Verlag, 2001.

[24] T. Guinan, C. O'Hare, and N. Doikov. Enter: The personalisation and contextualisation of 3-dimensional worlds. In *Parallel and Distributed Processing, 2000. Proceedings. 8th Euromicro Workshop on*, pages 142–148, 2000.

[25] C. Hand. A survey of 3d interaction techniques. In *Computer graphics forum*, volume 16, pages 269–281. Wiley Online Library, 1997.

[26] M. Hatala, L. Kalantari, R. Wakkary, and K. Newby. Ontology and rule based retrieval of sound objects in augmented audio reality system for museum visitors. In *Proceedings of the 2004 ACM symposium on Applied computing*, pages 1045–1050, 2004.

[27] M. Hinz and Z. Fiala. AMACONT: a system architecture for adaptive multimedia web applications. In *Berliner XML Tage*, pages 65–74, 2004.

[28] H. Hoppe. Progressive meshes. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 99–108. ACM, 1996.

[29] S. Julier, M. Lanzagorta, Y. Baillot, L. Rosenblum, S. Feiner, T. Hollerer, and S. Sestito. Information filtering for mobile augmented reality. In *Augmented Reality, 2000.(ISAR 2000). Proceedings. IEEE and ACM International Symposium on*, pages 3–11, 2000.

[30] M. Kanbara and N. Yokoya. Geometric and photometric registration for real-time augmented reality. In *Mixed and Augmented Reality, 2002. ISMAR 2002. Proceedings. International Symposium on*, pages 279–280. IEEE, 2002.

[31] H. Kim, C. Joslin, T. Di Giacomo, S. Garchery, and N. Magnenat-Thalmann. Adaptation mechanism for three dimensional content within the mpeg-21 framework. In *Computer Graphics International, 2004. Proceedings*, pages 462–469, 2004.

[32] A. Kobsa. User modeling: Recent work, prospects and hazards. *Human Factors in Information Technology*, 10:111–111, 1993.

[33] A. Kobsa. Supporting user interfaces for all through user modeling. *Advances in Human Factors/Ergonomics*, 20:155–157, 1995.

[34] M. E. Latoschik. Designing transition networks for multimodal VR-Interactions using a markup language. In *Proceedings of the 4th IEEE International Conference on Multimodal Interfaces*, ICMI '02, page 411–, Washington, DC, USA, 2002. IEEE Computer Society.

[35] M. E. Latoschik. A user interface framework for multimodal VR interactions. In *Proceedings of the 7th International Conference on Multimodal Interfaces*, ICMI '05, page 76–83, New York, NY, USA, 2005. ACM.

[36] W. L. Lee and M. Green. A layout framework for 3D user interfaces. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*, VRST '05, pages 96–105. ACM, 2005.

[37] B. Lenzmann and I. Wachsmuth. A user-adaptive interface agency for interaction with a virtual environment. In *Adaption and Learning in Multi-Agent Systems*, pages 140–151. Springer, 1996.

[38] I. Lindt. Exchangeability of 3d interaction techniques. In *Proceedings of the IEEE Workshop on New Directions in 3D User Interfaces*, pages 93–94, 2005.

[39] I. Lindt. *Adaptive 3D-User-Interfaces*. PhD thesis, 2009.

[40] E. Marcotte. Responsive web design. *A list apart*, 306, 2010.

[41] A. Moreno, S. Zlatanova, B. Bucher, J. Posada, C. Toro, and A. García-Alonso. Semantic enhancement of a virtual reality simulation system for fire fighting. In *Proceedings of the Joint ISPRS Workshop on 3D City Modelling & Applications and the 6th 3D GeoInfo Conference*, 2011.

[42] L. G. Motti, N. Vigouroux, and P. Gorce. Interaction techniques for older adults using touchscreen devices : a literature review. In *25ème conférence francophone sur l'Interaction Homme-Machine, IHM'13*. ACM, 2013-09.

[43] J. R. Octavia, C. Raymaekers, and K. Coninx. A conceptual framework for adaptation and personalization in virtual environments. In *Database and Expert Systems Application, 2009. DEXA'09. 20th International Workshop on*, pages 284–288. IEEE, 2009.

[44] J. Ohlenburg, W. Broll, and I. Lindt. *DEVAL-A Device Abstraction Layer for VR/AR*, pages 497–506. Springer, 2007.

[45] J. Ohlenburg, I. Herbst, I. Lindt, T. Fröhlich, and W. Broll. The morgan framework: enabling dynamic multi-user ar and vr projects. In *Proceedings of the ACM symposium on Virtual reality software and technology*, pages 166–169. ACM, 2004.

[46] G. Reitmayr and D. Schmalstieg. An open software architecture for virtual reality interaction. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*, VRST '01, pages 47–54, New York, NY, USA, 2001. ACM.

[47] E. Ross. Intelligent user interfaces: Survey and research directions. *University of Bristol, Bristol, UK*, 2000.

[48] J. Royan, P. Gioia, R. Cavagna, and C. Bouville. Network-based visualization of 3d landscapes and city models. *IEEE Comput. Graph. Appl.*, 27(6):70–79, Nov. 2007.

[49] B. Schilit, N. Adams, and R. Want. Context-aware computing applications. In *Mobile Computing Systems and Applications, 1994. WMCSA 1994. First Workshop on*, pages 85–90, 1994.

[50] B. Schön, C. O'Hare, B. R. Duffy, A. N. Martin, M. Bradley, and J. F. An agent based approach to adaptive navigational support within 3D-environments. IEEE, 2004.

[51] W. J. Schroeder, J. A. Zarge, and W. E. Lorensen. Decimation of triangle meshes. In *ACM SIGGRAPH Computer Graphics*, volume 26, pages 65–70. ACM, 1992.

[52] M. Sendín and J. Lorés. Plastic user interfaces: Designing for change. In *MBUI*, 2004.

[53] F. Sparacino. Sto (ry) chastics: a bayesian network architecture for user modeling and computational storytelling for interactive spaces. In *UbiComp 2003: Ubiquitous Computing*, pages 54–72, 2003.

[54] R. M. Taylor, II, T. C. Hudson, A. Seeger, H. Weber, J. Juliano, and A. T. Helser. Vrpn: A device-independent, network-transparent vr peripheral system. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*, VRST '01, pages 55–61, New York, NY, USA, 2001. ACM.

[55] D. Thevenin and J. Coutaz. Plasticity of user interfaces: Framework and research agenda. In *Proceedings of INTERACT*, volume 99, pages 110–117, 1999.

[56] D. Valkov, B. Bolte, G. Bruder, and F. Steinicke. Viargo - a generic virtual reality interaction library. In *Software Engineering and Architectures for Realtime Interactive Systems (SEARIS), 2012 5th Workshop on*, pages 23–28, Mar. 2012.

[57] D. Wiebusch and M. Latoschik. Enhanced decoupling of components in intelligent realtime interactive systems using ontologies. In *Software Engineering and Architectures for Realtime Interactive Systems (SEARIS), 2012 5th Workshop on*, pages 43–51, Mar. 2012.