Universidad Carlos III De Madrid

PhD Dissertation

Integration of Accessibility Requirements in the Design of Multimedia User Agents Interfaces

Author: María González García

Advisor: Lourdes Moreno López

PhD Program in Computer Science and Technology

Computer Science Department

Leganés, October 2015

DOCTORAL THESIS

Integration of Accessibility Requirements in the Design of Multimedia User Agents Interfaces

Author: María González García

Advisor: Lourdes Moreno López

Signature from PhD committee:

Signature

President:

Vocal:

Secretary:

Grade:

Leganés, 30 October 2015

Aunque parezca mentira, todo llega a su fin

Agradecimientos

A ti, mi marido, por estar siempre a mi lado. Gracias por tu constancia, por tu disposición, por tu apoyo, por no dejarme caer, por tu ayuda infinita. Eres mi complemento perfecto, de ti aprendo algo cada día. Sin ti esto no habría sido posible.

A mis padres, por su esfuerzo, por su dedicación, por apoyarme. Gracias por todo lo que me habéis enseñado. Sin vosotros tampoco habría sido posible llegar hasta aquí.

A mi familia, por estar siempre ahí. Gracias por todos los momentos que hemos pasado, esas comidas, esas cenas, esas fiestas, esas "reuniones" imprevistas en las que de repente se llenaba la casa,…

A mis amigos, por todos esos ratos juntos. No tendría espacio suficiente en este documento para nombraros a todos y deciros por qué sois especiales para mí, por eso, solo puedo deciros "¡Gracias por formar parte de mi vida y dejarme ser parte de la vuestra!"

A todos los que me han ayudado con las evaluaciones, la mayoría amigos, muchas gracias por vuestra disposición.

Por último, no quiero olvidarme de los compañeros que he tenido en el grupo LaBDA, a los que considero mis amigos. Gracias a Paloma por estar siempre ahí, por hacer fácil lo que parece difícil, a Isa por todos sus consejos y todos los momentos que hemos pasado juntas, a Ricardo (nuestro "Becary") por sus revisiones, su alegría y su disposición, a José, a Harith, a Loli, a Yolanda, a Víctor, a Pilar,... Especialmente, quiero agradecerle a mi tutora, Lourdes, que me haya dado la posibilidad de realizar este trabajo y conocer el mundo de la accesibilidad. Gracias Lourdes por haberme acompañado en este camino, por estar ahí, por todo lo que me has enseñado, pero sobre todo por todo lo que he aprendido de ti y contigo.

Resumen

El continuo incremento del contenido multimedia en la Web, especialmente del contenido vídeo, no va acompañado de un incremento similar de accesibilidad, hay una falta de alternativas sincronizadas al contenido como subtitulado, audiodescripción, etc., que permitan acceder a cualquier persona con y sin discapacidad a dicho contenido.

Esta falta de accesibilidad en el acceso al contenido vídeo no solo se debe a la ausencia de alternativas, también es debido a que los agentes de usuario que entregan dicho contenido no proporcionan los medios necesarios para presentarlas.

Este hecho da lugar a que no se cumpla la normativa y la legislación vigente en materia de accesibilidad. Dicho incumplimiento, puede ser debido al desconocimiento, o a que aplicar esa normativa desde el punto de vista de la ingeniería no es trivial.

Hay una falta de herramientas de autor y de enfoques metodológicos que asistan en el desarrollo de un producto accesible en el ámbito de la Ingeniería, como es el caso del desarrollo de un agente de usuario con calidad que incluya requisitos de accesibilidad.

Todos estos hechos, el incremento progresivo del contenido multimedia en la Web, las barreras de accesibilidad tanto en el contenido como en el agente de usuario junto con la normativa y legislación vigente en materia de accesibilidad es lo que ha motivado la realización de esta Tesis Doctoral.

Con esta Tesis Doctoral se proporciona el conjunto de requisitos de accesibilidad que debe cumplir un agente de usuario que sirva contenido multimedia accesible. Además se proporciona un espacio de trabajo siguiendo un enfoque metodológico que asista en el diseño y desarrollo de la interfaz de un agente de usuario accesible que sirve contenido multimedia accesible. Este espacio de trabajo está compuesto de una arquitectura y modelos siguiendo el enfoque de Model-Based User Interface Development (MBUID) y está orientado a ser utilizado por diseñadores con conocimientos en modelado. Por último, como recurso de ayuda a cualquier profesional, independientemente de sus conocimientos en modelado y accesibilidad, se ofrece una herramienta de autor basada en modelos para crear agentes de usuario con requisitos de accesibilidad.

Abstract

The continuous increase of multimedia content in the Web, especially video content, is not accompanied by a similar increase of accessibility; there is a lack of synchronized alternatives for the content such as captions, audio description, etc. that allow anyone with or without disability to access such content.

This lack of accessibility in video content access is not only due to the lack of alternatives, but also because of the fact that user agents which deliver this content do not provide the necessary means to present them.

This fact leads to the noncompliance of the current regulations and legislation in terms of accessibility. This noncompliance could be due to the lack of knowledge, or because of the fact that applying these regulations from an engineering point of view is not trivial.

There is a lack of authoring tools and methodological approaches which assist in the development of an accessible product in the Engineering scope as it is the case of the development of a quality user agent which includes accessibility requirements.

All these facts, multimedia content"s progressive increase on the Web, accessibility barriers both in the content and in the user agent together with current regulations and legislation regarding accessibility is what has motivated the accomplishment of this Doctoral Thesis.

With this Doctoral Thesis, a set of accessibility requirements that a user agent which delivers multimedia content must fulfil is provided. Besides, a workspace is provided following a methodological approach which assists in the design and development of the interface of an accessible user agent which delivers accessible multimedia content. This workspace is composed of an architecture and models following a Model-Based User Interface Development (MBUID) approach and is oriented to be used by designers with knowledge in modeling. Finally, as a support to any professional regardless of their knowledge in modeling and in accessibility, an authoring tool based on models is offered in order to create user agents with accessibility requirements.

Contents

List of Figures

List of Tables

Chapter 1. Introduction

1.1. Introduction

Multimedia content on the Web is being increased at a staggering rate. For instance, regarding Cisco [1], online video users are expected to double to 1.5 billion in 2016 and, globally, online video traffic will be 55 percent of all consumer Internet traffic in 2016 [2].This increase is due to Web 2.0, social media or the digitalization of several resources such as books or newspapers. Apart from this, there are other surveys which show the importance of the video content. For example, video statistics provided in [3] and the video monetization report provided in [4].

Unfortunately, multimedia content increase does not lead to increase the accessibility within this content. Moreover, many user agents which provide multimedia content do not be prepared to offer this content in an accessible way.

Therefore, due to this increase, it is essential for multimedia content like video content to be accessible fulfilling standards such as Web Content Accessibility Guidelines (WCAG) [5] of Web Accessibility Initiative (WAI) [6]. In order to achieve access to all video content, it is fundamental to provide alternatives such as captions or sign language synchronized for deaf people or audio description for blind people with such content.

Additionally, accessible multimedia content on the Web requires that a particular chain of essential, interdependent, and accessible components [7] should be taken into account (see Figure 1 which is an adaptation of the essential components of Web accessibility of the WAI Guidelines and Techniques [8]).

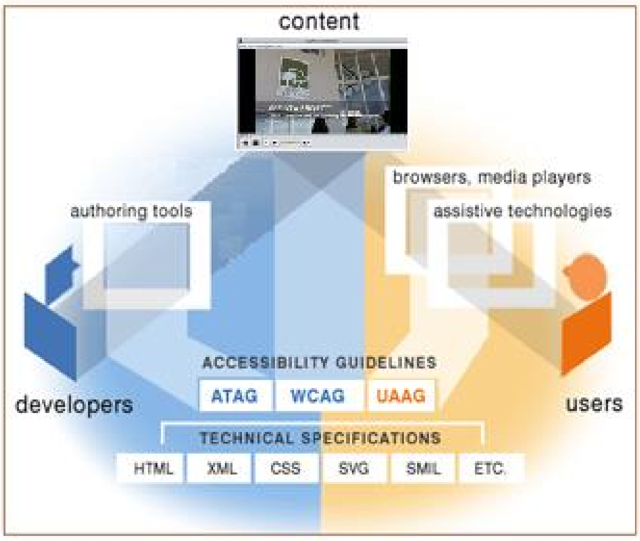


Figure 1 WAI Guidelines and Techniques regarding this PhD proposal

One of these components is the user agent such as Web browsers, media players, and assistive technologies. Therefore, it is important to follow standards such as User Agent Accessibility Guidelines (UAAG) [9] of WAI. Specifically, media players should enable the delivery of accessible multimedia content, enable alternatives and provide a friendly user-video interaction.

Nowadays, there are some barriers that do not allow multimedia content to be accessible. Between these barriers, it is important to highlight barriers within the user agent. For example, a user agent that provides video content or media player should provide mechanism to allow a final user to use, manage and control alternatives such as captions or audio description. Apart from this, it should react to user necessities and provide her/him aid. However, most of the time, a media player offers an interface less intuitive that needs some help or former knowledge when it is used. It is also important that the accessibility features which a user agent has will last until the user sets a new change or session. This fact does not usually happen, consequently, the user has to change her/his preferences every time she/he uses the user agent.

Apart from barriers within the media player, it is also crucial to take into account barriers that appear due to the lack of knowledge of the accessibility concept or the general belief that an accessible web page or media player produces drawbacks in the development process such as great corporate or human cost.

Furthermore, the question of how to apply the WAI standards in the design is in no way trivial, since many professionals are unable to distinguish between requirements aimed at accessibility. The developers' perception about web accessibility is that they require methodologies which incorporate web accessibility issues throughout the entire development process [10, 11].

And last, standards are necessary and useful, but never have to forget the final users of the user agent. It is necessary to follow an approach of user-centered design (UCD) [12], focusing on the importance of the user and involving her/him directly in the design process.

All this is what has motivated this Doctoral Thesis proposal. The aim of the Doctoral Thesis is to offer a design solution of an accessible user agent that provides accessible video content. That is to say, media players, which are capable of serving video content fulfilling the necessary requirements taking into consideration accessibility standards. So as to ensure access to the content by as many people as possible with independence of having any kind of disability or their context of use.

In conclusion, this Doctoral Thesis proposes a design solution that includes accessibility requirements in user agents using a methodological approach based on models. The aim of using this type of approach is to separate the platform-independent design from the platform-specific implementation of applications, delaying as much as possible the dependence on specific technologies [13].

This approach allows broadening this Thesis proposal from web environments to other environments in which it can interact with user interfaces of user agents. Moreover, this kind of approach provides independence of technology and platform and also facilitates the design of accessible media players by designers new to the area of accessibility. Apart from this, it can be compatible with different interaction modalities such as graphical interaction, vocal interaction, etc.

1.2. Context

The research work which is described in this Doctoral Thesis is mainly defined within the discipline of Software Engineering. This is due to this work tries to offer a design solution of an accessible multimedia user interface following methodologies which belong to this discipline such as model-driven or model-based approaches.

Moreover, this work is also related to Human-Computer Interaction because it takes into account standards which study part of the aspects regarding the interaction that occurs when a user accesses multimedia content through a user agent.

Although, this work does not directly interact with final users with disabilities, it will assist designers to design a user agent which takes into consideration preferences and needs of these users.

In addition, as a complement to this Thesis proposal should be applied a design approach focused on the user (UCD) to force her/his participation in the design process, as this proposal which is based on accessibility standards may be very focused on its architecture.

1.3. Objectives

This Doctoral Thesis aim is to provide a design solution that includes the necessary accessibility requirements for the development of an accessible user interface that delivers accessible multimedia content taking into account international accessibility regulations and standards.

Therefore, the main objectives of this Doctoral Thesis are:

Obj 1. Perform a literature review from which the accessibility requirements in a media player are obtained. This review includes an exhaustive analysis of standards, best practices and related work.

Obj 2. Create a methodological design approach which provides a workspace for designers to assist in the design of an accessible user agent that provides accessible video content.

Due to pursuing these objectives, in this Doctoral Thesis, a proposal composed of three main contributions is obtained:

Cont 1. A selection of accessibility requirements and its modeling to be included in the design.

Cont 2. A workspace oriented to designers which follows a model-based methodological design approach to design and develop an accessible media player.

Cont 3. As a proof of concept derived from the two previous contributions, a support tool which facilitates the design tasks of the media player is presented.

In addition, it would be necessary to apply a user-centered design (UCD) approach to force the user participation in the design process in order to complement this Doctoral Thesis proposal. It is due to the fact that the proposal is based on accessibility standards, and as aforementioned, it may be very focused on its architecture.

1.4. Structure

The structure of this document is organized as follows.

In Chapter 2, a review in order to determine the proposal"s bases is made. This review includes type of disabilities, standards and media players functionalities regarding accessibility, a study regarding the design of user interfaces and related works. Apart from this, a discussion of the review is carried out.

Chapter 3 presents one of the main contributions of this Thesis, the accessibility requirements which an accessible media player should include fulfilling accessibility standards.

In Chapter 4, the proposal of a methodological design approach using a user interface description language, which is the second contribution of this Thesis, is introduced.

Chapter 5 presents an authoring tool based on the PhD proposal for the design and development of an accessible media player, which is the third contribution of this Thesis.

The validation of the proposal is indicated in Chapter 6.

Chapter 7 presents the concluding remarks, the future works regarding the proposal of this Doctoral Thesis and finally, the results of the dissemination. These results are obtained starting from the different studies accomplished within this Doctoral Thesis.

Chapter 2. State of the Art

2.1. Introduction

In order to establish the bases of this proposal a literature review is made. This review is composed of different studies about accessibility related to disabilities, standards and media players, user interfaces design and related works regarding accessibility requirements, modeling or authoring tools among other things.

2.2. Accessibility and Disability

According to ISO 9241-171:2008 ("*Ergonomics of human-system interaction - Guidance on software accessibility*") [14], accessibility is the usability of a product, service, environment or the easiness for people within a wide range of capabilities.

Nowadays, there are plenty of barriers of various kinds to which people with disabilities have to face. Among these barriers it can be found architectural, urban, social, cultural and bureaucratic barriers. These barriers not only complicate the integration and participation of people with disabilities in different areas of the society, but they also exclude the rest of people who have any kind of temporary difficulty or elderly people.

Regarding the overview of how people with disabilities use the Web [15], there are different kind of disabilities which can affect Web access and its relation with accessibility problems on the Web:

- Visual disabilities:
    - Blindness. Blind people may use screen readers, text-based browsers, voice browsers or rapid navigations strategies to access the Web.
    - Low vision. In order to access the Web, this type of users may use extra-large monitor or increase the size of system fonts and images. They also use screen magnifiers or specific combinations of text and background colors.
    - Color blindness. They may use their own style sheets to access the Web.
- Hearing impairments:
    - Deafness. Deaf people may use captions for audio content in order to access all Web content.
    - Hard of hearing. In order to access the Web, they may use captions and/or audio amplification.
- Physical disabilities:

    - Motor disabilities. They may use a specialized mouse, voice recognition software, an eye-gaze system, etc. Besides, they may activate commands by typing single keystrokes in sequence with a head pointer.
- Speech disabilities: they may use voice recognition as well as any kind of alternative input mode.
- Cognitive and neurological disabilities:
    - Visual and auditory perception. They may access the Web getting information through several modalities at the same time.
    - Attention deficit disorder. People with this kind of disability may need to turn off animations on a site in order to be able to focus on the site's content.
    - Intellectual disabilities. Users with this disability may take more time on a Web site, rely more on graphics to enhance understanding of a site and benefit from the level of language on a site.
    - Memory impairments. They may rely on a consistent navigational structure throughout the site.
    - Mental health disabilities. People with mental health disabilities may need to turn off distracting visual or audio elements or to use screen magnifiers.
    - Seizure disorders. Users may need to turn off animations, blinking text or certain frequencies of audio.
- Multiple disabilities: user"s flexibility may be reduced when a combination of disabilities appears.
- Aging-related conditions: it includes changes in abilities or combinations of abilities.

Taking into account these types of disabilities and the proposal field, the Doctoral Thesis aim is to avoid as many accessibility barriers as possible regarding user interfaces design. In this case, accessibility barriers can be avoided including alternative content such as captions or sign language for people with hearing impairments or audio description for people with visual disabilities. Apart from this alternative content, there are assistive technologies which are used to access ICT; therefore, a compatible access to ICT through these assistive technologies has to be provided.

As aforementioned, people with disabilities are directly affected by accessibility barriers, although, due to the functional and technological diversity, these barriers are also experienced by people without any disability. Owing to this, it is essential to take into account international standards and regulations such as the ones which are going to be explained in the following section (Section 2.3).

2.3. Accessibility Standards and Regulations

As stated previously, it is essential for people to allow accessing to the great amount of multimedia content which is delivered on the Web. In order to achieve it, it is crucial that every country has a legislative framework in accessibility that benefits accessibility seating in all areas of modern society.

In order to avoid the discrimination of people at risk of social exclusion such as people with some kind of disability, on 10 December 1948, the United Nations Universal Declaration of Human Rights (UDHR) [16] was passed in which its first article highlights that "*all human beings are born free and equal in dignity and rights. They are endowed with reason and conscience and should act towards one another in a spirit of brotherhood*".

On the other hand, the International Organization for Standardization [17] has developed international standards in order to regulate accessibility. Accessibility extends beyond web environments, it affects every user interface. Due to this fact, it also crucial to consider different standards related to software and not only standards regarding web environments. Between these standards is important to highlight standards such as:

- ISO 9241-171:2008 (*Ergonomics of human-system interaction -- Part 171: Guidance on software accessibility*) which provides ergonomics guidance and specifications for the design of accessible software for use at work, in the home, in education and in public places. It covers issues associated with designing accessible software for people with the widest range of physical, sensory and cognitive abilities, including those who are temporarily disabled, and the elderly. This standard is applicable to the accessibility of interactive systems and it promotes the increased usability of systems for a wider range of users. It does not cover the behavior of, or requirements for, assistive technologies (including assistive software) and it does address the use of assistive technologies as an integrated component of interactive systems. In addition, this standard is intended for use by those responsible for the specification, design, development, evaluation and procurement of software platforms and software applications.
- ISO/IEC 40500:2012 (*Information technology -- W3C Web Content Accessibility Guidelines (WCAG) 2.0*) [18] which covers a wide range of recommendations for making Web content more accessible as it is going to be explained in the following paragraph.

Although this standard is relatively new, ISO/IEC 40500:2012 is based on Web Content Accessibility Guidelines (WCAG) 2.0 [19] of Web Accessibility

Initiative (WAI) [6]. As it has been mentioned, WCAG 2.0 covers a wide range of recommendations for making Web content more accessible. These guidelines allow making content accessible to a wider range of people with disabilities, including blindness and low vision, deafness and hearing loss, learning disabilities, cognitive limitations, limited movement, speech disabilities, photosensitivity and combinations of these. Moreover, these guidelines also often allow making Web content more usable to users in general. Regarding its structure, WCAG 2.0 contains three main layers of guidance: principles, guidelines and success criteria. On the one hand, there are four principles which are the foundation of Web accessibility: perceivable, operable, understandable and robust. On the other hand, there are twelve guidelines that provide the basic goals that Web designers should work toward in order to make content more accessible to people with functional diversity. And last but not least, there are sixty one success criteria. Several success criteria are provided for each guideline. These criteria allow WCAG 2.0 to be used where requirements and conformance testing are necessary, such as in design specification, purchasing, regulation and contractual agreements. In order to meet the needs of different groups and different situations, three levels of conformance are defined: A (lowest), AA, and AAA (highest). Therefore, WCAG 2.0 is referenced worldwide in most regulations [20]. Apart from that, there are other important standards, very similar to WCAG 2.0 although most of them are less extensive, which are totally or partly related to Web accessibility, such as technical standards:

- Section 508 [21] which is a federal law mandating that all electronic and information technology developed, procured, maintained, or used by the federal government be accessible to people with disabilities.
- BITV (*Barrierefreie-Informationstechnik-Verordnung*) 2.0 [22] which specifies the minimum requirements for online information and services provided by federal authorities.
- RGAA (*République Française, Référentiel Général d'Accessibilité pour les Administrations*) [23] which defines the set of requirements and evaluation process for determining if a web site is accessible.
- AODA (*Accessibility for Ontarians with Disabilities Act*) [24] which allows the government to develop specific standards of accessibility and to enforce them.
- UNE 139803:2012 [25] which establishes accessibility requirements for Web content.

Apart from WCAG 2.0, there are other standards of WAI which are also important to highlight. Among these standards, User Agent Accessibility Guidelines (UAAG) 2.0 [26] and Authoring Tool Accessibility Guidelines (ATAG) 2.0 [27] stand out.

On the one hand, UAAG 2.0 provides guidelines for designing user agents that lower barriers to Web accessibility for people with disabilities. A user agent includes browsers, media players and applications that retrieve and render Web content. The same as WCAG 2.0, UAAG 2.0 contains three main layers of guidance: principles, guidelines and success criteria. To start with, there are five principles which provide a foundation for accessible user agents (perceivable, operable, understandable, programmatic access and specifications and conventions). Among the guidelines, there are twenty eight guidelines which guide to make user agents more accessible to users with disabilities. And finally, there are also one hundred and fourteen success criteria. Each success criterion is also assigned a level: A (low or basic conformance), AA (recommended conformance) and AAA (highest conformance). These standards are applicable to this PhD proposal, therefore, they are going to be analized in Chapter 3. Due to the fact that this proposal is accomplished in non-web environments, Guidance on Applying WCAG 2.0 to Non-Web Information and Communications Technologies (WCAG2ICT) [28] is also taken into account.

On the other hand, ATAG 2.0 is a standard which provides guidelines for designing web content authoring tools that are both more accessible to authors with disabilities and designed to enable, support, and promote the production of more accessible web content by all authors. As happened with the other WAI standards, the ATAG 2.0 also contains three main layers of guidance: principles, guidelines and success criteria apart from being divided into two parts, A and B. To begin with, part A relates to the accessibility of authoring tool user interfaces to authors with disabilities, while part B relates to support by authoring tools for the creation, by any author (not just those with disabilities). Secondly, there are eight principles which organize the guidelines. Thirdly, there are twenty four guidelines which provide the basic goals that authoring tool developers should work toward in order to make authoring tools more accessible to both authors and end users of web content with different disabilities. And finally, there are sixty three success criteria which are used where requirements and conformance testing are necessary, such as in design specification, purchasing, regulation, and contractual agreements. In order to meet the needs of different groups and different situations, multiple levels of full and partial conformance are defined. Again, each success criterion is assigned a level: A (lowest), AA (middle) and AAA (highest).

Besides these standards, the Twenty-First Century Communications and Video Accessibility Act (CVAA) of 2010 [29] of U.S.A. has to be born in mind. The aim of this legislation is to establish new guarantees to ensure that people with disabilities do not fall behind as the technology changes and advances. The CVAA is divided into two titles: Title I (Commucation

Access) which amends specific sections of the Communications Act to increase the scope of communications services and equipment that must be accessible to disabled users. And Title II (Video Programming) which is focused on video programming and broadly requires that the FCC conduct inquiries and enforce regulations for making video programming, services and equipment accessible to disabled users.

And last but not least, the new European Standard EN 301 549 V1.1.1 [30]. This Draft specifies the functional accessibility requirements applicable to ICT products and services, together with a description of the test procedures and evaluation methodology for each accessibility requirement in a form that is suitable for use in public procurement within Europe. One of its chapters is related to media content requirements such as captions, audio description, etc.

2.4. Media player and Accessibility

Some related works and several media players that provide accessibility requirements have been reviewed.

After browsing the Internet, two groups of media players have been distinguished, standalone and embedded media players. On the one hand, standalone media players have more controls and are considered more accessible (see Figure 2).

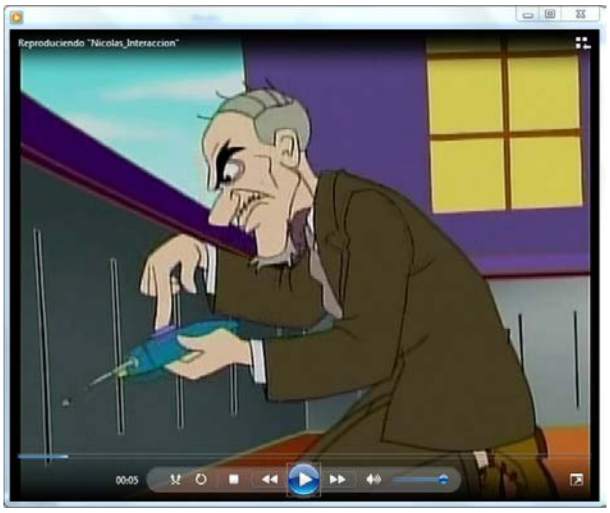


Figure 2 An example of standalone media player: Real Player

On the other hand, embedded media players are often considered more usable (Flash has been one of the most used formats for this type of players), see Figure 3.

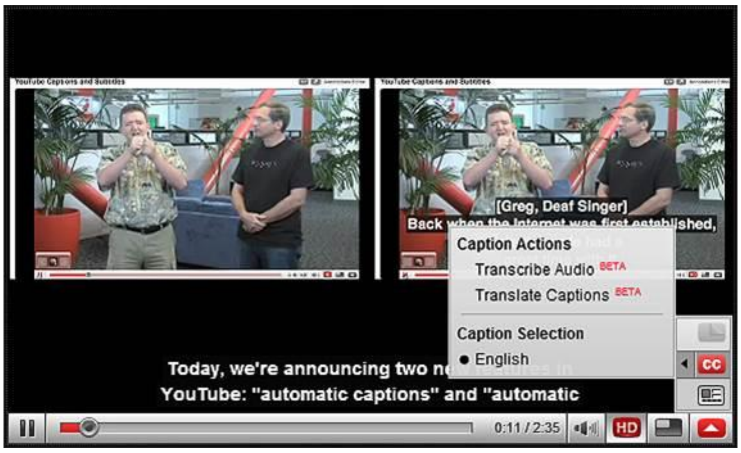

[Greg, Deaf Singer]
Back when the Internet was first established,
Caption Actions
Transcribe Audio BETA
Translate Captions BETA
Caption Selection
● English
Today, we're announcing two new features in
YouTube: "automatic captions" and "automatic
0:11 / 2:35
HD
CC

Figure 3 An example of embedded media player: Youtube

An accessibility study of some media players (CCPlayer[1], BBC iPlayer[2], YouTube[3]) framed in this research work, indicates compliance with some accessibility requirements on the part of the media players. In this study, CCPlayer was the most accessible of the three media players studied being BBC iPlayer the least one [31]. Although YouTube was less accessible than the CCPlayer, it is important to highlight ongoing efforts to improve accessibility, for example Easy YouTube [32], the development of automatic captions for six languages [33] and Access:YouTube [34], which simplifies the standard YouTube site through the use of assistive technologies and facilitating video clip search and play (see Figure 4).

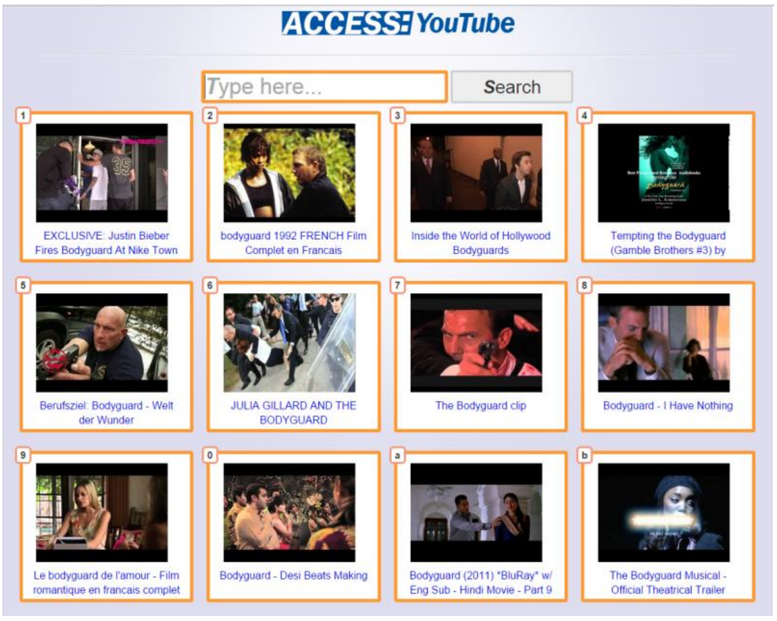

ACCESS: YouTube
Type here...
Search
1
EXCLUSIVE: Justin Bieber Fires Bodyguard At Nike Town
2
bodyguard 1992 FRENCH Film Complet en Francais
3
Inside the World of Hollywood Bodyguards
4
Tempting the Bodyguard (Gamble Brothers #3) by
5
Berufsziel: Bodyguard - Welt der Wunder
6
JULIA GILLARD AND THE BODYGUARD
7
The Bodyguard clip
8
Bodyguard - I Have Nothing
9
Le bodyguard de l'amour - Film romantique en francais complet
0
Bodyguard - Desi Beats Making
a
Bodyguard (2011) *BluRay* w/ Eng Sub - Hindi Movie - Part 9
b
The Bodyguard Musical - Official Theatrical Trailer

Figure 4 Screenshot of Access:Youtube

[1] CCPlayer, http://ncam.wgbh.org/invent_build/web_multimedia/tools-guidelines/ccplayer (September 2015)
[2] BBC iPlayer, http://www.bbc.co.uk/iplayer/tv (September 2015)
[3] Youtube, http://www.youtube.com (September 2015)

Other user agents that provide video content with accessibility features are: JW Player[4] that provides captions and audio description, BSPlayer[5] which provides captions and gives users the possibility of changing both the type and size of the font (see Figure 5) and VideoLan[6] media player that provides, for example, keyboard shortcuts and allows users to change the size, the font or the colour of the captions among other things. Apart from them, other example is KMPlayer[7], this media player provides closed captions and allows user to change the size, the format and rotate the captions among other characteristics. Another media player is OzPlayer[8]. OzPlayer is fully compliant with the Version 2.0 (Level AA) of the W3C Web Content Accessibility Guidelines. This video player has no keyboard traps, supports captions and audio descriptions and has a unique system for providing a moving transcript.

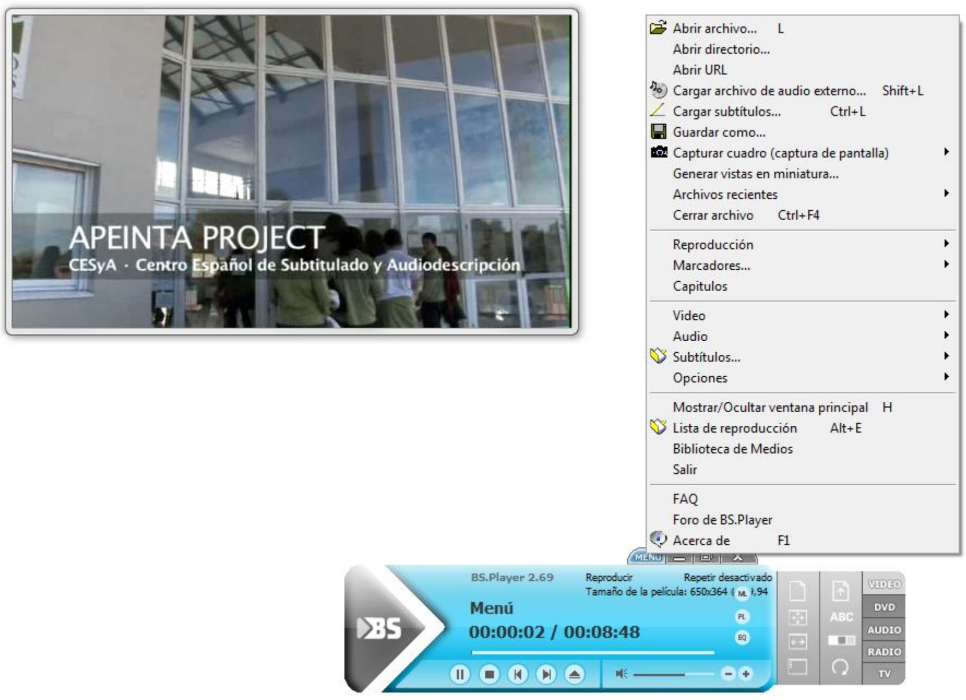


Figure 5 Screenshot of BSPlayer

It is also necessary to consider the new standard HyperText Markup Language (HTML) 5 which has been established as W3C Recommendation since October 2014 [35]. It defines the 5th major revision of the core language of the World Wide Web: the Hypertext Markup Language (HTML). In this version, new features are introduced to help Web application authors and new elements are introduced based on research into prevailing

[4] JWPlayer, http://www.longtailvideo.com/jw-player/ (September 2015)
[5]BSPlayer, http://www.bsplayer.com (September 2015)
[6] VLC, http://www.videolan.org/vlc/index.html (September 2015)
[7] KMPlayer, http://www.kmplayer.com/ (September 2015)
[8] OzPlayer, http://www.accessibilityoz.com/ozplayer/ (September 2015)

authoring practices. Apart from that, an special attention has been given to define clear conformance criteria for user agents in an effort to improve interoperability.

This standard allows users to play videos without installing plugins through new labels such as <video> or <audio>. Pfeiffer and Green thoroughly review these two elements in [36]. Apart from that, several user interfaces used by modern browsers are reviewed. In [37], the same authors, Pfeiffer and Green, discuss in their work the features which HTML5 offers to satisfy the accessibility and internationalization needs of media users. First of all, a requirements analysis which provides an overview of alternative content technologies from media content is accomplished. And after this, the HTML5 features offered to satisfy these requirements are introduced.

In addition to these elements, HTML5 provides new juicy tags, a canvas, drag and drop functionality or new form controls. Some HTML5 players with accessibility features have been found, such as Acorn Media Player[9] which provides full keyboard control, allows external SRT files to be used in order to include captions and provides a dynamic transcript generated from the selected captions. Another example is LeanBack[10] which gives support to <video> and <audio> labels, subtitles through HTML5 <track> element and provides keyboard shortcuts for desktop browsers. Other example is Video JS[11] (see Figure 6), this HTML5 video player has been plugged into JavaScript for some extra features, like a true full screen mode, and the ability to play subtitles. Apart from supporting captions it is easy to use. An improvement on the VideoJS Player is the Accessible HTML5 Media Player[12]. Among the features of this media player, it can be found support to captions and fully access to keyboard and screen readers. It is also important to take into consideration SublimeVimeo[13]. Among the features which this video player offers, it can be found multi-language subtitles support on all desktop browsers and the latest mobile platforms and it also allows controlling the video with simple keys like the space bar and arrows keys in order to improve the site's accessibility. And last but not least, it is fundamental to highlight that from this year, 2015, Youtube has realized that Flash is not the best solution for web video. Therefore, from this year onwards Youtube uses HTML5 like its default player.

[9] Acorn Media Player, http://ghinda.net/acornmediaplayer/ (September 2015)
[10] LeanBack, http://leanbackplayer.com/ (September 2015)
[11] VideoJs, http://videojs.com/ (September 2015)
[12] Accessible HTML5 Media Player, http://paypal.github.io/accessible-html5-video-player/ (September 2015)
[13] SublimeVideo, http://www.sublimevideo.net (September 2015)

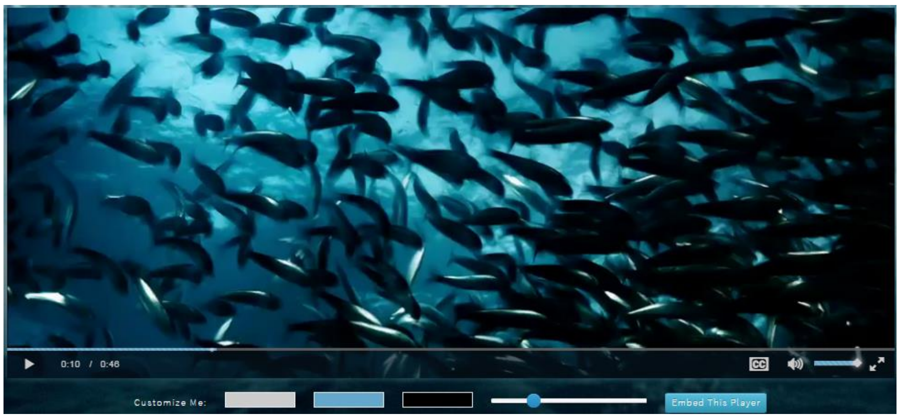


Figure 6 Screenshot of Video JS

After analyzing several media players, it can be observed that the media players are becoming more accessible.

2.5. User interfaces design

This Doctoral Thesis follows two paradigms in order to design a user interface, the model-based paradigm and the model driven paradigm. Apart from these paradigms, there are also methodologies, user interface description languages and graphical user interface technologies that can be used in order to design user interfaces.

2.5.1. Model-based User Interface design

The purpose of Model-Based Design is to identify high-level models that allow designers to specify and analyze interactive software applications from a more semantic oriented level rather than starting immediately to address the implementation level. This allows concentrating on more important aspects without being immediately confused by many implementation details and then having tools which update the implementation in order to be consistent with high-level choices [38].

The general architecture of a model-based user interface design is shown in Figure 7.

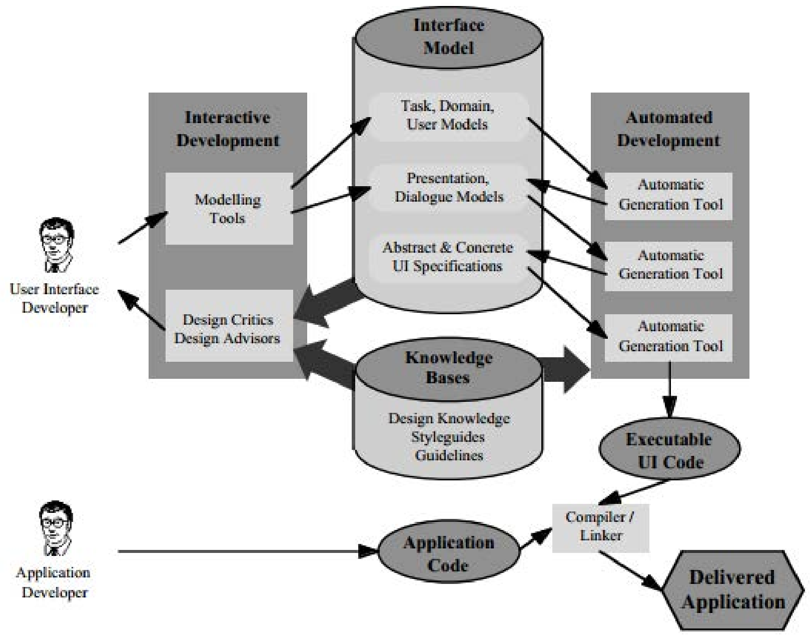

Interface Model
Task, Domain, User Models
Presentation, Dialogue Models
Abstract & Concrete UI Specifications
Interactive Development
Modelling Tools
Design Critics Design Advisors
Automated Development
Automatic Generation Tool
Automatic Generation Tool
Automatic Generation Tool
User Interface Developer
Knowledge Bases
Design Knowledge Styleguides Guidelines
Executable UI Code
Compiler / Linker
Application Developer
Application Code
Delivered Application

Figure 7 General architecture of a model-based user interface design [39]

As it can be seen, the central component of the Model-Based User Interface Development Environment (MB-IDE) is the Interface Model which includes different declarative models. MB-IDEs include tools for interactive development (Modeling Tools, Design Critics, Design Advisors) and automated development. Automatic Generation Tools deal with transformations between different declarative models and implement an executable representation of the desired user interface. Design Critics, Design Advisors and Automatic Generation Tools require additional knowledge represented in the Knowledge Bases. As far as User Interface Developers are concerned, they use Modeling Tools to create and manipulate the declarative models [39].

In the following paragraphs, the main models which were initially considered in user interface development are described. These models are task model, domain model, user model, dialogue model and presentation model.

- Task model: it helps to understand how the user interacts with the system and allows identifying the data which will be manipulated. There are many notations that are used in the task model. Among them, the most known and extended is ConcurTaskTree [40].
- Domain model: it describes the objects which are manipulated through the interface. Among its functions, it can be highlighted to identify and classify software components which can be applied within the application environment. This type of model is represented using entity relationship diagram and class diagram.

- User model: it describes the characteristics of the desired end users or groups of end users of the interactive system to be developed. Its main purpose is to support the creation of individual user interfaces.
- Dialogue model: it describes the communications between a human and a machine. It can be represented through state diagram, sequence diagram and transition diagram. Due to the evolution of techniques related to Mb-UID environment, these types of diagrams are included within the task model.
- Presentation model: it presents the description of the interface with which the user interacts.

2.5.2. Model Driven Development

Model-Driven Development (MDD) is an approach that uses models as a specification of software and transformations of those models to get the source code. Models are created before the source code is written or generated. MDD aims at speeding up the software development and making it more cost efficient, by using the models to visualize the code and, if used to raise the abstraction, the problem domain. MDD also separates implementation technology from the business logic of a program [41].

Regarding MDD, the most commonly known example of MDD is Model-Driven Architecture (MDA) [42] from the Object Management Group (OMG) [43]. In 2001 the OMG adopted the Model Driven Architecture (see Figure 8) as an approach for using models in software development. Its three primary goals are portability, interoperability and reusability through architectural separation of concerns [44].

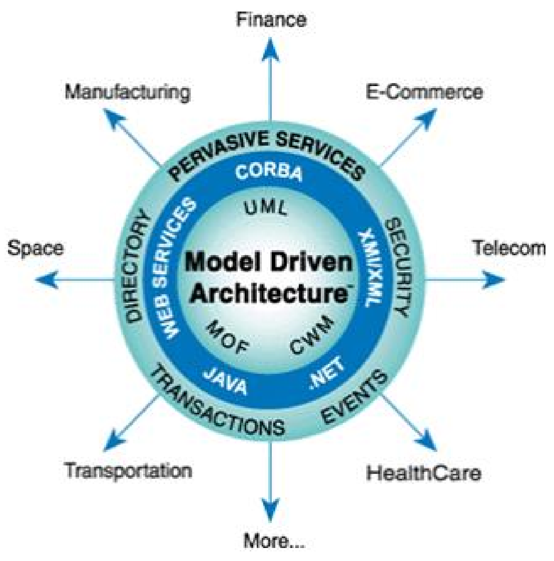


Figure 8 Model Driven Architecture [44]

MDA has three main advantages against other methodologies of software development [45]:

- Transferability that is connected with platform independency.
- Interoperability that is closely related to standard development.
- Reusability that is the result of the previous two advantages.

In addition, the life cycle of MDA is composed of four abstraction levels (see Figure 9): Computation Independent Model (CIM), Platform Independent Model (PIM), Platform Specific Model (PSM) and Implementation Specific Model (IM) – Code [46].

- CIM is a model that describes a system from the computation independent viewpoint.
- PIM is a model that contains no information specific to the platform.
- PSM is a model that includes information about the specific technology that is used in the realization of it on a specific platform.
- Implementation Specific Model - Code is a specification of the system in the source code.

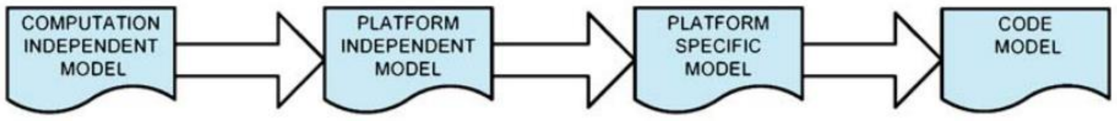


Figure 9 MDA abstraction levels

Apart from these models, it is also important to consider the Transformation Model of MDA. Although the main transformation is from the PIM to PSM transformation, there are other many transformations. On the one hand, the vertical transformations are the transformations in which the abstraction level changes. On the other hand, the transformations in which the abstraction level does not change are called horizontal transformations [47]. As for vertical transformations, these transformations are:

- CIM to PIM which might be somewhat abstract as computation independent, or business, models are typically not appropriate to express all the details needed for PIM. For this reason, PIM may be drawn separately, but all the requirements and other aspects defined in CIM are bearing in mind.
- PIM to PSM which is used when the PIM is sufficiently defined and its function is secured. In this transformation, the platform specific issues are attached to the PIM to form PSM, which should then be completely aware of its platform.
- PSM to code which is used when all the platform specific details are defined and the model is ready for actual implementation.

Regarding horizontal transformations, this type of transformations can be used during the design process to enhance, filter and specialize.

2.5.3. Model-based User Interface Development Methodologies

Some methodologies proposals which explain how to accomplish the user interface development process have been found. This is the case of Model-based User Interface Development (Mb-UID), an approach whose aim is to identify high-level models that allow designers to specify and analyse interactive software applications from a more semantic oriented level rather than starting immediately to address the implementation level. This allows them to concentrate on more important aspects without being immediately confused by many implementation details and then to have tools which update the implementation in order to be consistent with high-level choices.

2.5.3.1. TRIDENT

TRIDENT (*Tools foR an Interactive Development ENvironmenT*) [48, 49, 50] consists of a methodology and a support environment for developing user interfaces for business-oriented interactive applications. It uses ERA-diagrams for the description of the problem domain model.

TRIDENT presents two models:

- Task model which uses Entity Relationship diagram and Activity Chaining Graph.
- Presentation model in which user interface design is based on [51, 52]:
    - PU (*Presentation Unit*) which is a complete presentation environment required for carrying out a particular interactive task.
    - LW (*Logic Window*) which is a logical container for other interaction objects as a physical window, a dialog box or a panel. One or many LW composes a PU.
    - AIO (*Abstract Interaction Object*) which consists of an abstraction of all CIOs (*Concrete Interaction Object*) from both presentation and behavioral viewpoints that is independent of any given computing platform.
    - CIO which is a real object belonging to the user interface world that any user can see (e.g., text, image, animation) or manipulate such as a check box.

Apart from that, TRIDENT presents an architecture with three elements (see Figure 10):

- The Control Objects (CO) class which is generic with instances decomposed into COs of different types which both manage dialogue and preserve the correspondence between application Data and the presentation. Each CO has a specific behaviour that combines management of a portion of the dialogue and some application-presentation correspondences.
- The Application Objects (AO) class which is not generic, instances cannot be decomposed, since they represent the application functions.
- The Interaction Objects (IO) class which is generic and provides two types of CIOs: application-dependent CIOs that translate input and output information for functions; and application-independent CIOs that are required for a dialogue (e.g., command buttons that trigger functions).

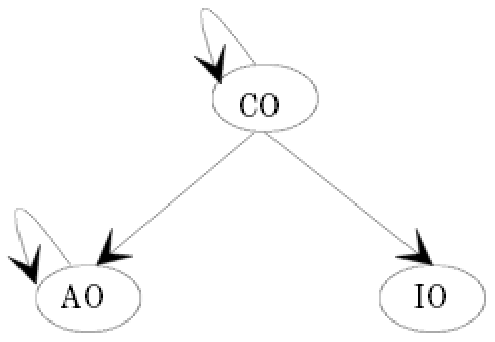


Figure 10 Architectural model of TRIDENT

2.5.3.2. WISDOM

WISDOM (*Whitewater Interactive System Development with Object Models*) [53, 54] is a methodological proposal to develop user interfaces based on UML which appears as an evolution of UCEP (*User-Centered Evolutionary Prototyping)* [55] and is based on OMT (*Object Modeling Technique*).

Wisdom is influenced by both Software Engineering and Human-Computer Interaction.

Its aim is to accomplish the final application starting from modeling cycles and evolutive prototype.

WISDOM has three important components: a process, a notation and a project philosophy [56]:

- A software process, based on a user centered, evolutionary and rapid prototyping model, ideally suited for constructing and maintaining interactive systems, such as the ones with a strong web user access component.
- A set of conceptual modeling notations, based on a very simple subset of the UML language, but offering effective support for modeling functional and non-functional requirements, in particular regarding user tasks and interactions.
- A pragmatic project management philosophy, based on agreed and open standards for documentation and tool usage, and requiring a flexible team with efficient and open channels of communication between contractors, users and developers.

2.5.3.3. IDEAS

IDEAS (*Interface Development Environment within OASIS*) [57, 58] is a model-based user interface development methodology which allows the user interface to be specified in UML diagrams formally using the OASIS specification language [59].

According to the common principles of Model-based User Interface Development Environments, the user interface specification process is tackled in parallel to the application development (see Figure 11).

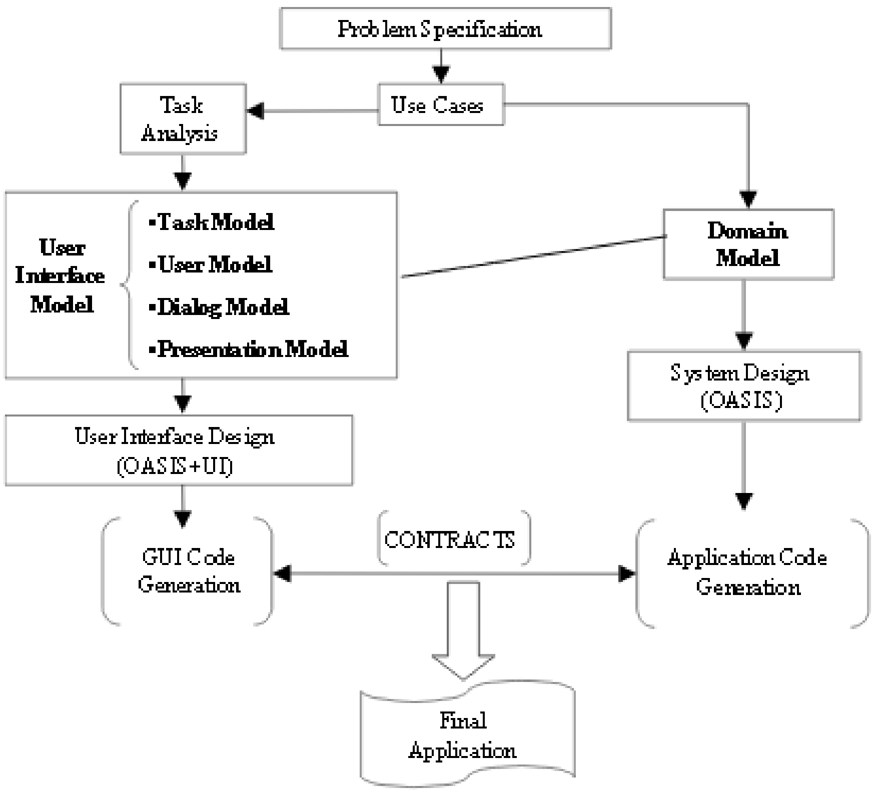


Figure 11 Development process of IDEAS

IDEAS proposes a user interface development process composed of four levels (see Figure 12).

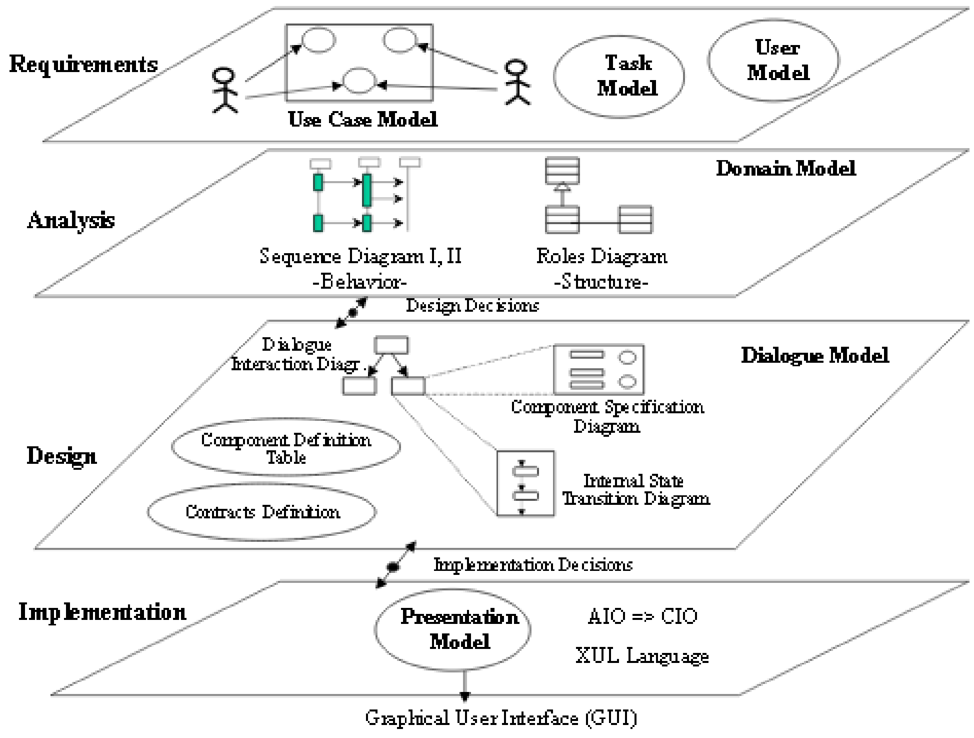


Figure 12 User Interface Development Process

These four levels are:

- Requirement level. In this level three models are created:
  - Use Case Model in which users are identified.
  - Task Model in which the ordered set of activities and actions the user has to perform to achieve a concrete purpose or goal are defined.
  - User Model in which the characteristics of the different types of users are described.
- Analysis level. The Domain Model is performed at this level. This Domain Model is composed of two diagrams, the Sequence Diagram, which defines the behavior of the system and the Roles Model, which defines the structure of the classes that take part in the associated sequence diagram together with the relationships among these classes, specifying the role of each one of them.
- The Dialog Model is performed at Design level. From now on, the graphical aspect of the final user interface starts to be addressed and the way in which the user-system interaction will be performed is especially important. The Dialog Model generates four different kinds of diagrams: Dialogue Structure Diagram, Component Specification

Diagram, Internal State Transition Diagram and Component Definition Table.
- Implementation level performs the Presentation Model. This model describes the concrete interaction objects (CIOs) composing the final graphical user interface (GUI), its design characteristics and visual dependencies among them. These CIOs are defined taking into account implementation decisions, the final implementation platform and according to the style guide followed.

2.5.3.4. Cameleon Reference Framework

Cameleon Reference Framework results from two key principles: a model-based approach and coverage of both the design and run-time phases of a multi-target user interface [60].

The aim of Cameleon Reference Framework is to build methods and environments supporting design and development of highly usable context-sensitive interactive software systems by:

- Providing the means to express context-dependent information in a set of models usable at design-time by developers and at run-time by dynamically reconfigurable systems.
- Developing tools that support the use of information contained in abstract representations to drive the design and development of concrete interfaces for multi-context applications while preserving usability.
- Developing techniques and components that facilitate the development of adaptive, context-dependent applications.
- Providing prototypes for validating the methods, techniques and tools developed.

Cameleon Reference Framework is structured in four levels (see Figure 13):

- The Task and Domain models correspond to the hierarchies of tasks that need to be performed on/with domain objects (or domain concepts) in a specific temporal logical order for achieving users" goals (during the interaction with the UI).
- The Abstract User Interface (AUI) model expresses the UI in terms of Abstract Interaction Units (AIU) or Abstract Interaction Objects (AIOs) as well as the relationships among them. These AIUs are independent of any implementation technology or modality (e.g., graphical, vocal, gestural). They can be grouped logically to map logically connected tasks or domain objects.
- The Concrete User Interface (CUI) model expresses the UI in terms of Concrete Interaction Units (CIU) or Concrete Interaction Objects

(CIOs). These CIUs are modality-dependent, but implementation technology independent, thus platform specific (PSM). The CUI concretely defines how the UI is perceived and can be manipulated by end users.

- The Final User Interface (FUI) model expresses the UI in terms of implementation technology dependent source code. A FUI can be represented in any UI programming language (e.g., Java UI toolkit) or mark-up language (e.g., HTML). A FUI can then be compiled or interpreted.

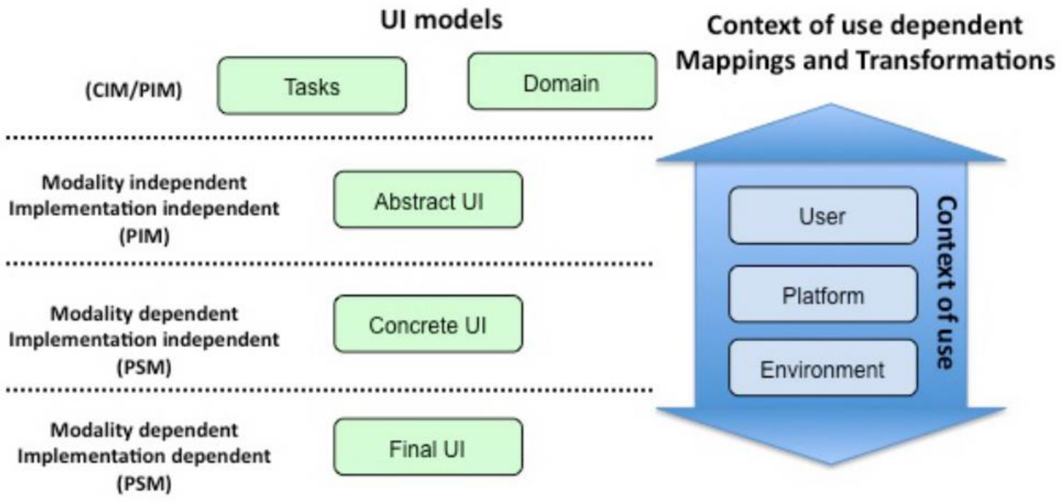

UI models
Context of use dependent Mappings and Transformations
(CIM/PIM)
Tasks
Domain
Modality independent Implementation independent (PIM)
Abstract UI
Modality dependent Implementation independent (PSM)
Concrete UI
Modality dependent Implementation dependent (PSM)
Final UI
User
Platform
Environment
Context of use

Figure 13 Simplified version of Cameleon Reference Framework [38]

2.5.3.5. Other methodologies

Among other methodologies, it can be highlighted:

- UIDE (*Use Interface Design Environment*) is designed to allow interface designers to easily create, modify and generate an interface to an application through high-level specifications. The purpose of the environment is to support the interface design process through its life cycle – from its inception to its execution. This support is made possible using a model which describes various details of an application interface including partial application semantics [61].
- MASTERMIND (MM) [62] is a research effort in the area of model-based user interfaces. It is a joint effort of the Graphics, Visualization, and Usability Center (GVU) of the Georgia Institute of Technology (GT) and the Information Sciences Institute (ISI) of the University of Southern California.
- FUSE (*Formal User interface Specification Environment*). FUSE system presents a methodology and a set of integrated tools for the automatic generation of graphical user interfaces. It provides tool-based support for all phases of the user interface development process. Based on a

formal specification of dialogue and layout guidelines, it allows the automatic generation of user interfaces out of specifications of the task, problem domain and user model. Furthermore, the FUSE system incorporates a component for the automatic generation of powerful help and user guidance components [63].

2.5.4. User Interface Specification Languages

The user interface development based on models uses high-level specification languages. These languages called User Interface Description Language (UIDL) [64] are used by MB-UID to describe the models in a formal way [65]. In the following subsections some of these languages are going to be explained.

2.5.4.1. USIXML

UsiXML (*User Interface eXtensible Markup Language*) [66] is a UIDL based on XML. This UIDL is a declarative language capturing the essence of what a User Interface is or should be independently of physical characteristics. Moreover, it is also considered a methodology which specifies how the information on the user is modelled and used to customize the user interface.

UsiXML is characterized by three principles [67]:

- Expressiviness of UI: any UI is expressed depending on the context of use thanks to a suite of models that are analysable, editable and manipulable by a software agent.
- Central storage of models: each model is stored in a model repository where all UI models are expressed similarly.
- Transformational approach: each model stored in the model repository may be subject to one or many transformations supporting various development steps.

This language is structured according to the four levels of abstraction defined by the Cameleon Reference Framework (see Figure 14): Task and Domain, Abstract User Interface (AUI), Concrete User Interface (CUI) and Final User Interface (FUI).

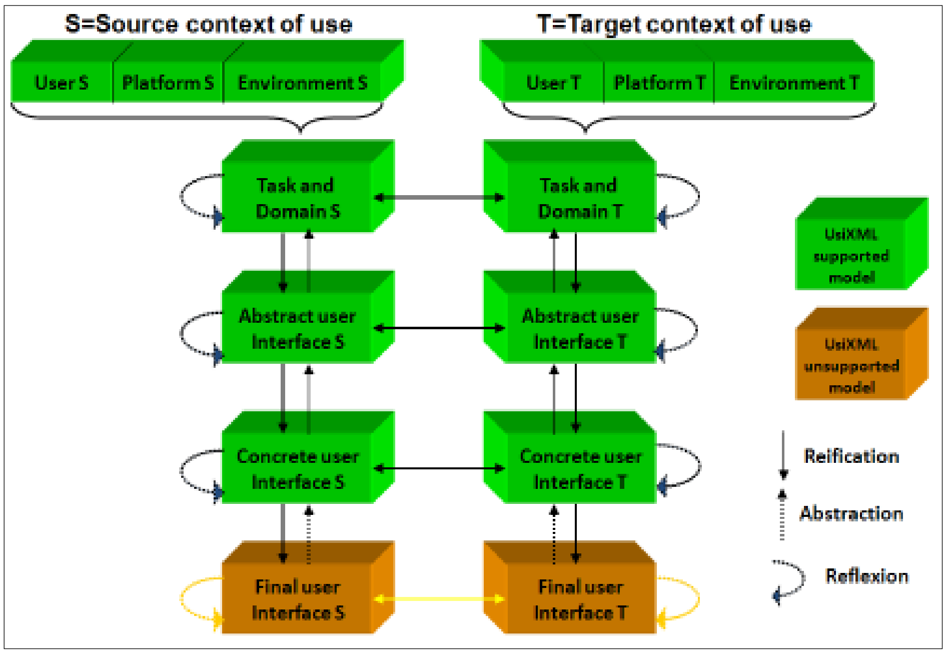


Figure 14 UsiXML MDE-compliant approach [UsiXML wikipedia]

UsiXML is universally recognized and is used to design and develop interactive systems due to the richness of the models offered. In addition, it supports device independence, platform independence and modality independence. UsiXML describes the User Interface for multiple contexts of use such as Graphical User Interfaces, Character User Interfaces, Multimodal User Interfaces and Auditory User Interfaces. Thus, interactive applications with different types of computing platforms, interaction techniques and modalities of use can be described in a way that sustain the design independently of the peculiar characteristics of the physical computing platform.

2.5.4.2.XIML

XIML (*eXtensible Interface Markup Language*) is a XML-based language which enables a framework for the definition and interrelation of interaction data items. It can provide a standard mechanism for applications and tools to interchange interaction data and interoperate within integrated user-interface engineering processes, from design, to operation, to evaluation.

The requirements which are found essential for a language of its type are [68] (see Figure 15):

- To support design, operation, organization and evaluation functions.
- To be able to relate the abstract and concrete data elements of an interface.
- To enable knowledge-based systems to exploit the captured data.

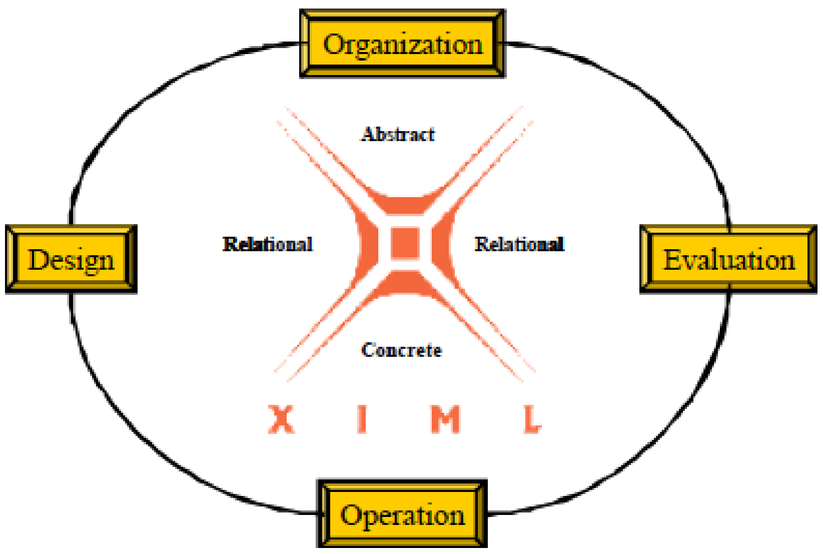


Figure 15 XIML major requirements

The structure of XIML is composed of five components [68]:

- Task component which captures the business process (that is, the part of the business which the interaction with a user is required) and/or user tasks which the interface supports. Task component defines a hierarchical decomposition of tasks and subtasks which also defines the expected flow among those tasks and the attributes of those tasks.
- Domain component which is an organized collection of data objects and classes of objects structured into a hierarchy, which is similar in nature to that of an ontology, but at a very basic level. Objects which are defined via attribute-value pairings are restricted to those that are viewed or manipulated by a user and can be either simple or complex types.
- User component which defines a hierarchy. A user in the hierarchy can represent a user group or an individual user. The characteristics of the users are defined by attribute-value pairs. This component does not attempt to capture the mental model (or cognitive states) of users but rather, data and features that are relevant in the functions of design, operation and evaluation.
- Presentation component which defines a hierarchy of interaction elements that comprise the concrete objects that communicate with users in an interface. It is generally intended that the granularity of the elements in this component will be relatively high so that the logic and operation of an interaction element are separated from its definition. In this way, the rendering of a specific interaction element can be left entirely to the corresponding target display system.

- Dialog component which defines a structured collection of elements which determine the interaction actions that are available to the users of an interface. It also specifies the flow among the interaction actions that constitute the allowable navigation of the user interface. This component is similar in nature to the Task component but it operates at the concrete levels.

2.5.4.3. UIML

UIML (*User Interface Markup Language*) is a declarative XML-based language that can be used to define user interfaces [69]. There are two aspects of this language which are important to highlight, on the one hand, the ability to generate user interfaces for different platforms, for example Java AWT, WML or VoiceXML, and on the other hand, the ability to construct a library of reusable components of the interface.

Among the purposes of designing this language, it can be found [70]:

- To provide a natural separation between user interface code and non-interface code.
- To facilitate reuse by non-programmers.
- To reduce the time to develop user interfaces for multiple device families.
- To permit rapid prototyping of user interfaces.
- To facilitate internationalization and localization.
- To allow efficient download of user interfaces over networks to Web browsers.
- To enhance security.
- To allow the language to be extensible to support future technologies.

Apart from this, UIML provides five elements:

- Structure element. A user interface description in UIML includes an enumeration of the set of interface parts comprising the interface.
- Content element. The UIML document specifies the interface content in a separate XML document.
- Behavior element is described by enumerating a set of conditions and associated actions. This is motivated by rule-based systems.
- Style element specifies presentation style which is device-specific for each class of interface parts, or for individual named instances of a class. It specifies the mapping of interface parts to a vocabulary of names of user interface widgets in the platform to which the user interface will be mapped.
- Peers element specifies what widgets in the target platform and what methods or functions in scripts, programs or objects in the application logic are associated with the user interface.

2.5.4.4. MARIA

MARIA (*Model based lAnguage foR Interactive Applications*) [71] is a universal, declarative, multiple abstraction-level, XML-based language for modeling interactive applications in ubiquitous environments. For designers of multi-device user interfaces, one advantage of using a multi-layer description for specifying user interfaces is that they do not have to learn all the details of the many possible implementation languages supported by the various devices, but they can reason in abstract terms without being tied to a particular UI modality or, even worse, implementation language. In this way, they can better focus on the semantics of the interaction, namely what the intended goal of the interaction is, regardless of the details and specificities of the particular environment considered.

This language supports user interfaces description at abstract and concrete levels. The abstract language is independent of the interaction platform. A number of concrete languages are part of MARIA and provide refinement of the abstract description for various platforms (graphical desktop, graphical touch-based smartphone, graphical mobile, vocal, multimodal which is a combination of graphical and vocal).

On the one hand, Maria Abstract User Interface (AUI) level describes a user interface only through the semantics of the interaction, without referring to a particular device capability, interaction modality or implementation technology. This model is composed of various presentations which group model elements presented to the user at once. The model elements are of two types: Interactor (selection, edit, control and only output) or Interactor Composition (grouping, relation, composite description and repeater). Apart from the presentation, the AUI describes the interactive behavior through Data Model, Generic Back End, Event Model, Dialog Model, Continuous update of fields and Dynamic Set of User Interface Elements.

On the other hand, MARIA Concrete User Interface provides platform-dependent but implementation language-independent details of a user interface. A platform is a set of software and hardware interaction resources that characterize a given set of devices. Every platform meta-model is a refinement of the AUI specifying how a given abstract interactor can be represented in the current platform.

2.5.4.5. Other specification languages

- XUL (*XML-Based User Interface Language*) [72]. XUL is a user-interface language developed by Mozilla that enables developers to design cross-platform applications that can run in both online and offline modes. It is customizable using different graphics, text and layouts to support localized and internationalized markets.

- OpenLaszlo, a free, open source platform, was built from the ground up for application development and is centered on standard development approaches. Its applications are written in LZX, an XML language that includes embedded JavaScript. Besides, the applications are portable across browsers [73].
- AUIML (*Abstract User Interface Markup Language*) [74] is an intent based interface definition language, primarily developed at IBM. It allows user interface designers to design the intent of the interface without tying the tasks to any concrete, device specific realization. AUIML consists of two major sets of elements, those that define data model and those that define presentation model.
- Teresa XML [75] is the XML-compliant language that was developed inside the Teresa project, which is intended to be a transformation-based environment. It provides an environment that supports the design and the generation of a concrete user interface for a specific type of platform.

2.5.5. Graphical User Interface Technologies

Apart from methodologies and languages in order to design a user interface based on models, it is also important to take into consideration other ways to develop a graphical user interface. Therefore, this section introduces several technologies that help designers and developers to accomplish this objective.

2.5.5.1. Java

Java is a high-level programming language originally developed by Sun Microsystems and released in 1995. Java runs on a variety of platforms, such as Windows, Mac OS and the various versions of UNIX.

Among the features which can be found include [76]: object-oriented, platform independent, simple, secure, architectural-neutral, portable, robust, multithreaded, interpreted, high performance, distributed and dynamic.

2.5.5.2. Python

Python is a programming language that lets you work more quickly and integrate your systems more effectively [77]. Python is freely available and that makes solving a computer problem almost as easy as writing out one's thoughts about the solution. It can be written once and run on almost any computer without needing to change the program.

Phyton is interpreted, interactive, object-oriented and beginner"s language. Besides, it is easy to use, easy to read, easy to maintain, a broad standard library, interactive mode, portable, extendable, databases, GUI programming and scalable.

2.5.5.3. C Sharp

The definition of C Sharp evolves different goals [78]. Some of these aims are to be a simple, modern, general-purpose, object-oriented programming language. Besides, both the language and its implementations should provide support for software engineering principles. C Sharp is intended for use in developing software components suitable for deployment in distributed environments. It is also important its source code portability, its support for internationalization, its suitability for writing applications for hosted and embedded systems and its applications are economical regarding memory and processing power requirements.

2.5.5.4. Visual Basic

Visual Basic is an ideal programming language for developing sophisticated professional applications for Microsoft Windows [79]. It makes use of Graphical User Interface for creating robust and powerful applications. The Graphical User Interface as the name suggests, uses illustrations for text, which enable users to interact with an application. This feature makes it easier to comprehend things in a quicker and easier way.

2.5.5.5. Perl

Perl is by far the most popular programming language for creating scripts that add powerful interactive features to Web pages. It is available free of charge for Windows and Macintosh and is included in most UNIX platforms. Among other capabilities, Perl allows placing forms on a Web site that collect and process user input, enables visitors to conduct keyword searches and allows integrating a database into a site [80].

2.6. Related work

Due to the fact that this work collects a set of requirements based on standards, accomplishes an abstraction of these requirements and concretizes the abstract elements through a graphical editor taking into consideration approaches which use models and develops a graphical editor through a graphical user interface technology, in this section, different works related to these different aspects treated in this Doctoral Thesis are presented.

2.6.1. Study of Accessibility Requirements

The start point of this research is to analyze different standards in order to gather a set of accessibility requirements related to a media player. Therefore, several works in which the analysis of accessibility requirements is presented have been found.

Apart from the work [31] explained in Section 2.4 (Media Player and Accessibility) in which an accessibility study of three media players is carried out, there are other examples. For example, Lourdes et al. [81] present a proposal of accessibility requirements to be considered in the design and development of an electronic guide in museums. Other example is Brunet et al. [82]. This paper discusses accessibility requirements for accommodating users with vision impairments from the complementary perspectives of the systems architect, the assistive technology developer and the application developer.

On the other hand, Rosas Villena et al. [83] present a study whose aim is to examine users' needs, expectations and requirements for accessible videos. To accomplish the evaluation with users, an accessible video player, called Facilitas, is developed. After the evaluation is carried out, the results are presented in the form of guidelines.

The last example is Peissner et al. [84]. In this paper a set of requirements that adaptive user interfaces must comply before being able to reach significant impact on the software market is presented. Besides, it identifies strategies and individual answers, how these requirements can be addressed and met in future systems building on the Prosperity4all approach. It gives a comparison of existing research solutions and how they compare with the stated requirements.

2.6.2. Improving content accessibility

There are several works whose aim is to improve the accessibility of video content.

As far as alternative content is concerned, the majority of the works are related to captions. For example, Federico and Furini [85] propose an architecture that automatically creates captions for video lessons by exploiting advances in speech recognition technologies or Universal

Subtitles[14], a web service that allows users to create and share subtitled versions of YouTube, Vimeo[15], Blip.tv[16] and USTREAM[17] videos. Wald [86] offers a tool that allows using crowdsourcing of the correction of speech recognition captioning errors in order to provide a method to make audio or video recording accessible because there are people who cannot understand speech through hearing alone. Another example is the work of Hughes et al. [87]. This work presents a new approach to display subtitles in the video content following a responsive web design paradigm. It allows enabling subtitles to be formatted appropriately for different devices whilst respecting the requirements and preferences of the viewer. Besides, a responsive video player prototype and a report of initial results from a study to evaluate the value perceived by regular subtitle users are presented.

In another work, Lim et al. [88] propose a dynamic subtitle authoring method based on audio analysis for the hearing impaired. The analysis includes different features such as STE, ZCR, Pitch and MFCC. Thanks to these features, dynamic subtitle which allows hearing impaired users to understand the scene contexts are created. As for the work of Hong et al. [89], this work describes a dynamic captioning scheme to enhance the accessibility of videos towards helping the hearing-impaired audience better enjoy videos. Dynamic captioning put scripts at suitable positions to help the hearing-impaired audience better recognize the speakers. It also synchronously highlights the scripts by aligning them with the speech signal and illustrates the variation of voice volume to help the audience better track and perceive scripts. The effectiveness of the presented scheme has been demonstrated through a comprehensive user study with 60 hearing-impaired participants.

Apart from these long paper, there are also short papers related to captions. One example is the work of Lasecki at al. [90] in which a real time demo of Legion:Scribe is presented. Scribe is a crowd-powered captioning system that allows untrained participants and volunteers to provide word-for-word transcription of spoken content with a latency of less than 5 seconds. This system uses multiple individual captionists, each of whom type a piece of what they hear, and then stitches the partial captions back together automatically. Another example is the work presented by Stinson et al. [91]. This paper presents C-Print a typing-based transcription system which provides real-time captioning. The system requires a trained transcriptionist who uses computerized abbreviations and condensing strategies to produce the text display of spoken information. Moreover, the

[14] Universal Subtitles, http://boingboing.net/2010/12/09/universal-subtitles.html (September 2015)
[15] Vimeo, https://vimeo.com/ (September 2015)
[16] Blip.tv, http://blip.tv/ (September 2015)
[17] USTREAM, http://www.ustream.tv/new (September 2015)

transformation from spoken information to text can be viewed by consumers approximately two seconds later.

Another work in which captioning as well as other types of alternative content are presented is the work of Hansen et al. [92]. This work describes basic requirements in order to accomplish accessible films in Web platform such as captions for deaf people, audio description for blind people and text transcripts which is a collated combination of a text equivalent of the auditory track and a text equivalent of the visual track that are essential for deaf-blind people and helpful for many others.

Regarding alternative content such as audio description. In Kobayashi et al. [93], an initial attempt to develop a common platform for adding audio description to an online video is described. Or the work accomplished by Chapdelaine and Gagnon [94] in which is presented a Web platform for rendering audio description using an adapted player. The aim of this work is to test the usability of an accessible player that provides end-users with various levels of audio description, on-demand.

Another alternative content is the sign language. According to sign language, a system to enable the embedding of selective interactive elements into the original text in appropriate locations, which act as triggers for the video translation into sign language is designed and developed by Debevc et al. [95].

With respect to the use of annotations to provide accessibility two works, [96] and [97], stand out. Both works are based on the Collaborative Annotation for Video Accessibility (ACAV) project whose goal is to improve videos accessibility on the web for people with sensory disabilities. In order to do that, Saray Villamizae et al. [96] present additional descriptions of key visual/audio information of the video using accessible output modalities (such as Text-To-Speech or refreshable Braille display). On the other hand, Encelle et al. [97] present an exploratory work that is focused on video accessibility for blind people with audio enrichments composed of speech synthesis and earcons (for instance, non-verbal audio messages). Apart from presenting a scientific and technical context of the ACAV project, they introduces their technical proposal for enriching video with audio elements (including earcons) and they focus on the experiments with blind people in order to know the usability and utility of these earcons.

Besides these works and taking into account the research where this work is framed, two other works appear. González-García et al. [98] establish accessibility requirements that should have a media player in order to be considered accessible. And starting from this first work, in other work from the same authors [99], an approximation based on models about how to design an accessible user agent that provides video content is accomplished.

Apart from the previous works, there are others where a media player is developed among other things. For example, in a work of Nishimura and Cohen [100], two versions of a media player led to people with different capabilities are developed. In this case and taking into account the user needs, tools that are easy to use and with appropriate contents for the presented applications, considering the educational environment, are developed.

It is also important to highlight works such as the work of Niederl et al. [101] in which is provided a guide about how to produce sign language based synchronization for movies and a video player which plays and shows two different movies at once. Or works such as the work of Mourouzis et al. [102] where a media player is developed. In this work, apart from the traditional functionality of a media player, the player itself allows users to transform Web content into its interactive equivalent in audio format.

Other work is the work of Rosas Villena et al. [103].This work describes the three phases of the User Centered Design. First of all, the phase of design research in which the users and their needs are assessed. The second phase is the design which starts from the previous research. Thanks to this research, a brainstorming and conceptualizing and sketching initial drafts of the design are carried out. Therefore, an accessible media player called Facilitas Player is developed. The last phase is the design evaluation where the interface is evaluated with users and is revised basing on the results of the evaluation.

Besides accessibility requirements in video content, it is relevant to increase Web pages" usability. In relation to this, in a work accomplished by Hanson and Crayne [104], user controls that make a number of dynamic adaptations to the page presentation and input that can greatly increase the usability of Web pages for older users are discussed.

2.6.3. User Interface Design approach

This Doctoral Thesis proposal presents an approach from the task model to the final user interface. Therefore, different approaches, ones based on models and others following a model driven approach, have been studied.

2.6.3.1. MDD approach

The first approach of this Doctoral Thesis uses MDD to avoid the technology and platform dependency. Besides, MMD allows guiding the design and development of accessible media player by accessibility"s non-expert professionals integrating accessibility requirements within the model.

Regarding MDD and approaches in which transformations among different levels of abstraction are presented, some works have been found. For example, Stanciulescu et al. [105] present a four step transformational approach from the task and domain model to the final user interface. In order to accomplish it, three steps are necessaries. The first step is to derive one or many abstract user interfaces from a task model and a domain model. Secondly, concrete user interfaces are deriving from each abstract one. And the last step is to produce the final user interface code (being this FUI a multimodal interface due to the fact that graphical and vocal interaction are involved).

Apart from this work, Stanciulescu also defends the definition of a design space-based method in his Doctoral Thesis. This method is supported by model-to-model colored transformations. Therefore, this fact allows obtaining multimodal user interfaces of information systems from a task model and a domain model [106]. Within this work, it is argued that the development of multimodal UIs is an activity that would benefit from the application of a methodology which is typically composed of a set of models gathered in an ontology, a method which manipulates the involved models and tools which implement the defined method.

There are other examples of developing software. For example, Link et al. [107] present a case study which demonstrates the usage and benefit of model-driven approach applied to a common software development process. Therefore, this work is concentrated on the aspect of user interaction by presenting an approach for model-driven software development of graphical user interfaces for any kind of platform. Other example is the work of Lu and Wan. In this paper, they propose a model-driven development of complex user interface. In this approach the process data using an Extended Object Model is captured [108]. The last example is presented by Huang et al. [109]. In this work, a model-driven approach to design the software part of a mechatronic system is proposed. It is composed of systematic modeling and correctness-preserving synthesis.

2.6.3.2. Model-based approach

The aim of model-based approaches is just to identify relevant models allowing designers to specify and analyze software artefacts from a more semantics-oriented point of view, rather than being forced to address immediately the low level details of the implementation level [110].

This type of approach is used in different kind of applications and interfaces. For example, Chesta et al. [111] present a model-based approach for designing and developing multi-platform applications. Besides, this approach is discussed through an experimental evaluation. This work presents some innovative techniques which provide software engineering support for developing applications accessible through multiple heterogeneous platforms studied within Cameleon Reference Framework.

Other example is Melchior et al. [112]. This work describes a model-based approach to design distributed user interfaces (DUIs). This approach is composed of three main pillars. Among them, it can be found, a Concrete User Interface model for DUIs, a specification language for DUI distribution primitives and a step-wise method for modeling a DUI.

And last, Rodriguez et al. [113] propose a model-based design approach of multi-user interfaces for groupware applications. In order to accomplish the user interface design for this type of applications, first of all the foundation of the proposal is described, secondly, connections between system models and those used in order to create user interfaces are established and finally, a general description of the design process is introduced.

Besides, thanks to model-based, different environments are created. For instance, Puerta describes Mobi-D in [114]. Mobi-D (*Model-Based Interface Designer*) is a comprehensive environment that supports user-centered design through model-based interface development. This environment offers three innovations. Among them, it can be found meta-level modeling language, formal definition of interface design and design philosophy of decision-making support.

Apart from these works, there are others which present a review about model-based developments among other things. At this point, Saleh et al. [115] present a review of the history of model-based user interface design and development. The studied approaches had built an applicable solution which can allow designing and developing multi-devices user interfaces thanks to model-transformations. And in Meixner et al. [116], first of all, it presents an overview of the research field of User Interface Management Systems (UIMS) and Model-Based User Interface Development (MBUID), secondly, it describes current and important approaches and finally, it discusses future challenges which must be sorted out to deal with the challenges and achieve the goals.

2.6.3.3.Adaptive user interfaces

As far as user interfaces adaptation is concerned, this research work introduces several adaptation rules. Therefore, some works in which this type of rules is also used have been found.

Hanumansetty in her Doctoral Thesis [117] presents a framework for adaptive user interface generation where adaptation occurs when context changes. Thanks to this framework three new concepts are introduced. First of all, it introduces formalization for representing context. Secondly, it studies a user interface generation life cycle and defines a context model on top of task model to introduce the contextual conditions into user interface generation process. And finally, it achieves a context aware adaptation of user interfaces. In order to do that, context specifications to various levels of user interface generation life cycle are mapped.

Adaptive user interfaces can be used to increase the accessibility of user interfaces, as it can be see in different works. On the one hand, Miñón et al. [118] present a design space for adaptation rules for accessible applications. These rules are classified according to user disability and considering other relevant criteria useful to ease their integration in other design tools. Apart from that, the design of a repository and the necessary meta-information to share these rules across several applications is also presented. On the other hand, Peissner et al. [119] present MyUI. MyUI generates adaptive user interfaces in order to support accessibility. MyUI provides a framework for extensive run-time adaptations to user characteristics, environmental conditions and used devices.

However, other works are led to adapt different environment or context of use. For example, Ranaweera and Withanage [120] present a study on how the adaptive user interface concept can be applied for personal desktop and how desktop environments can be personalized depending on each user. As a result, a system which enables the computer desktop to adapt its layout and screen elements automatically regarding the needs or preferences of a user is obtained. While, Giani et al. [121] present an architecture in order to create Service Front Ends which can be able to adapt different context of use. Besides, it is described a set of modules which manages the different aspects of adaptation. Apart from that, it also describes a set of languages in order to define User Interface structure, describes when adaptation should take place and what the effects are and interconnects different context sensing devices and applications to create shared representation of the context of use.

Apart from these works, there are several projects which also illustrate adaptive user interfaces. A special mention is dedicated to Serenoa Project [122] in accessibility and adaptability area. This project is driven by the following principles: 1) new concepts, languages, runtimes and tools are needed to support multi-dimensional context-aware adaptation of Service Front-Ends; 2) keep humans in the loop; 3) open adaptiveness and 4) covering the full adaptation lifecycle to support a full adaptation life-cycle which will result into feedback loops to inform any future adaptation.

2.6.4. Graphical Editor Development

Due to the fact that a graphical editor wants to be developed in this Doctoral Thesis work in order to provide a tool support in the design of a media player, other works related to the development of a graphical editor are presented.

There are works in which apart from the tool, they present a methodology. Among these works, it can be found the work of Mori et al. [123] which presents a method and a tool to support the design and development of nomadic applications. It starts with the design of a task model of nomadic applications. Then, it allows designers to obtain effective user interfaces for the various platforms.

Montenegro et al. [124] design a Domain-Specific Model (DSM) for the construction of platform-independent modules of learning management systems (LMS). This approach aims at building a meta-model for the construction of a domain specific language (DSL), using model-driven engineering (MDE) techniques, and applying the appropriate transformations to achieve a platform-independent model. They use two Eclipse [125] plugins to create the DSM: Eclipse Modeling Framework (EMF) and Graphical Modeling Framework (GMF). These same plugins are used in the first approach of this work.

It is also important to take into account the work of Ayotte et al. [126]. This work presents both a methodology for inclusive design and designs for flexible and adaptable preference management tools. These tools empower users to personalize their experience, adapting the interface to their own unique needs and preferences.

Other works in which only the tool is presented are also considered. For example, Creissac and Alves [127] present FlexiXML. FlexiXML is an interpreter tool which supports the automatic generation of user interfaces through models expressed in User Interface Extensible Markup Language (UsiXML).

Another graphical tool called SPA4USXML is presented by Miñon et al. [128]. This tool assists the designer of ubiquitous services to create specifications for the Task and the Abstract User Interfaces (AUI) required by the EGOKI adaptive system [129]. SPA4USXML also provides EGOKI with a resource model for selecting the most appropriate type of multimedia resource (text, video, audio, image, etc.) for the user.

Apart from this work, in Kanai et al. [130], a 3D tool for digital prototyping and usability assessment of information appliances is proposed. In this work, firstly, the UsiXML specification is extended; secondly, 3D prototyping and simulation functions are developed; and then, automated user test and usability assessment functions are also developed.

Finally, there are examples of editors developed in programming technology. For instance, Mr.Document[18] which is a light and simple text editor written in Java. It has support for saving and opening .txt files on a pc and for opening other types of text documents. Other example is EJE[19]. This simple Java editor is perfect to learn Java. EJE is multi-platform, light-weight, user-friendly and has several useful basic features such as Javadoc support, multilanguage support or printing support. Smith et al. [131] present a tool, called JavaSpeak, which assists students with visual disabilities in learning how to program. Besides, the motivation and philosophy of the full tool and details of a prototype implementation of it are also presented.

2.7. Discussion

After accomplishing the literature review, some conclusions have been obtained.

In spite of the fact that citizens are more conscious about the difficulty of people with disabilities to cope with daily life, the number of accessibility barriers inevitably grows. These barriers are not only architectural, but they are also barriers to content, especially multimedia content.

Although, there is an extensive legislation on accessibility, neither the content nor the user agent which delivers this content does fulfil the directives included in accessibility standards. This fact happens, not because designers or developers do not want to fulfil them, but because sometimes, they do not know their existence, they do not be aware of the problem or these directives are difficult to use or even to understand.

Apart from that, a lack of support in order to integrate accessibility requirements has also been found. For example, regarding a user agent which delivers video content, some works have been found, but these works do not include accessibility requirements from the lower levels of design. Therefore, the aim of this Doctoral Thesis is to establish a set of accessibility requirements related to media players and based on accessibility standards, accomplish an abstraction of these requirements

[18] Mr.Document, http://mrdocument.sourceforge.net/ (September 2015)
[19] EJE, http://sourceforge.net/projects/eje/ (September 2015)

using user interface specification languages and propose a design solution which includes these accessibility requirements from the lower levels of the design process.

In order to fulfil the Doctoral Thesis aim, different paradigms, methodologies, languages and technologies have been studied. Regarding the paradigms, both of them (MDD and MBUID) have been used. In relation with methodologies, Cameleon Reference Framework is selected. The reason of use it is that thanks to its structure, it covers both the design phase and the run-time phase. Besides, this methodology allows using two different user interface specification languages, UsiXML and MARIA, due to the fact that both languages are based on its architecture.

After accomplishing these tasks, the reality is that these modeling approaches are oriented to experts in the field of modeling. Therefore, so as to expand this work to designers or developers without any expertise in this field, this Doctoral Thesis also presents a graphical editor development taking into consideration a universal programming technology how it is the case of Java and every requirement or necessity found in the previous approaches.

As a result, the lacks which have been found in this review have motivated the accomplishment of this Doctoral Thesis whose final objective is the design of an accessible media player.

Chapter 3. Accessibility requirements analysis

3.1. Introduction

In order to obtain a subset of essential accessibility requirements of a user agent that provides video content, it is necessary to carry out a review of the standards indicated in section 2.3 covering a wide range of guidelines for providing web multimedia content in an accessible way such as User Agent Accessibility Guidelines (UAAG) 2.0 [26] and ISO 9241-171 Ergonomics of human-system interaction - Guidance on software accessibility [14].

3.2. Accessibility requirements for a user agent that provides video content

UAAG 2.0 of WAI provides guidelines for designing user agents (such as browsers, media players, assistive technology) which lower barriers to Web accessibility for people with disabilities. A user agent which fulfils these guidelines will promote accessibility through its own user interface and its ability to communicate with other technologies (especially Assistive Technology (AT)). In order to achieve the needs of different audiences, UAAG provides three layers of guidance: overall principles, general guidelines and testable success criteria. A user agent can be a browser, a media player or an AT like screen reader, etc, therefore, there are guidelines which can be applied to all user agents while there are others which depend on the user agent. For example, according to a media player, this kind of user agent has to satisfy guidelines such as Guideline 1.1 (Provide access to alternative content) related to the inclusion of alternative content (captions, audio description, sign language) or the Guideline 1.3 (Provide highlighting for selection, keyboard focus, enabled elements, visited links) regarding highlighted items (enabled input elements, visited links) and highlighting options (foreground or background colour). Annex I shows these guidelines.

ISO 9241-171 includes four guidelines that are necessary in order to consider this type of user agent accessible: 1) the user agent shall enable users to stop, start and pause the playback; 2) it should enable users to replay, rewind, pause and fast forward or jump the playback; 3) it should enable users to select media streams which are presented; 4) it should enable equivalent alternatives to be updated when the content of a media presentation changes. This standard also includes other requirements related to captions, such as whether the contrast between the captions and the background is going to be enough or if the system-wide preferences change during playback, the new settings shall be used. Furthermore, the captions position should not interfere with the visual content and the captions must be enabled or disabled.

In the following subsections, a set of guidelines that have to be considered to create a media player accessible is presented. First, a group of guidelines is established and starting from them, a set of requirements is derived. This group is essential to design an accessible media player.

3.3. Key guidelines to design an accessible media player

Before designing any type of user agent, the first step is to know which elements make up the user agent, in the present case, a media player.

Then, a set of accessibility requirements that are necessary to be included in a media player are obtained [98]. These requirements are:

- To provide alternative context such as captions (see Figure 16), audio description, sign language, transcription or extended audio description.

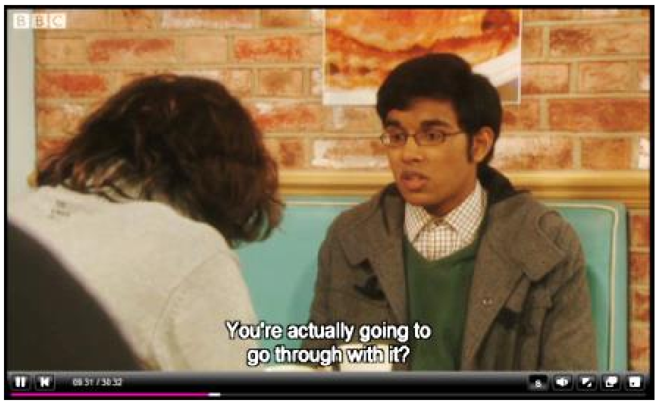


Figure 16 BBC iPlayer screenshot showing captions

- To provide complete access to all the features via the mouse, the keyboard or through AT.
- To provide help and access to documentation to know what accessibility features the media player has and how to use them.
- To provide a keyboard focus cursor to know what element has the focus on the user interface and a text cursor to know the location of the focus within a text element (see Figure 17).

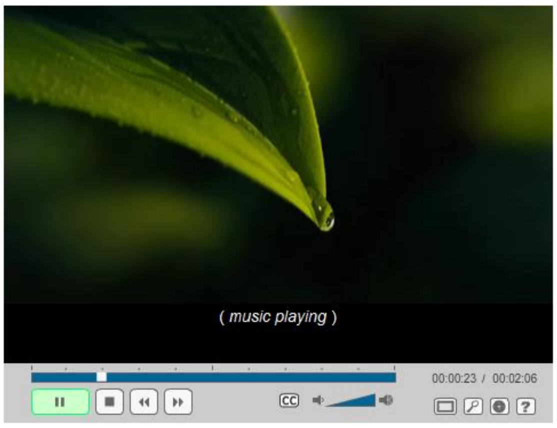


Figure 17 CCPlayer screenshot showing focus cursor

Taking into account the previous requirements, there is a set of elements which is necessary to be included to satisfy these guidelines. Among these elements the following controls can be found (see Figure 18):

- Play, pause or stop: allows users to start, pause or stop the playback.
- Forward or rewind seconds: allows users to move forward and backward within the playback.
- Resize: allows users to change the size of the interface.
- Adjust the volume: allows users to modify (increase or decrease) the volume of the playback.
- Enable or disable captions: allows user to show or not captions during the playback.

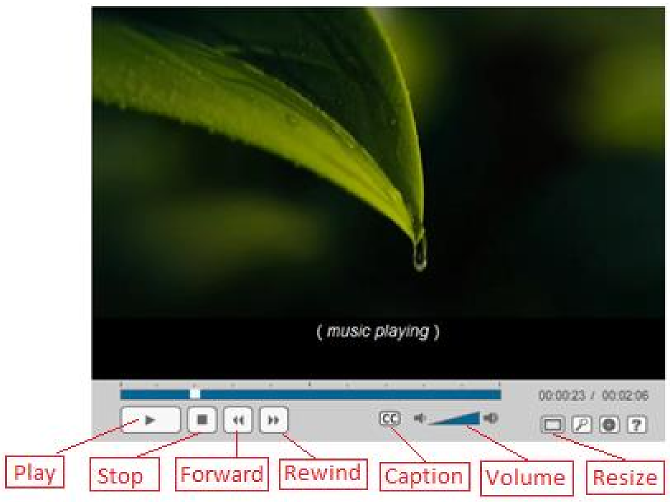


Figure 18 CCPlayer screenshot showing elements of the interface

- Enable or disable audio description: allows user to play or not the audio description track during the playback.

- Search captions: allows users to seek a word within the captions of the playback (see Figure 19).

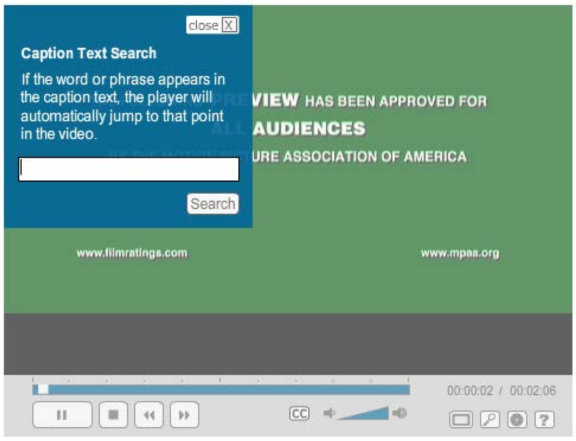

close X
Caption Text Search
If the word or phrase appears in the caption text, the player will automatically jump to that point in the video.
Search
00:00:02 / 00:02:06

Figure 19 CCPlayer screenshot showing caption text search

- Change the size, colour or font of the captions: allows users to modify some features (size, colour and font) of the captions (see Figure 20).

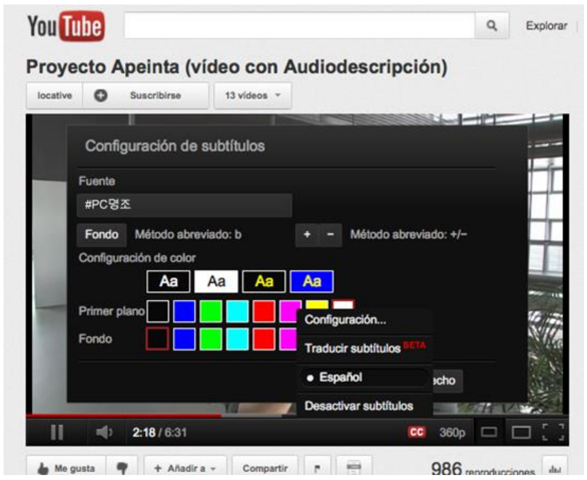
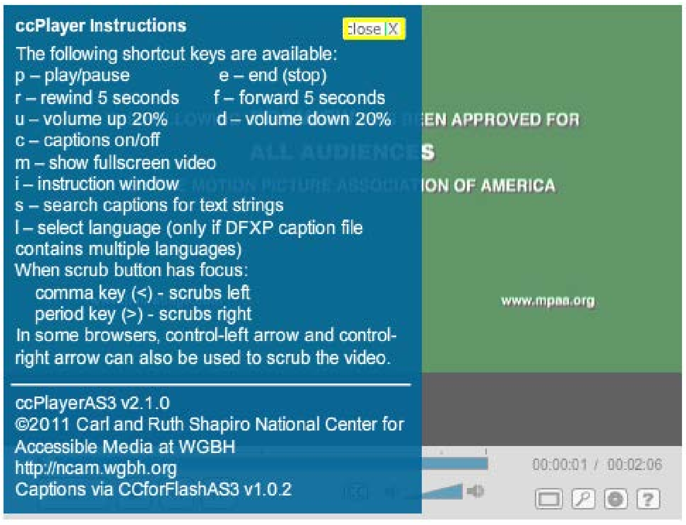


Figure 20 YouTube's screenshot with captions and text setting menu

- Access to help documentation: allows users to understand all the features and functions offered by the interface (Figure 21 shows a type of help documentation such as shortcuts).



ccPlayer Instructions

The following shortcut keys are available:

p – play/pause	e – end (stop)
r – rewind 5 seconds	f – forward 5 seconds
u – volume up 20%	d – volume down 20%

c – captions on/off
m – show fullscreen video
i – instruction window
s – search captions for text strings
l – select language (only if DFXP caption file contains multiple languages)
When scrub button has focus:
comma key (<) - scrubs left
period key (>) - scrubs right
In some browsers, control-left arrow and control-right arrow can also be used to scrub the video.

ccPlayerAS3 v2.1.0
©2011 Carl and Ruth Shapiro National Center for Accessible Media at WGBH
http://ncam.wgbh.org
Captions via CCforFlashAS3 v1.0.2

Figure 21 CCPlayer screenshot showing the shortcut keys menu

Once these elements are established, it is essential to set up a group of relationships and constraints between them.

For example, the „Play“ control is one of the first elements within the interface which has to be enabled. That is essential as it makes no sense to enable other elements of the user agent such as „Stop“ or „Pause“ if playback has not started.

In the following sections, these constraints are taken into account to define accessibility requirements and model them in the design of an accessible media player.

3.4. Accessibility requirements of a media player

Among standard guidelines, it is important to highlight that a user agent has to combine different alternatives together with video, such as captions, audio description, sign language, transcription and extended audio description, among others. In addition, it has to provide complete access to all features, via the mouse, the keyboard as well as through AT (such as screen readers). Moreover, it is necessary that this content provides help and documentation on its accessibility characteristics in the user interface to report the availability of those characteristics to the user, as well as the information on their purpose and use. The result of this review of standards is a set of accessibility requirements shown in Table 1.

This set of accessibility requirements is one of the contributions of this Doctoral Thesis regarding objective 1 (Obj 1).

It is necessary to include this set of accessibility requirements in the design. Table 1 shows these requirements groups by categories. These categories are the result of abstracting the different requirements depending on whether the requirements are basic (native in Table 1), which group the traditional requirements included in the user agent which provides video content or media player or if the requirements add new functionalities (additional in Table 1), which are necessary to satisfy specific accessibility requirements.

Table 1 User agent accessibility requirements

Code	Name	Description	Group	Subgroup	Source	
					ISO	**UAAG 2.0**
NP01	Play	Play the video content	Native	Playback	10.8.2	2.11.6
NP02	Stop	Stop the video content	Native	Playback	10.8.2	2.11.6
NP03	Pause	Pause the video content	Native	Playback	10.8.2	2.11.6
NS01	Resize	Resize the viewports	Native	Size	10.5.8	1.8.3
NV01	Mute	Enable or disable the audio content	Native	Volume	10.6.2	1.5.1
NV02	Volume	Adjust the volume	Native	Volume	10.6.2	1.5.1
AP01	Rewind	Delay seconds within a playback	Additional	Playback	10.8.3	2.11.7
AP02	Forward	Forward seconds within a playback	Additional	Playback	10.8.3	2.11.7
AA01	Caption	Enable or disable captions	Additional	Alternatives	10.1.3, 10.7.2	1.1.2
AA02	Audio Description	Enable or disable audio description	Additional	Alternatives	10.1.3	1.1.2
AA03	Size	Change the size of the captions	Additional	Alternatives	10.7.3	1.4.1
AA04	Font	Change the font of the captions	Additional	Alternatives	10.7.3	1.4.1
AA05	Colour	Change the colour of the captions	Additional	Alternatives	10.7.3	1.4.1
AA06	Language Caption	Change the language of the captions	Additional	Alternatives	8.2.1	2.7.1
AA07	Language Audio	Change the language of the audio description	Additional	Alternatives	8.2.1	2.7.1
AH01	Help	Help documentation about accessibility features	Additional	Help	11.1.5	3.3.2
AF01	Find	Search within playback captions	Additional	Find		2.4.5

As it is shown in Table 1, every requirement has a code in order to identify itself. This code is made up of the first letter of the group, the first letter of the subgroup and two numbers, for example, the requirement NP01 belongs to native group, playback subgroup and has been assigned the number 01.

Besides these requirements, there are other important requirements which are related to the presentation. This type of requirements are non-functional requirements such as aspects concerning the interface contrast. In these requirements must be followed the guidelines of the WCAG 2.0 regarding content and a DCU approach in order to ensure the accessibility through the user guidelines.

Apart from these non-functional requirements, there are requirements which are included in concrete level of modeling not in the first levels of modeling. Among these requirements, it is very important to keep accessibility features that are configured by users in previous sessions and change only when the user wants to change them. It is essential to allow users to enable or disable and adjust accessibility features, which also have to be easy to find and be operable.

Moreover, it is fundamental to consider the keyboard focus in order to show the user which element is active, as it is indicated in the ISO.

It is also necessary to provide users with information in order to know the keyboard shortcuts that can be used in the user agent and allow navigation through the content without enabling any controls. Likewise, moving through menus, submenus and lists should be easy to use through different keyboard combinations or direct keyboard commands. Last but not least, it is important to allow users to set their preferences to configure the keyboard shortcuts.

3.5. Conclusions

In this chapter, a review of accessibility standards (UAAG 2.0 and ISO 9241-171) in order to obtain a set of accessibility requirements regarding a user agent which provides video content or media player has been accomplished.

It is important to take into consideration that the requirements established in Table 1 are the ones which can be included in the first levels of modeling related to the design of a user interface, that is, in the abstract levels of modeling.

As aforementioned, apart from these requirements, it is essential to include other requirements as it passes to more concrete levels such as the configuration of preferences, access via keyboard or the interface contrast.

Chapter 4. Methodological approach

4.1. Introduction

In order to define a methodological support to design an accessible user agent that provides accessible video content with the integration of the accessibility requirements that have been presented in Chapter 3, two approximations defined by two architectures have been followed: the first one following a model driven development (MDD) approach and the second one following a model-based user interface development (MBUID) approach (see Figure 22).

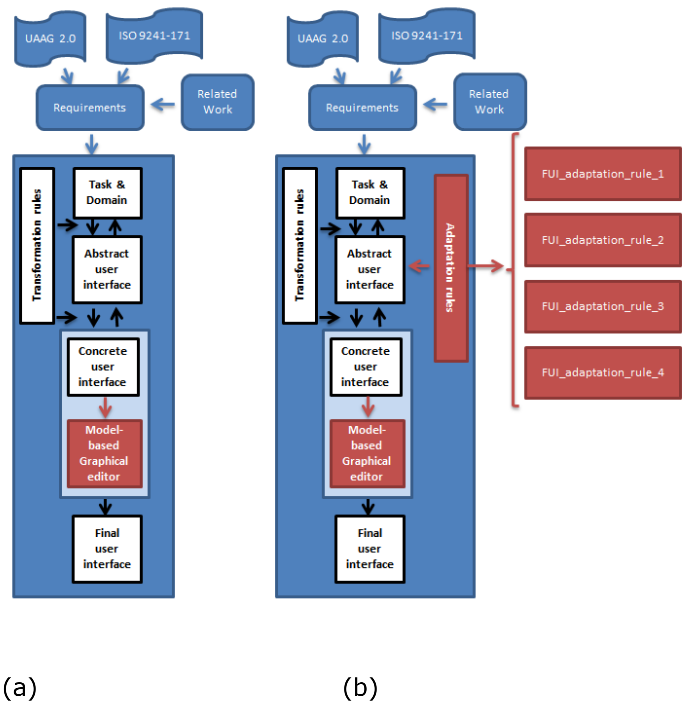


Figure 22 Proposal of architectures of the methodological approaches: (a) First proposal using UsiXML; (b) Second proposal using MARIA

These two approximations which follow different paradigms are provided in order to extend the scope of this proposal to as many as possible designers with expertise in modeling.

4.2. Methodological approaches to design an accessible media player

The reason of using two architectures is to provide the possibility of including adaptation rules. Apart from that, two User Interface Description Language (UIDL) are also used, in the first approach UsiXML technology, while in the second one is used MARIA framework. Although the adaptation rules can also be included using the UsiXML framework, the necessity of simplifying the process of design, through the different models of Cameleon Reference Framework using less authoring tools, was the reason of selecting MARIA framework in the second architecture.

As it can be seen in Figure 22, both architectures are structured according to the four levels of abstraction of Cameleon Reference Framework. The following subsections present the levels of abstraction of the proposal defined in Cameleon Reference Framework together with the transformation rules.

As aforementioned, the abstraction levels and their functionalities are:

- Task model which shows the interaction between a user and an accessible user agent.
- AUI which shows the structure of the interaction elements.
- CUI which concretizes an AUI for a given context of use.
- FUI which is the operational user interface on a particular computing platform.

Besides these different levels of abstraction provided, adaptation rules are also created and integrated in the methodological approach proposal. In order to optimize the final user interface design to meet different user needs, preferences, and situations (access by people with disabilities, people with temporary disabilities and people whose abilities have changed due to aging) some adaptation rules are included.

In addition, in order to provide a tool support for designers, and not to require a high level of professional expertise, a graphical editor is going to be provided. This editor is going to assist in the design of an accessible user interface and has to be complemented with a user-centered design approach to force the user participation in the design process. This tool is presented as a proof of concept derived from the objectives of this Doctoral Thesis: a literature review in order to obtain accessibility requirements regarding a media player and based on accessibility standards and a workspace which follows a methodological approach to develop an accessible media player.

4.3. First approximation: UsiXML framework

As aforementioned, this first approximation is structured according to Cameleon Reference Framework and uses UsiXML framework as UIDL in order to describe the user interface (see Figure 22 (a), Figure 23). Therefore, the following subsections explain each one of the Cameleon Reference Framework abstraction levels. Annex II shows the meta-models used in this first approximation.

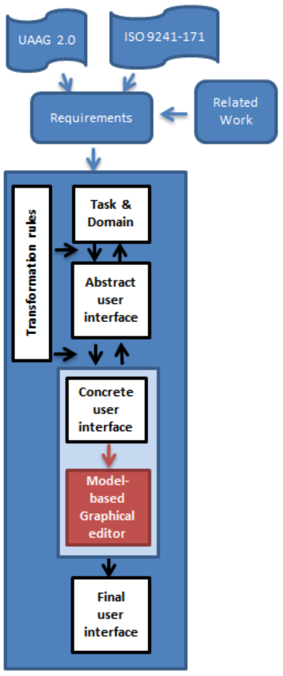


Figure 23 Proposal of architecture of the methodological approach using UsiXML

4.3.1. *Task Model*

The accessibility requirements described in Chapter 3 have been modelled through the Task model. This model uses the ConcurTaskTrees (CTT) [132] notation. Figure 24 shows the model taking into consideration how a user agent providing accessible video content operates. There are also several tasks defined together with the relationships between them.

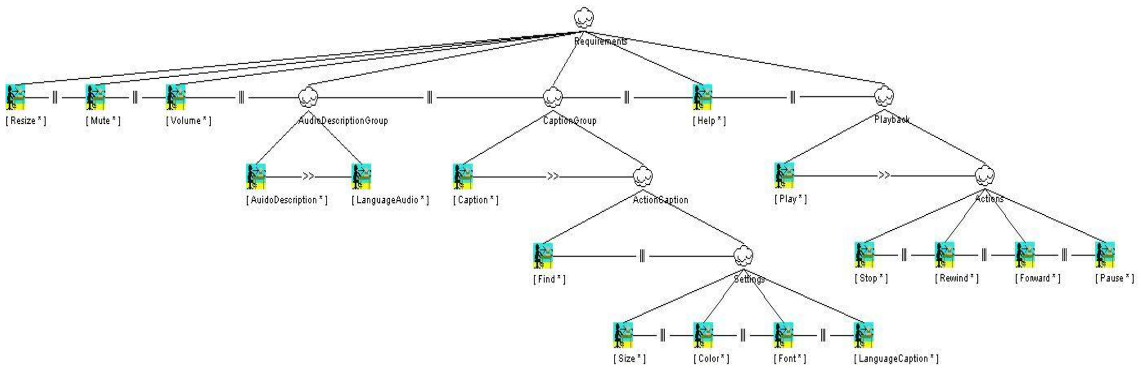


Figure 24 Task model of accessible user agent that provides accessible video content

In order to gather the semantics of interacting through playing, stopping, pausing, rewinding or forwarding the video playback an abstract task (Playback) composed of an interaction task „Play" (which includes the NP01 requirement) and an abstract task (Actions) that includes four interaction tasks: „Stop", „Pause", „Rewind" and „Forward" (which includes the requirements NP02 (Stop), NP03 (Pause), AP01 (Rewind) and AP02 (Forward)) have been defined. These tasks are related to each other or with themselves through temporal relationships. There is a temporal unary relationship, iteration (this relationship is represented with an asterisk in task model, see Figure 24), in task „Play" and there are two temporal unary relationships, optional (represented with square brackets in Task model, see Figure 24) and iteration, in tasks „Stop", „Pause", „Rewind" and „Forward". This happens because „Play" task has to be carried out at least once, while the remaining tasks can be carried out or not (Guideline 2.11.2 of UAAG 2.0). On the other hand, the temporal binary relationship between „Play" and the abstract task (Actions) is defined as enabling. It means, "Play" enables the set of tasks in which the tasks „Stop", „Pause", „Rewind" and „Forward" are included when it finishes. Therefore, the user agent allows users to stop, pause, rewind and forward the playback after finishing the „Play" task. As far as the general abstract task is concerned, this task has the temporal binary relationship defined as enabling with the rest of the task.

The „Resize" task (which includes the NS01 requirement) is created to gather the semantics of enlarging the size of the user agent screen. Like other tasks, this also has two unary relationships (optional and iteration) with itself and it has a relationship defined as an independent concurrency with other tasks. In this case, the task related to enlarging the size of the viewport can be carried out at any time if the user wishes to see the playback in a larger way and in any order.

The semantics of establishing the volume i.e., increasing or decreasing the volume or muting the volume, is carried out by defining two interaction tasks: „Mute" and „Volume", which include the requirements NV01 and NV02. Both tasks have an optional and an iteration relationship with themselves and binary relationships of independent concurrency with another task. The fulfilment of these tasks is similar to the previous tasks. In this case, the user agent has to allow users to modify the volume depending on their preferences at that moment.

The tasks „AudioDescription" and „LanguageAudio" (which include the requirements AA02 and AA07 respectively) are contained within an abstract task. Both tasks have two temporal unary relationships with themselves, optional and iteration, and a temporal binary relationship between themselves, enabling, which means the language of the audio description cannot be changed within the established values until the audio description is enabled and therefore this action (enable audio description) is considered finished.

Another abstract task „Caption" is made up of a set of interactive and abstract tasks. This overall task has the temporal binary relationship independent concurrency, as a result, it can be executed at any time and in any order. Besides that, this task presents three levels.

The first level of this abstract task is made up of an interaction task „Caption" (which includes the AA01 requirement) that has two temporal unary relationships, optional and iterative, and an abstract task „ActionCaption" that includes the second level of the overall task.

The tasks regarding the second level have a temporal binary relationship between themselves called enabling where the end of the task „Caption" (which means the enabling of this task) enables the other set of tasks of this level (the word search and the captions settings). The „ActionCaption" task is made up of an interaction task „Find" that includes the AF01 requirement (which allows user to seek words within the reproduction) which is optional and iterative and a set of tasks „Settings". Both of them present an independent concurrency as a temporal binary relationship because they can be carried out at any time and in any order.

Within the group of settings (the third level), a set of optional and iterative tasks can be found: "Size", „Font", „Color" and „LanguageCaption", which include the requirements AA03, AA04, AA05 and AA06 respectively together with the relationships of independent concurrency among them that allow users to change the size, color, font and language of the captions.

The semantics of helping users is gathered through the interaction task „Help" following the requirement AH01. For this reason users can obtain information on accessibility features that allows them to interact with the user agent in a proper and satisfactory way.

4.3.2. Abstract User Interface

The AUI is a model independent of technology and modality. It starts from the Task model (see Figure 24).

The UsiXML AUI is made up of Abstract Containers (ACs), Abstract Individual Components (AICs) and the abstract relationships between them. AIC represents basic system interactive functions: input, output, navigation and control. In this sense, AICs are an abstraction of widgets found in graphical toolkits (such as windows, buttons) and in vocal toolkits (such as vocal input and output widgets in the vocal interface). Therefore, requirements are grouped into six AC (that include AICs, abstract relationships and/or ACs) and seventeen AICs.

As far as ACs are concerned, the AC AudioDescriptionGroup is made up of two AICs (requirement AA02 (AudioDescription) and requirement AA07 (LanguageAudio)).

The AC CaptionGroup is made up of one AC (ActionCaption) and one AIC (requirement AA01 (Caption)). For its part, this AC is also made up of one AIC (requirement AF01 (Find)) and antoher AC called Settings which is composed of four AICs (requirements (Size), AA04 (Font), AA05 (Colour) and AA06 (LanguageCaption)).

The last general AC is the Playback which is made up of an AIC (requirement NP01 (Play)) and one AC (Actions). This last AC is composed of four AICs (requirements NP02 (Stop), NP03 (Pause), AP01 (Rewind) and AP02 (Forward)).

Among the AICs, nine requirements (NP01 (Play), NP02 (Stop), NP03 (Pause), AP01 (Rewind), AP02 (Forward), NS01 (Resize), NV01 (Mute), AA01 (Caption) and AA02 (AudioDescription)) are made up of a control element. The functionality of these interaction objects allows the user to interact with the user agent through playing, stopping, pausing, rewinding, forwarding, resizing, muting the volume, showing captions without modifying any of their features or allowing to listen to the audio description.

The functionality of the requirement NV02 (Volume) is different. In this case, users can increase or decrease the volume of the playback, so this requirement needs an input element that allows the user not only to interact with the user agent but also to modify its current value.

On the other hand, requirements related to captions or audio description settings are made up of two elements, an input and an output. This is because the semantics of these characteristics can change the settings of the captions (requirements AA03 (Size), AA04 (Font), AA05 (Colour) and AA06 (LanguageCaption)) or the audio description (requirement AA07 (LanguageAudio)) by selecting a value within a list of values as has already been mentioned.

Other requirement AF01 (Find) presents an input, an output and a navigation element because the interaction is achieved when the user establishes a word, the user agent seeks this word within the captions and then shows the corresponding caption within the playback.

An last but not least, the AH01 (Help) requirement is made up of at least an input, an output and a control element in order to allow users to select and show the help needed when stopping the playback.

This composition of ACs and AICs is essential to follow the correct order of implementation where NP01 (Play) has to be enabled before other requirements (NP02, NP03, AP01 and AP02).

4.3.3. Concrete User Interface

Until now, the models have not been referred to a particular device capability, interaction modality or implementation technology. From now on, the models are platform-dependent but language independent (see Figure 23).

Thanks to the previous abstract levels, the interaction between a user and a user agent and the structure of this interaction are established. Therefore, in order to finish the design, these abstract elements are transformed into concrete elements. Due to the difficulty of using this type of meta-models by designers and to accomplish the third level of Cameleon Reference Framework, the idea of providing a graphical editor arises [133]. This editor facilitates the design process and provides support to novel designers in the area of accessibility. The editor development is supported by two plugins of Eclipse framework, Eclipse Modeling Framework (EMF) and Graphical Modeling Framework (GMF). On the one hand, EMF is used in order to replicate some parts of the CUI, in this case, a restricted CUI using design primitives related to media players and general design primitives. On the other hand, GMF is used to generate the graphical editor.

Due to the fact that the CUI is platform dependent, at this point it is fundamental to choose a platform. In this case, UsiXML presents different modalities (graphical, vocal, multimodal) in order to start the development of the editor.

The CUI allows the definition of the specification of the appearance and behavior of a UI for a given context of use [134]. A CUI is composed of Concrete Interaction Objects (CIO) and Concrete Relationships (CR).

In this level, a restricted CUI meta-model was created with EMF in the domain of accessible media player design. This meta-model includes only the design primitives required for this domain. Specifically, a CUI describes a potential user interface after selecting a particular interaction modality (graphical, vocal, multimodal) [135]. It was decided to use the graphical modality for this approach, since it was essential that the user should be able to interact via external devices, such as keyboards or mice.

Based on the graphical modality of the CUI meta-model, the accessibility requirements established in Section 3.4 and a review of some media players to understand their traditional functionality in depth, all the design primitives were selected. Some of them were selected indirectly (for example, the design primitives related to listener), while other primitives were established directly. These primitives include: ToolBar, ToolBarButton, ToolBarSeparator, CommandButton, Slider, ComboBox and ComboItem, which are related to accessibility; and primitives such as VideoComponent, AudioComponent, ProgressionBar, Menu, MenuItem, MenuSeparator, MenuBar and MenuBarItem, which provide the traditional functionality of a media player.

Four abstract concepts were distinguished to define the accessibility requirements using the design primitives of the EMF meta-model. These allow a model-based accessible player to be designed using the graphical editor:

- First concept: define a requirement type where a user action triggers another action during the playback of the video content that directly affects the playback (e.g. the video content is played, stopped, paused, etc. as a result of this action). The Play („NP01"), Stop („NP02"), Pause („NP03"), Rewind („AP01"), Forward („AP02"), Resize („NS01"), Mute („NV01"), Caption („AA01") and AudioDescription („AA02") requirements can be defined using this concept. This concept can be defined using the ToolBarButton design primitive; therefore, the ToolBar and ToolBarSeparator design primitives are also needed.
- Second concept: define a requirement type where a user action triggers an action after stopping the video playback. The Help („AH01") requirement (which shows additional information) and Find („AF01") requirement (which searches a caption within the playback), can be defined using this concept. In this case, the CommandButton design primitive is used.
- Third concept: define a requirement type where a set of options is shown. The Size („AA03"), Font („AA04"), Colour („AA05") and LanguageCaption („AA06") requirements (which are related to captions) and the LanguageAudio („AA07") requirement (which is related to audio description) can be defined using this concept. This concept can be defined using the ComboBox and ComboItem design primitives.
- Fourth concept: define a requirement type where an element"s value is increased or is decreased. Requirement NV02 („Volume") can be defined using this concept. In this case, a Slider design primitive is used to define the concept.

Table 2 Mapping between accessibility requirements and design primitives of the CUI meta-model for modeling with the EMF meta-model

Requirement code	Requirement Name	Design primitives of UsiXML CUI model	
NP01	Play	ToolBar	
NP02	Stop		
NP03	Pause		
NS01	Resize		
NV01	Mute		
AP01	Rewind	ToolBarButton	ToolBarSeparator
AP02	Forward		
AA01	Caption		
AA02	AudioDescription		
AH01	Help	CommandButton	
AF01	Find		
AA03	Size	ComboBox	ComboItem
AA04	Font		
AA05	Colour		
AA06	LanguageCaption		
AA07	LanguageAudio		
NV02	Volume	Slider	

Table 2 shows all the accessibility requirements mapped to the design primitives. Depending on the types of design primitives, the requirement types and even the designers" preferences, the number of design primitives could or could not be determined. For example the ToolBar and Slider design primitives are always the same; however, the ToolBarButton, ToolBarSeparator, CommandButton and ComboBox design primitives depend on the number of requirements, while the ComboItem design primitive depends on the designer preferences.

In conclusion, Table 2 shows a summary of which design primitives of the CUI meta-model of UsiXML are used to define the accessibility requirements. An EMF meta-model was thus obtained for the accessible player domain starting from this restricted CUI meta-model. This EMF was the first step in the development of a Graphic editor with GMF (see Figure 25).

Figure 25 shows the CUI elements of the UsiXML restricted CUI.

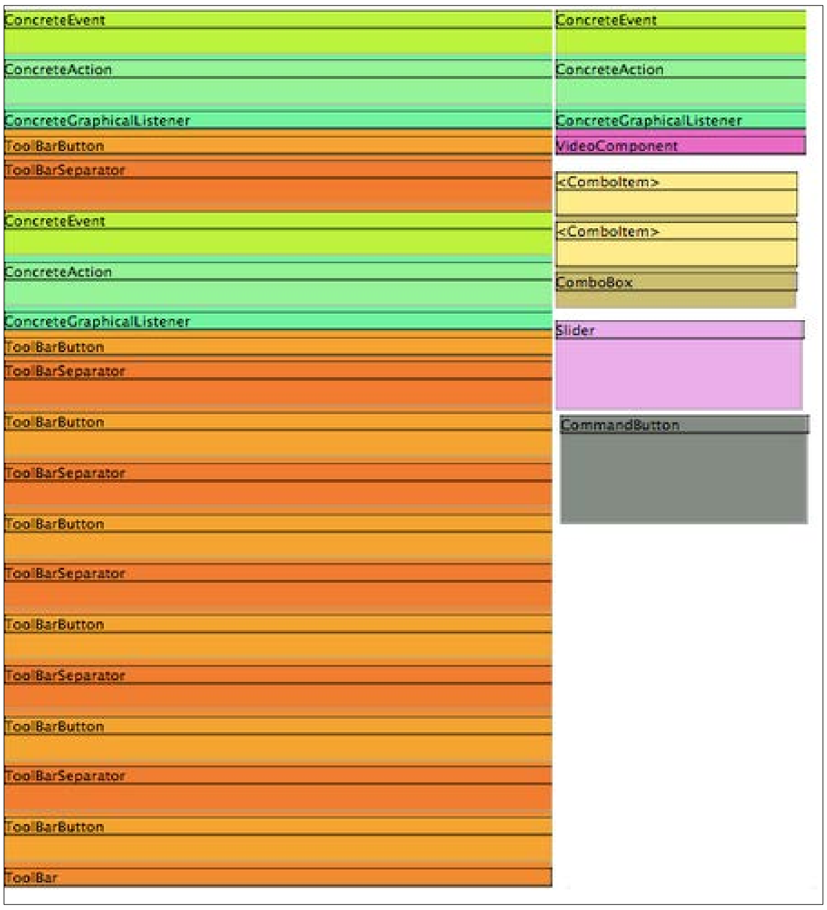


Figure 25 Representation of CUI elements in the graphical editor

4.3.4. Support for accessibility using the graphical editor

So far, the integration of accessibility requirements in the graphical editor has been described. These requirements are focused on the functionalities of the media player and its features, as well as providing the possibility of applying adaptations. In order to provide a proposal of solution as accessible as possible, which includes the largest number of accessibility guidelines, a review of accessibility guidelines is made. With this review, the accessibility guidelines which have not been included in the design are detected and are integrated in the editor through constraints of the design primitives or through messages to guide to designer in the accessible design, for instance, accessibility requirements to comply with the image which the designer should use in every button, to provide keyboard shortcuts or to maintain the accessibility features set by users among different sessions.

4.3.5. Final User Interface

The generation of the FUI from the CUI model of UsiXML has been implemented via an XSL transformation. An option to perform this process has been integrated in the editor (see Figure 26). For this work, the HTML5 language was selected for generating the FUI. This language provides native support for embedding video elements without the need of installing third-party plugins and, in addition, it is supported by the most widely used platforms.

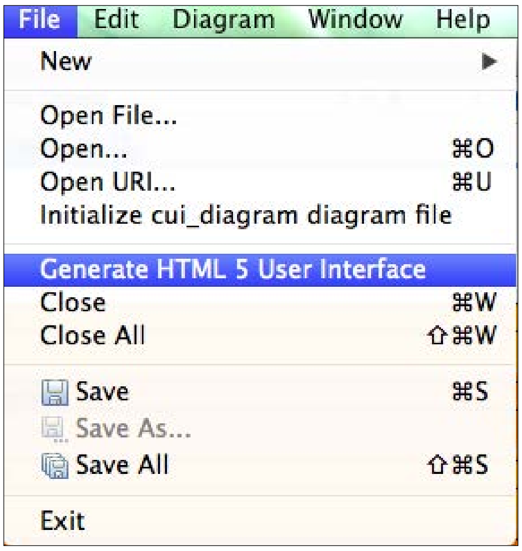


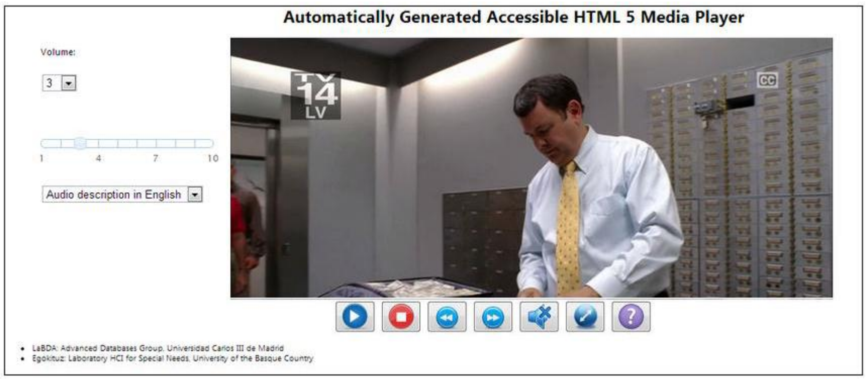
Figure 26 Generation of HTML5 User Interface

As mentioned in the previous subsection, the elements of the CUI have been labelled with their matching semantic, providing the information necessary to perform the transformation. Table 3 illustrates how the CUI elements have been mapped to the HTML5 methods. It must be pointed out that since HTML5 does not provide a specific element for rendering sliders, an accessible slider element [136] (labelled with WAI-ARIA [137]) based on jQuery UI library [138] has been used.

Table 3 Mapping between CUI elements and HTML5 methods

CUI elements	HTML 5 elements
ToolBar	Div | Menu + Type attribute
ToolBarButton	Button + Image | Command element (inside menu element)
VideoComponent	Video
ConcreteGraphicalListener	JavaScript function
Slider	jQuery UI slider
ComboBox + ComboItems	Select + Options+track
CommandButton	Button
ToolBarSeparator	CSS Layout | Separator (<hr>) (inside menu element)

Figure 27 shows the result of the generation process. As can be seen, the elements required for an accessible media player have been integrated in the UI. In addition to the controls that have been implemented using non-intrusive JavaScript, the default controls of the video element have been kept. This is because otherwise users with agents that have JavaScript disabled would not be able to interact with the video content. For this work, only basic default CSS rules have been integrated, just for aligning the components in the UI.


Automatically Generated Accessible HTML 5 Media Player
Volume:
3
1
4
7
10
Audio description in English
TV 14 LV
CC

- LaBDA: Advanced Databases Group. Universidad Carlos III de Madrid
- Egokituz: Laboratory HCI for Special Needs. University of the Basque Country

Figure 27 FUI developed with HTML5

4.4. Second approximation: MARIA framework

As was the case in the previous section, this approximation is also structured according to Cameleon Reference Framework and uses MARIA framework as UIDL (see Figure 22 (b), Figure 28). As it is going to be explained in the following paragraph, only the most abstract models of the Cameleon Reference Framework (the Task model and the Abstract User Interface model) have been accomplished. Annex III shows the meta-models used in this second approximation.

The reason of developing only the two first levels of Cameleon Reference Framework is that although there are tools that help designers and developers in order to complete the four abstraction levels, these tools do not allow introducing adaptation rules in a direct way.

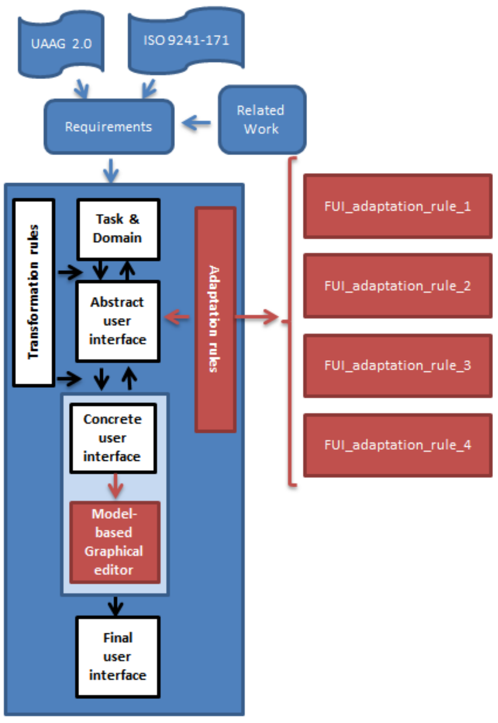


Figure 28 Proposal of architecture of the methodological approach using MARIA

As can be seen, this proposal of architecture starts from the proposal of architecture which uses UsiXML (see Figure 22 (a), Figure 23) due to the fact that its more abstract level, the Task model, is the same. Besides, because of the difficulty of introducing adaptation rules, this architecture has a support in the form of a graphical tool (see Chapter 5) in order to facilitate the design tasks.

4.4.1. Task Model

The task model developed using MARIA framework has only a difference regarding the same task model obtained in Seccion 4.3.1. This difference is related to the tool used in order to develop the model. In this case, it is used ConcurTaskTree Environment (CTTE) [139] instead of IdealXML as happened in the previous aproximation.

4.4.2. Abstract User Interface

This model uses presentation task sets (PTSs) to meet the semantic established regarding the temporal binary relationship set in the previous model. A PTS is a set of basic tasks that should be associated with a given presentation [140, 141], that is, the set of elements enabled at the same time. Therefore, the binary relationships which want to be met are:

- Activate the option to change the language of the audio description after activating the audio description.
- Activate the set of actions regarding captions (Find, Size, Color, Font and LanguageCaption) after activating the captions.
- Activate the set of actions (Stop, Pause, Rewind and Forward) after playing the playback.

Hence, the PTSs established are four:

- Set 1: {Resize, Mute, Volume, AudioDescription, Caption, Help, Play}
- Set 2: {LanguageAudio}
- Set 3: {Find, Size, Color, Font, LanguageCaption}
- Set 4: {Stop, Pause, Rewind, Forward}

After including the PTSs, the automatic transformation into AUI elements, starting from the task model, is done. Generally, the set of tasks created in the task model are transformed into a set of Interactors (selection, edit, only output and control) which represent every type of user interaction object and Interactor Compositions (grouping, relation, composite description and repeater) which groups together elements that have a logical relationship [142] in the AUI model. In this case in particular, Table 4 shows the types of interactor used and the accessibility requirements established in the Task model.

Table 4 Interactors and requirements

Selection Single Choice	**Edit Numerical**	**Control Navigator**	**Control Activator**	**Control Activator/Edit Text Field**
Language Audio, Size, Color, Font and Language Caption	Volume	Help	Play, Stop, Pause, Rewind, Forward, Resize, Mute, AudioDescription and Caption	Find

According to the interactor compositions, the types of interaction composition which are used are grouping and ordering. This composition presents a generic group of interactor elements [142].

In this second architecture, adaptation rules are going to be integrated. This integration allows designers to create adaptive user interfaces according to the special needs and preferences of users which provide, for example, simplified interfaces removing controls, such as buttons, which are not used by users.

4.4.3. Integrating adaptation rules

Prior to establish the rules, a basic analysis of types of access and groups of users with disabilities is done in order to define types of adaptations [143]. The analysis is based on the following statements taking into account the diversity of users that exists [144]:

1) Some people who access to the visual and auditory information may prefer to access a simplified user interface, wherein the user interface presents only those elements essential to access playback (controls, in this case, buttons).
2) People who are deaf or have a hearing loss can access the auditory information within the synchronized media content through captions.
3) People who are blind or have low vision can access the visual information within the synchronized media content through audio description.
4) Some people who access to the visual and auditory information may prefer to access a user interface that includes all the elements in the user interface according to accessibility requirements.

Table 5 shows the summary of the outcomes of this analysis.

Table 5 Analysis of the elements required by types of access / user groups

Types of access / groups with disabilities	**Resources necessary to access**	**Required accessibility elements**	**Rule Code**
Access visual and auditory/ simplified user interface	(essential) Playback and volume elements	Buttons: Play/Pause, Stop, Rewind, Forward and Mute Slider: Volume	1
Access visual / Auditory impairment	Playback, help, volume and caption elements	Buttons: Play/Pause, Stop, Rewind, Forward, Mute and Caption Select menu: Font, Color, Size and LanguageCaption Navigator: Find and Help Slider: Volume	2
Access Auditory / Visual impairment	Playback, help, volume and audio description elements	Buttons: Play/Pause, Stop, Rewind, Forward, Mute and AudioDescription Select menu: LanguageAudio Navigator: Help Slider: Volume	3
Access visual and auditory/ user interface with all the elements	Playback, help, volume, caption* and audio description** elements	Buttons: Play/Pause, Stop, Rewind, Forward, Mute, Caption* and AudioDescription** Select menu: Font*, Color*, Size*, LanguageCaption* and LanguageAudio** Navigator: Find* and Help Slider: Volume	4/5

*If user wants to use captions
**It user wants to use audio description

In order to present the rules, an Event-Condition-Action (ECA) rules are used. The general syntax of these rules is [145] (see Table 6):

Table 6 Syntax of ECA rules

Syntax of the rules
on event if condition do actions

Therefore, the event part specifies when the rule should be triggered. The condition part is a query which determines if a certain state occurs. And the action part states the actions to be performance automatically if the condition holds.

Considering the analysis of the elements required by types of access / user groups, five adaptations rules have been defined (see fourth column of Table 5) together with the general syntax of the ECA rules (see Table 6), the following rules have been established (see Table 7, Table 8, Table 9, Table 10 and Table 11):

Table 7 Adaptation rule 1

Rule 1
EVENT: the UI is activated. CONDITION: the user does not have any kind of disability. ACTION: all menus except the playback menu and the volume are disabled. Description: the rule is triggered when the UI is activated; then, it checks if the user wants to use a basic media player; finally, only the playback menu and the volume are shown.

Table 8 Adaptation rule 2

Rule 2
EVENT: the UI is activated. CONDITION: the user has any kind of auditory impairment. ACTION: the menu related to audio description is disabled. Description: the rule is triggered when the UI is activated; then, it checks if the user has any kind of auditory impairment; finally, the audio description menu is disabled.

Table 9 Adaptation rule 3

Rule 3
EVENT: the UI is activated. CONDITION: the user has any kind of visual impairment. ACTION: the menu related to caption is disabled. Description: the rule is triggered when the UI is activated; then, it checks if the user has any kind of visual impairment; finally, the caption menu is disabled.

Table 10 Adaptation rule 4

Rule 4
EVENT: the UI is activated. CONDITION: the user does not have any kind of disability. ACTION: the menu related to caption is disabled depending on the user preferences. Description: the rule is triggered when the UI is activated; then, it checks if the user wants to disable the menu; finally, the caption menu is disabled.

Table 11 Adaptation rule 5

Rule 5
EVENT: the UI is activated. CONDITION: the user does not have any kind of disability. ACTION: the menu related to audio description is disabled depending on the user preferences. Description: the rule is triggered when the UI is activated; then, it checks if the user wants to disable the menu; finally, the audio description menu is disabled.

This set of rules will allow designers to adapt the final user interface according to the needs and access preferences of the user, from the initial stages of the design process.

4.5. Conclusions

In this chapter, two approaches have been presented. As aforementioned, this fact has allowed to, on the one hand, extend this proposal to a greater number of designers with expertise in modeling, and on the other hand, the possibility of simplifying the design process when adaptation rules are included.

Both approaches are structured according to the four levels of abstraction of Cameleon Reference Framework. Therefore, this fact has allowed to use the same first level of abstraction (the Task model) as their starting point.

On the one hand, the first approach allows developing this PhD design proposal through model driven development (MDD) using UsiXML like UIDL. On the other hand, the second approach is developed using MARIA language and therefore, this approach follows a model-based user interface development (MBUID).

As it can be seen, this proposal can be accomplished using both approaches, the choice of one of them depends on the experience or the facilities which the approach provides in every moment.

In this chapter, it is presented a solution proposal through meta-models. Therefore, this proposal is oriented to modeling designers. In the following chapter (Chapter 5), an authoring tool support is presented in order to facilitate the design of an user agent which delivers video content to experts and no experts in the field of modeling or even accessibility.

Chapter 5. Authoring tool support

5.1. Introduction

In this chapter, it is presented an authoring tool based on the PhD proposal of design and development of an accessible media player. This tool is a resource whose aim is to guide professionals independently of their expertise in MDD, MBUID and accessibility. Therefore, in order to carry out this proposal, all the knowledge obtained in Chapter 3 and Chapter 4 is going to be utilized. This means that thanks to this knowledge, a set of concrete objects is going to be obtained starting from the design primitives achieved through the modeling approaches presented in Chapter 4.

At this point an authoring tool, in this case a graphical editor, is developed to allow the design of an accessible media player (see Figure 29). The reason of choosing this type of authoring tool is the possibility to include design primitives through constructors with graphical elements which are located within the editor.

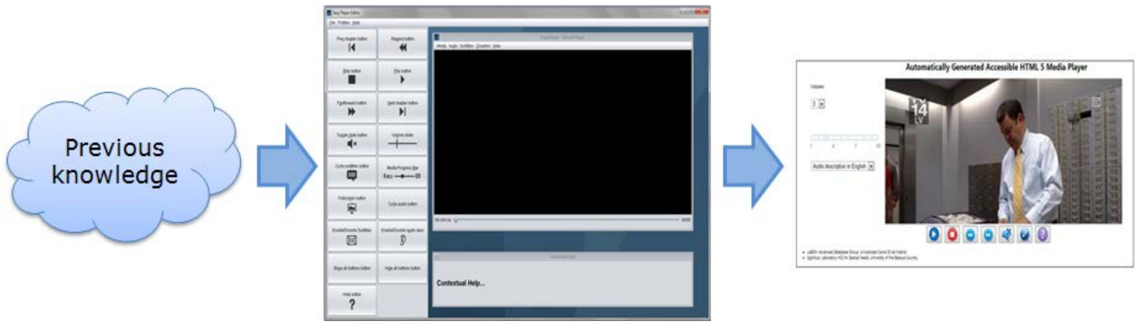


Figure 29 Authoring tool as support to the requirements" abstraction

To sum up, in order to accomplish the proposal of this Doctoral Thesis, in this chapter the two following main aims are going to be completed:

- Development of an authoring tool which contains all the design primitives of the PhD proposal neccesaries so as to design an accessible media player.
- Generation of a real accessible media player which everybody can download and run in different locations to play any kind of video.

Therefore, the design process composed of four parts (analysis, design, deployment and evaluation) is going to be described in general terms.

Before starting with the explanation of the design process, it is important to emphasize that this development is based on the previous studied approaches, then the architecture used is also composed of four levels (see Figure 30).

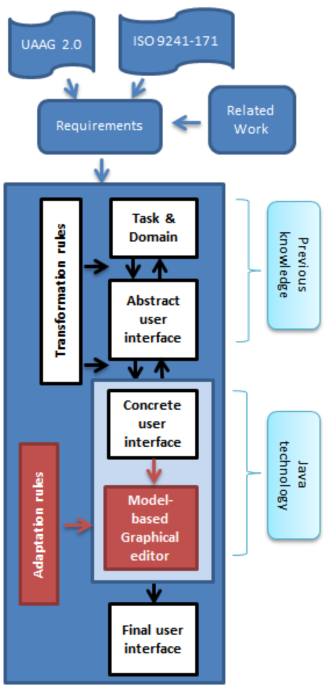


Figure 30 Proposal of architecture based on previous knowledge

As it can be seen, the two more abstract levels are the same as the previous knowledge, whereas the concrete level uses the design primitives which are dependent of platform and technology. In addition to provide the PhD proposal based on models and, in order to give this proposal support and facilitate the design of an accessible media player, the design primitives of the concrete level are included as constructors within a graphical editor.

This editor also includes the set of adaptation rules established previously (Chapter 4). Thanks to these rules, the final user interface (the media player) can be adapted to different necessities depending on the final user of it.

5.2. Analysis phase

The aim of this PhD proposal based on models is to present the design process of an authoring tool. This authoring tool helps designers and developers without any expertise in the field of modeling and accessibility to generate an accessible media player.

Therefore, in this section, first of all, some accessibility standards which are going to be considered to design the accessible authoring tool will be presented. Secondly, it is also studied and are going to be shown a set of technologies which can be used to accomplish the development. And finally, a functional requirements elicitation to collect the functional requirements of the system is going to be introduced.

In order to avoid misunderstandings, in this chapter, the person who designs the media player using the graphical editor is denoted like author, whereas the person who will use the accessible media player is denominated final user.

5.2.1. Accessibility standards

The authoring tool has to assist the author or at least to guide the development of an accessible player. Due to the fact that an authoring tool, but above all, the content that this tool provides, in this case a media player, have to be accessible, some accessibility standards have to be fulfilled.

The issue is what accessibility implies. In both cases, that are the case of the authoring tool and the case of the content provided by the tool, accessibility means to be used and accessed by users with or without disabilities. Therefore, due to the different handicaps, there are many types of barriers which have to be overcome.

In order to be accessible, the content provided by the tool has to fulfil the accessibility requirements set in Chapter 3 and consider accessibility standards such as the User Agent Accessibility Guidelines (UAAG), Web Content Accessibility Guidelines (WCAG) and ISO 9241:171.

Furthermore, so as to be accessible, the tool has also to fulfil the Authoring Tool Accessible Guidelines (ATAG). On the one hand, the ATAG provides guidelines for designing authoring tools which are accessible to author with disabilities. On the other hand, it encourages to provide a design which enables, supports and promotes the production of more accessible web content.

Therefore, the development of an authoring tool is necessary in order to extend the scope of the proposal to all kind of people (with or without disabilities, experts or not in accessibility or modeling). Besides, it is needed that the content provided by the tool can also be accessed by them.

Regarding the first aim of the ATAG, the authoring tool has to offer a friendly interface which fulfils the WCAG, for example, all the functionality has to be operable through different elements such as the keyboard or the mouse. Moreover, it is also essential to be compatible with assistive technologies such as screen readers, magnifiers or speech synthesizers among others in order to avoid the exclusion of people with disabilities.

According to its second goal, in order to assist the author, the authoring tool offers a contextual help as a mechanism which guides the author through the design process. Therefore, this mechanism allows the authoring tool to help the authors even if they do not have expertise in the design process of an accessible media player.

And last but not least, it is important to highlight that this tool also has to be adaptable depending on the preferences of the final user, that is, the designer is going to be able to select different user profiles, which are related to the age or type of disability of the user, regarding the user for whom the media player will be designed. These profiles are obtained starting from the adaptation rules which have been included in the proposal of this Doctoral Thesis in Chapter 4.

5.2.2. Accessible software technology

Due to the fact that it is going to be used technology which allows implementing the authoring tool that fulfils the accessibility standards shown previously. In this subsection, different reasons have been given to confirm the final selection.

There are different types of technologies in order to develop this type of authoring tool. On the one hand, there are technologies which develop desktop applications (such as Java, Visual Studio, .NET, C Sharp). The applications developed by these technologies are multiplatform applications. On the other hand, Web applications are applications which can be used by users accessing a Web server through the Internet or an intranet via a browser and are developed by technologies such as Perl, HTML5, jQuery, php or Javascript.

Although both types of technologies have advantages and disagvantages, the final technology selected in order to develop the authoring tool is Java. This high-level programming language is selected due to all the advantages which provides. Among these advantages, it can be found that Java is object oriented, platform independent, simple, secure, portable and robust. If the downsides are taken into account, the main disadvantage of Java is the speed of the programs developed with it.

Other crucial feature of Java is that, apart from being one of the most popular programming languages worldwide [146, 147, 148], it also provides accessibility support. Accessibility on the Java platform is built in four areas: Java Accessibility API (JAAPI), Java Accessibility Utilities, Java Access Bridge (JAB) and Java Foundation Classes (JFC):

- JAAPI defines a contract between user-interface components and an assistive technology that provides access to those components. If a Java application fully supports the JAAPI, then it should be compatible with and friendly toward assistive technologies such as screen readers, screen magnifiers, etc [149].
- Java Accessibility Utilities [150] provides the ability to get the information from the application and processess it for further displaying with special devices. Therefore, it helps assistive technologies to take advantage of applications developed using the JAAPI.
- Java Foundation Classes is a library of GUI components which fully implements the JAAPI [151].
- JAB is a way to communicate accessible tools which are not developed in Java with Java applications [152].

Therefore, all the accessibility features and all the advantages which Java provides are the main reasons to choose this technology.

5.2.3. Functional requirements

It is essential to consider other requirements which are related to the tool. On the one hand, there are non-functional requirements associated with the accessibility and usability of the interface which take into consideration how accessibility is going to be achieved. While on the other hand, there are functional requirements regarding the functions which the tool accomplishes.

As far as non-functional requirements are concerned, in order to achive the goal of developing an accessible and usable interface, it is crucial to bear in mind different aspects of the interface.

First of all, it is necessary to assure the access to the interface through assistive technology such as screen reader. This fact can be made thanks to the use of the accessible technology explained in Section 5.2.2 and especially Java Access Bridge package.

Secondly, the interface should be simple, that is, without many colours which can confuse users or even complicate their access due to a dreadful choice of the contrast of the interface.

Thirdly, the interface must be used and controlled by users, therefore, taking into account accessibility and usability, users can enable or disable and adjust the accessibility features (which it is achieved through the different controls provided by the interface).

Fourthly, it is important to provide information about keyboard shortcuts and keyboard combination which is supplied via a help menu and through the different options of the menus. And last but not least, users must be able to move across the interface using the keyboard and the mouse without this means activate any control.

Regarding functional requirements, these requirements are those which define the functions and its components. Figure 31 shows a use case which represents the set of actions that can be accomplished in the tool.

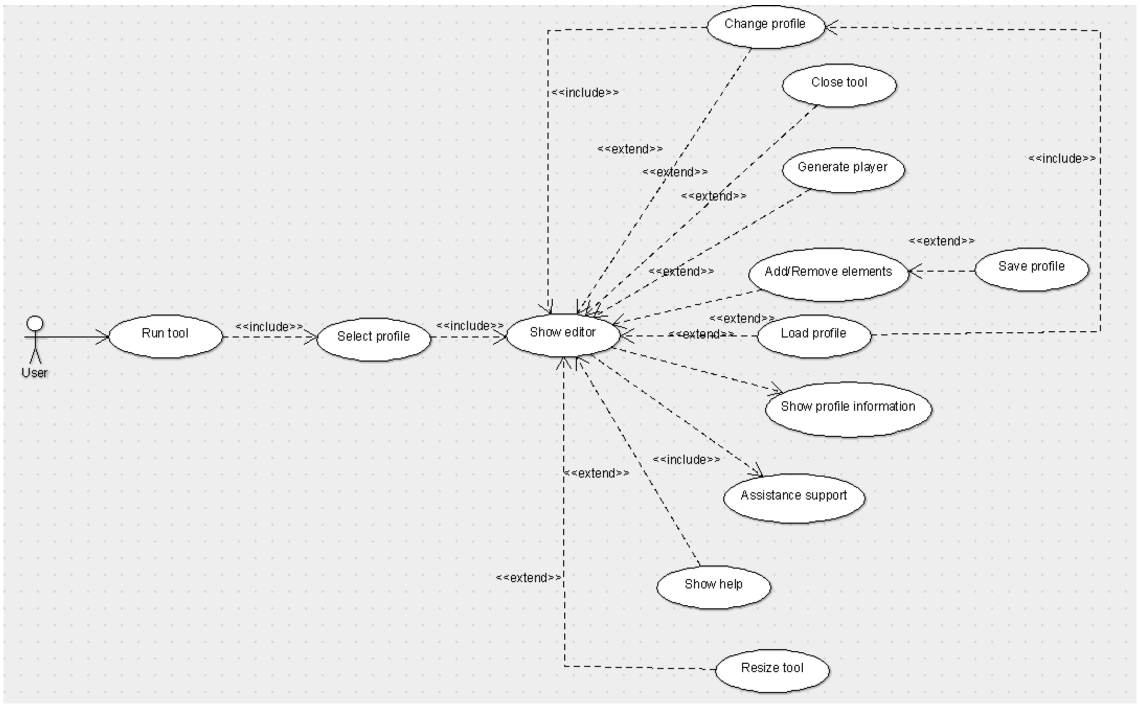


Figure 31 Tool functionality use case

After running the tool, the user has to select a profile. Every profile has been established according to the adaptation rules set in Chapter 4. For example, if the user wants to design a media player to be accessed by blind people, the profile which has to be selected is “Auditory Access Player”. This profile represents the adaptation rule coded as number 3, “Access Auditory/Visual Impairment” (see Table 5 in Chapter 4).

Once the profile has been selected, the graphical editor and the assistance support (which is the contextual help) are shown. The kind of elements inside the interface depends on the profile selected. The graphical editor is the completed interface, whereas the contextual help is a part of the interface in which the editor is giving the author support through the design process.

At this moment of the process, the author can choose differents options such as change the profile, close the editor, generate the player, add or remove elements, load a profile, show profile information, show help and resize the tool. For example, if the author wants to change the profile, the graphical editor is shown again and the author has to choose an option again.

Other option is to generate the final player. After designing the media player, the author can want to generate it in order to use the standalone media player outside the graphical editor.

If the author adds or removes elements from the media player, she/he can save this profile to use it when she/he wants in the future. Loading a profile is also related to save a profile, because the user can only load a profile which has been saved previously. This action includes to change the present profile and of course to show the editor again.

There are two options which show information. On the one hand, if the author wants to be informed about the established profiles provided by the graphical editor, she/he can obtain this information through the profile information option which is located in the profiles menu. On the other hand, if the author wants to be informed about the different shortcuts which she/he can use, the help menu of the editor is utilized. Moreover, the author can select other two options, close the tool and resize the tool, which simply is to close and maximize the interface.

Basically, the behavior of the graphical editor tries to facilitate the design process through simple and concrete actions. These actions guide the author to develop an accessible media player even though she/he does not have any knowledge about accessibility.

5.3. Design phase

In this phase, different aspects of the design taking into account the analysis phase are going to be presented.

First of all, it is important to highlight that this authoring tool is classified as a WYSIWYG ("What You See Is What You Get") editor. In general terms, the main components of the graphical editor"s interface are shown in Figure 32.

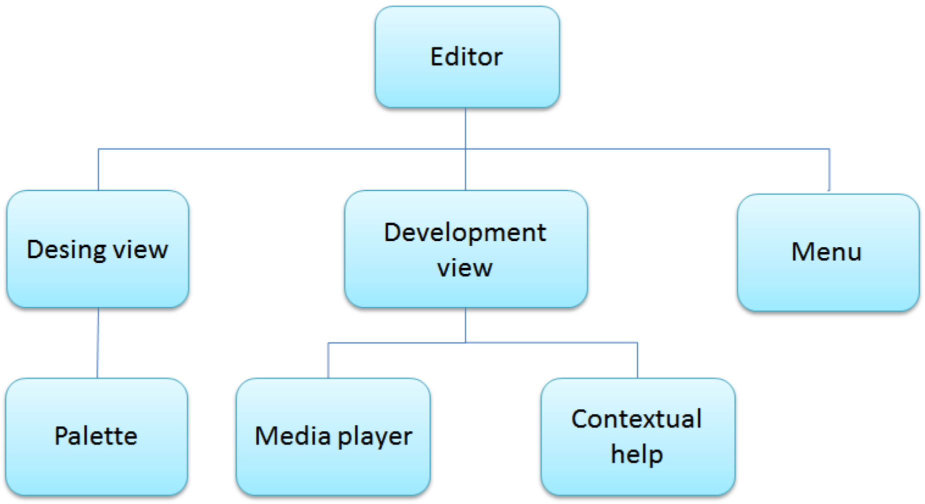


Figure 32 Components of the graphical editor´s interface

The final interface is composed of three main components: a design view, a development view and a menu. These components are going to be explained in the following subsections.

5.3.1. General view

Figure 33 shows the general view of the authoring tool interface.

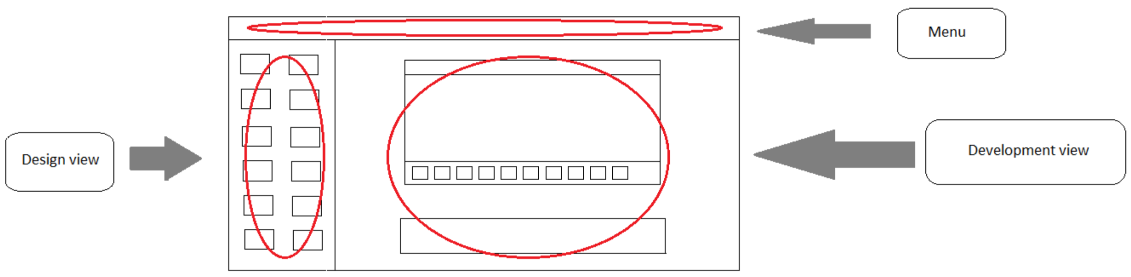


Figure 33 General view of the authoring tool

The upper menu offers different options which the system allows such as generate the player, exit, be informed, load, save or change the profile or the help option. The other components of the interface, the Design view and the Development view, are going to be explained in next subsections.

5.3.2. Design view

The Design view (see Figure 34) has a palette in which a set of design primitives (which represent the requirements identified in Chapter 3) have been established.

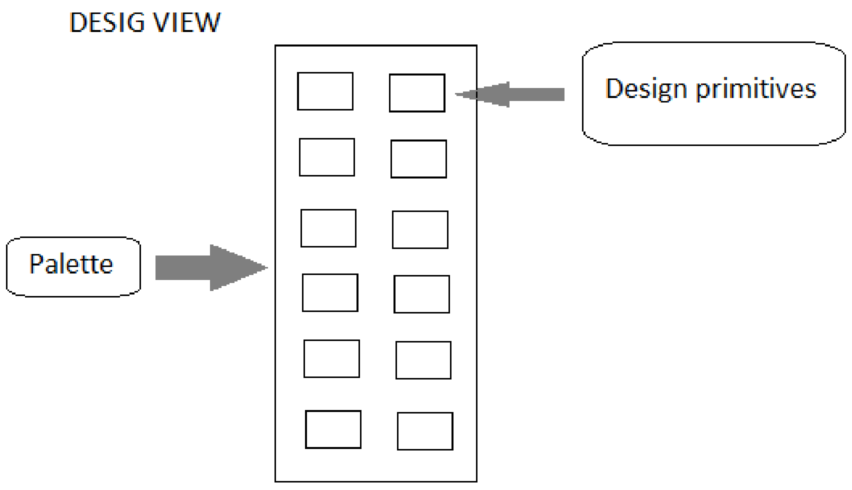


Figure 34 Wireframe of the Desgin view

Due to the final aim of the graphical editor is to create accessible content. To produce this type of content, the editor offers a group of design primitives which are based on the requirements shown in Chapter 3. Taking into account this fact, Table 12 shows the mapping between the requirements and the design primitives.

Table 12 Mapping between accessibility requirements and design primitives of the Java technology

Requirement code	**Requirement Name**	**Design primitives of UsiXML CUI model**		**Design primitives of Java technology**	
NP01/NP03	Play/Pause	ToolBar		JPanel	JButton
NP02	Stop				
NS01	Resize				
NV01	Mute				
AP01	Rewind	ToolBarButton	ToolBarSeparator		
AP02	Forward				
AA01	Caption				
AA02	AudioDescription				
AH01	Help	CommandButton			
AF01	Find				
AA03	Size	ComboBox	ComboItem	JMenu	JMenuItem
AA04	Font				
AA05	Colour				
AA06	LanguageCaption				
AA07	LanguageAudio				
NV02	Volume	Slider		JSlider	

In order to clarify the transformations between the requirements and the design primitives offered by Java technology, next, an example which illustrates this transformation is explained. For example, the Play requirement allows user to start the execution of the playback. Therefore, considering the functionality of this requirement, the design primitive selected to symbolize it is a JButton. This choice is due to the fact that a button allows accomplishing an action when it is clicked (see Figure 35).

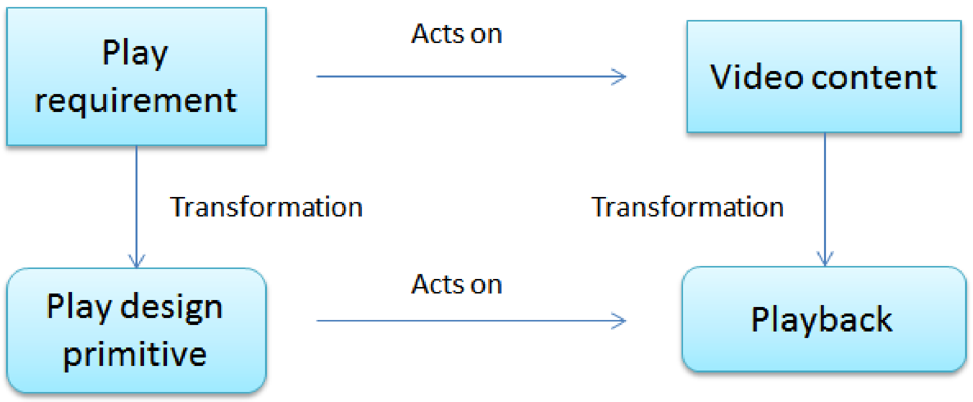


Figure 35 Description of the transformation between requirements and design primitives of Java technology

Likewise, all the requirements are transformed following the same scheme. Once the requirements have been transformed into the primitives, it is crucial to add more elements which are also important, but not essential to design an accessible media player. Among these elements it can be found full screen or progress bar. Thus, it is used a JButton in order to fulfil the functionality of resizing the screen and a JSlider to fulfil the functionality of the progress bar. Apart from these elements, there are others which have been established to add extra functionality and only appear in the palette such as the elements which allow showing or hiding all the buttons.

In order to sum the transformations between the requirements and the Java design primitives, in Figure 36, the Play requirement is going to be ilustrated. In this case, the Play design primitive is located in the palette of the Design view and is transformed into the Play button in the Development view.

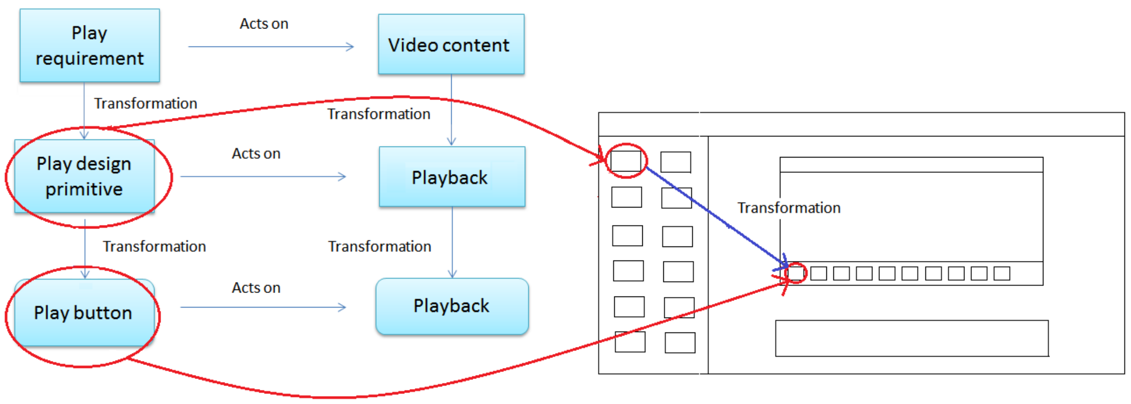


Figure 36 Example of Play transformation

It is also advisable to take into account the accessibility features which have been considered regarding aspects of the design such as the color or the simplicity of the interface among others.

5.3.3. Development view

The Development view (see Figure 37) is the part of the graphical editor in which it can be found the Media player designed and the Contextual help which gives the user support throughout the design process.

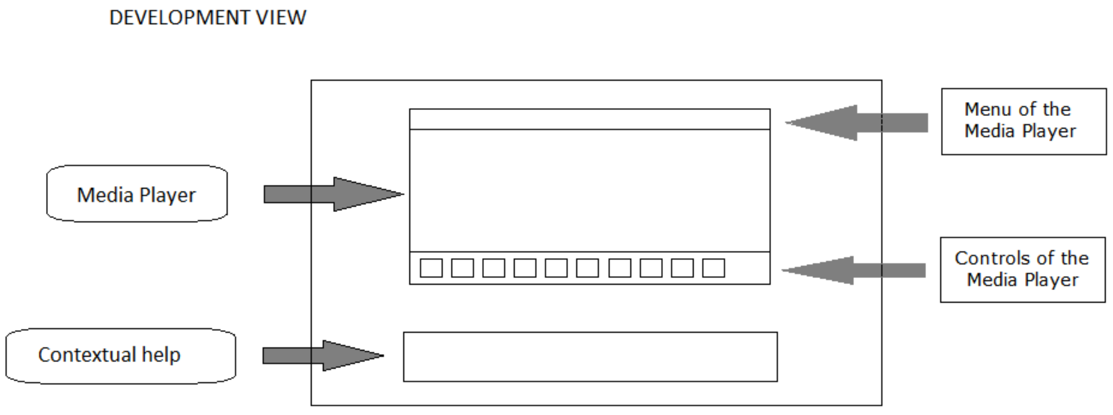


Figure 37 Wireframe of the Development view

The area labelled as Media player, is also composed of three main parts (see Figure 38), the upper menu with different submenus, the screen in which the video content is going to be visualized and finally, the lower menu in which the controls that allow acting on the video and which are the instances of the design primitives of the palette are located.

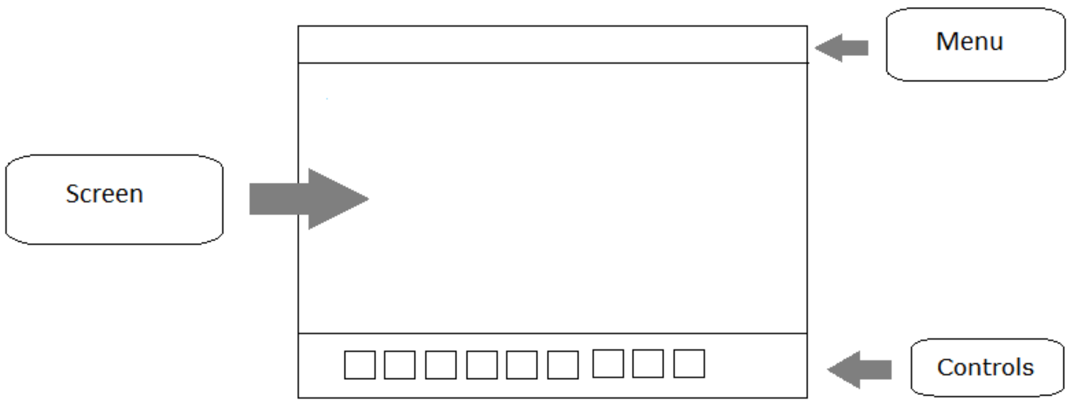


Figure 38 Wireframe of the Media player

The other area is the Contextual help. The graphical editor provides a contextual help so as to support the author about her/his steps during the design and about the missing objets to be an accessible design. This help tries to guide the author considering her/his decisions and acts in the design task facilitating this process.

5.4. Deployment phase

In this phase, the transition between the prototype and the final product is made. Therefore, it culminates with an executable version of the product, in this case, an executable version of a graphical editor.

Figure 39 shows the executable version of the graphical editor. As explained in Section 5.3, this interface is composed of three main parts: the design view, the development view and the menu.

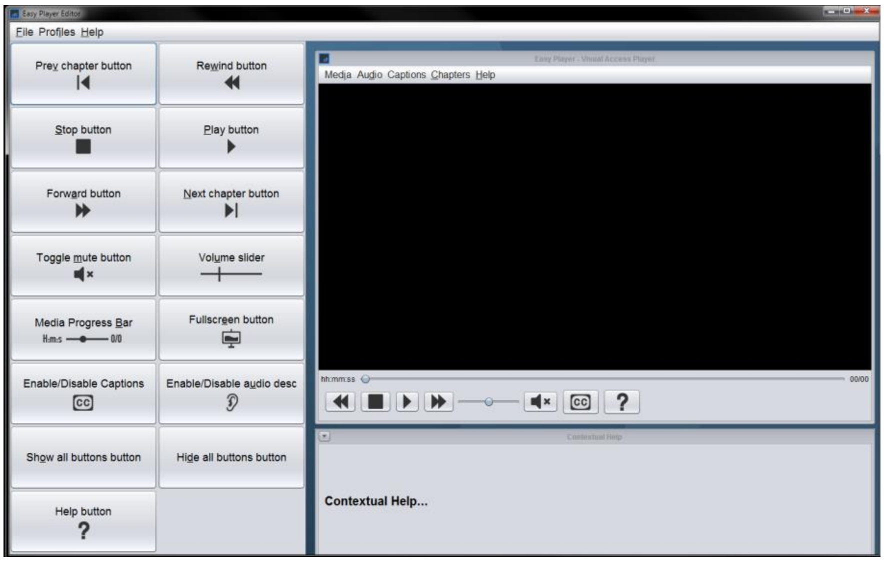


Figure 39 Screenshot of the graphical editor's final interface

The design view is created as a palette of elements (see Figure 40). This palette is composed of a set of constructors which have been selected in order to design the media player.

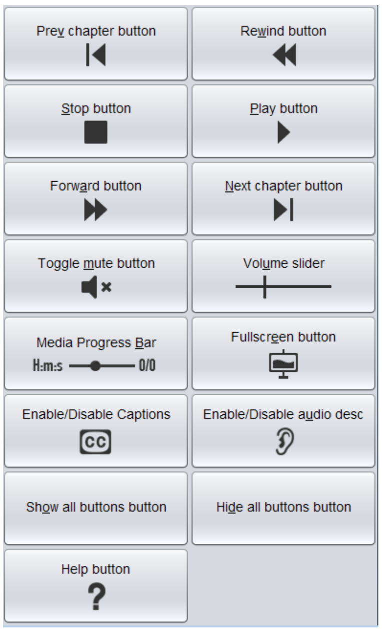


Figure 40 Screenshot of the graphical editor"s palette

The second part is a main panel which represents a kind of canvas (Figure 41). This canvas is composed of two internal frames. On the one hand, the frame in which the Media player is designed. And on the other hand, the frame which contains the Contextual help.

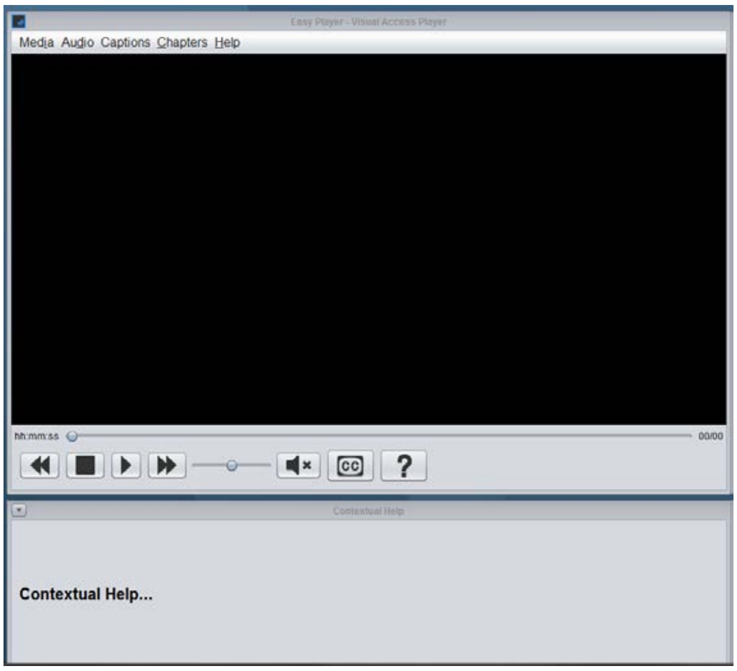

Easy Player - Visual Access Player
Media Audio Captions Chapters Help
hh:mm:ss
00/00
Contextual Help
Contextual Help...

Figure 41 Screenshot of the graphical editor"s canvas

The last part is the menu bar. This menu is the upper menu of the graphical editor (see Figure 42). As it can be seen, this menu is composed of different items such as File, Profiles or Help.

Figure 42 Screenshot of the graphical editor"s menu

Once the media player is designed, a video content can be displayed. Figure 43 shows a simplified media player. As described in Table 5 in Chapter 4, a simplified media player is a player which has only the necessary controls so as to display the video content (Play/Pause, Stop, Rewind, Forward, Mute and Volume). Besides these controls, the Progress Bar is also included in order to inform the final user about the progress of the video content.

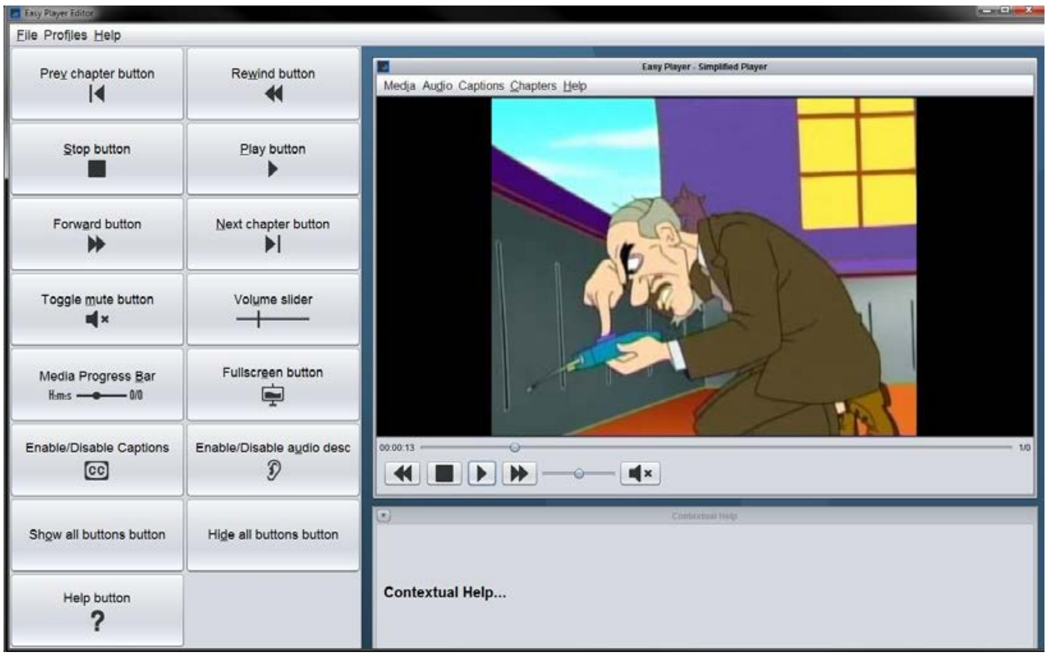


Figure 43 Screenshot of a simplified media player

5.5. Evaluation phase

In this chapter two evaluations, an accessibility evaluation and a functional evaluation, are going to be accomplished. Therefore, the following subsections explain both assessments.

5.5.1. Functional evaluation

The aim of this evaluation is to check if the functionality of the graphical editor is fulfilled, that is, if the editor does the actions for which the instances of the design primitives have been established.

Thus, a set of tests battery has been made in order to test that all the functionality established in the use case of Figure 31 is fulfilled.

Table 13 Requirement checklist

Requirements	Fulfilment			Observations
	Yes	No	N/A	
Play the playback	X			
Stop the playback	X			
Pause the playback	X			
Resize the window	X			
Turn on/off the audio	X			
Rewind the playback	X			
Forward the playback	X			
Turn on/off captions	X			
Turn on/off audio description	X			
Change the size of captions		X		The library used to develop the player does not offer this functionality for now
Change the font of the captions		X		The library used to develop the player does not offer this functionality for now
Change the color of the captions		X		The library used to develop the player does not offer this functionality for now
Change the language of the captions	X			If they are in the multimedia content
Change the language of the audio description	X			If they are in the multimedia content, if not it is the audio of the content itself
Provide accessibility features help	X			
Find captions in the playblack		X		The library used to develop the player does not offer this functionality for now

After testing the graphical editor, all the unit tests have been satisfactory. Every object in the Media player fulfils the functionality established through its requirement. For example, every button accomplishes its specific action, the progress bar shows the progress of the video content, the slider of the volume also fulfils its functionality and the elements of the different menus allow carrying out their functions.

There are requirements which are not fulfilled for now, because the library does not include them. Although there are manners to include them, the cost of it is huge.

5.5.2. Accessibility evaluation

In this case, the accessibility of the graphical editor is going to be tested. This evaluation is accomplished by experts in the field of accessibility following accessibility standards.

This assessment is focused on the ATAG standard. Therefore, both the graphical editor and the content which is provided by this editor are going to be assessed. Table 14 shows a checklist of the ATAG 2.0 considering the success criteria up to AA level and which are related to the graphical editor and the content which are studied in this Doctoral Thesis.

Table 14 ATAG 2.0 checklist

Part A: Make the authoring tool user interface accessible					
Principle A.1: Authoring tool user interfaces follow applicable accessibility guidelines					
Guidelines	**Success criterion**	**Fulfilment**			**Observations**
		Yes	No	N/A	
A.1.2 Ensure that non-web-based functionality is accessible.	A.1.2.1 Accessibility Guidelines.	X			
	A.1.2.2 Platform Accessibility Services.	X			
Principle A.2: Editing-views are perceivable					
A.2.1 Make alternative content available to authors.	A.2.1.1 Text Alternatives for Rendered Non-Text Content.	X			
	A.2.1.2 Alternatives for Rendered Time-Based Media.	X			They do not appear in a direct way

Principle A.3: Editing-views are operable					
Guidelines	**Success criterion**	**Fulfilment**			**Observations**
		Yes	No	N/A	
A.3.1 Provide keyboard access to authoring features.	A.3.1.1 Keyboard Access (Minimum).	X			To all controls and menus
	A.3.1.2 No Keyboard Traps.	X			
	A.3.1.3 Efficient Keyboard Access.	X			
A.3.3 Help authors avoid flashing that could cause seizures.	A.3.3.1 Static View Option.	X			In this case, the video content starts (auto-play), but it could be changed
A.3.4 Enhance navigation and editing via content structure.	A.3.4.1 Navigate By Structure.			X	
A.3.5 Provide text search of the content.	A.3.5.1 Text Search.		X		
A.3.6 Manage preference settings.	A.3.6.2 Save Settings.		X		
Principle A.4: Editing-views are understandable					
A.4.2 Document the user interface, including all accessibility features.	A.4.2.1 Describe Accessibility Features.	X			Example: Shortcuts
	A.4.2.2 Document All Features.		X		There is not explanatory documentation

Part B: Support the production of accessible content					
Principle B.2: Authors are supported in producing accessible content					
B.2.3 Assist authors with managing alternative content for non-text content.	B.2.3.1 Alternative Content is Editable (WCAG).		X		Content cannot be added and neither modified for now
B.2.5 Assist authors with accessible pre-authored content.	B.2.5.1 Accessible Pre-Authored Content Options.			X	
	B.2.5.2 Identify Pre-Authored Content Accessibility.			X	
Principle B.4: Authoring tools promote and integrate their accessibility features					
B.4.1 Ensure the availability of features that support the production of accessible content.	B.4.1.1 Features Active by Default.	X			Although in this case, it is in off for default
	B.4.1.2 Option to Reactivate Features.	X			
	B.4.1.3 Feature Deactivation Warning.		X		
	B.4.1.4 Feature Prominence.	X			
B.4.2 Ensure that documentation promotes the production of accessible content.	B.4.2.1 Model Practice (WCAG).		X		
	B.4.2.2 Feature Instructions.	X			

As it can be seen after the assessment, it can conclude that although both the graphical editor and the content fulfil the majority of the guidelines, there are guidelines that are still not fulfilled. Some of them are due to the fact that the library used to develop the design primitives of the graphical editor does not provide these functionalities in a direct way for now. The other guidelines are owing to the fact that this is the first version of the prototype and not all the criteria, especially the documentation, have been generated yet.

5.6. Conclusions

In this chapter, an authoring tool support has been presented in order to support the design of an accessible media player.

The development of this tool has been divided into three phases: analysis, design and deployment.

Regarding the analysis phase, accessibility standards and technologies to develop the authoring tool have been studied. After that, an elicitation of functional requirements to fulfil the functions offered by the graphical editor has been carried out.

The second phase has been the design phase, in this case, the different parts of this WYSIWYG editor have been explained.

And last but not least, the last phase has been the deployment phase in which the final executable version of the graphical editor has been presented.

Apart from these parts, an evaluation phase has also been accomplished. In this case, two evaluations (a functional evaluation and an accessibility evaluation) have been carried out. The conclusion obtained is that although the graphical editor fulfils some accessibility criteria, there are others that still has to be met.

In addition to these two evaluations, another assessment regarding if the editor gives author support and if the author is able to use the editor is going to be explained and accomplished in Chapter 6.

Chapter 6. Validation

6.1. Introduction

In this chapter, the three main contributions of this Doctoral Thesis are going to be validated through a set of hypothesis also established in this chapter.

The contributions obtained of the PhD proposal are:

Cont 1. A set of accessibility requirements obtained from an exhaustive analysis of standards, best practices and related work. These accessibility requirements should be included in the design of an accessible media player. This set has been presented in Chapter 3 section 3.4.

Cont 2. A workspace which provides a conceptual design proposal following a methodological approach (see Chapter 4). Designers with expertise in modeling can follow a MDD approach (see Annex II) or a MBD approach (see Annex III) to develop an accessible media player. The MBD modeling approach includes the possibility of adapting the final user interface and both approaches are accomplished using a User Interface Description Language (UIDL) (see Figure 22).

Cont 3. As a result of the two previous contributions, a support tool in order to facilitate the design of an accessible media player is also provided. This tool also offers a contextual help so as to guide authors (in this case, designers without any expertise in modeling neither in accessibility) through the accessible design process (see Figure 44). Besides, it is important to consider that this tool is based on the modeling proposal explained in Chapter 4.

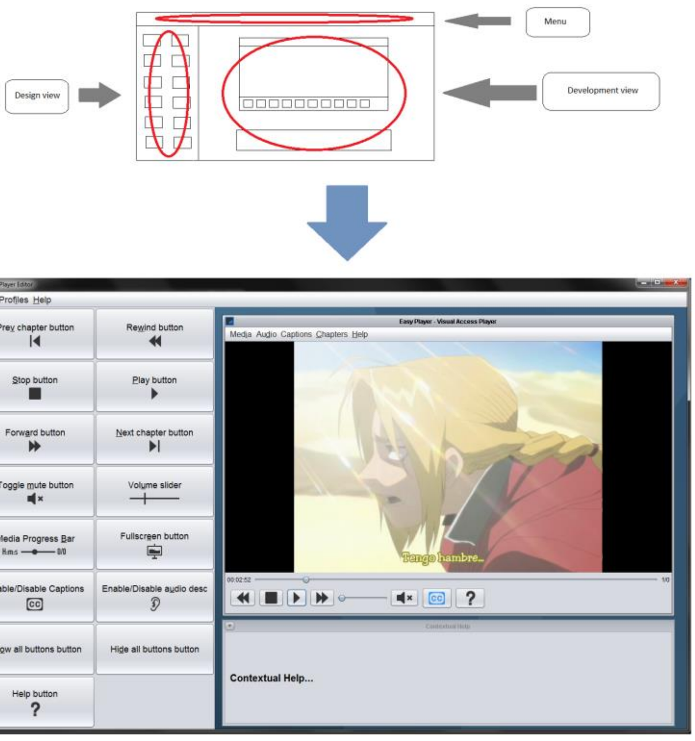


Figure 44 Prototype and screenshot of the graphical editor"s interface

It is important to emphasize that the main focus of this Doctoral Thesis is the abstraction of the requirements while the workspace is the support that provides an strategy of how these requirements can be used.

6.2. Research hypothesis

In order to validate these contributions, next points will be validated:

H1. A set of accessibility requirements has been obtained starting from accessibility standards and oriented to the design of an accessible user agent which delivers multimedia content. These accessibility requirements are sustainable at different levels of abstraction.

The different abstract levels of Cameleon Reference Framework used in order to check if the requirements have been satisfied in every abstraction level.

H2. A design solution based on models which follow MDD or MBD approaches allows carrying out the design of an accessible media player.

The design of an accessible media player has to follow the models established in Annex II (if it is wanted to follow a MDD approach) or Annex III (if it is wanted to follow a MBD approach).

H3. A final design solution through a graphical editor allows accomplishing the design of an accessible media player.

The designer can interact with the graphical editor provided as a support to accomplish the accessible design of a media player. Besides, this editor also provides a contextual help to give designer support to design a accessible media player.

Next sections are going to explain the different kind of validations which are going to be accomplished in order to validate the previous hypothesis.

6.2.1. Hypothesis 1

The validation will check if the requirements and the semantic established in Chapter 3 will be satisfied regarding elements and relationships of every abstraction level which are explained in Chapter 4. This validation certifies H1.

After establishing the requirements, this validation has been accomplished following the four levels architecture of Cameleon Reference Framework of each one of the UIDLs (UsiXML and MARIA) in order to check if at the last level of it, that is, the final user interface, the elements obtained are the same which have been established in this PhD proposal.

Checking a few requirements, it is possible to extrapolate the results to the others due to the fact that the process to obtain the outcome is the same in every case. As an example, in [99], these requirements are modelled through the two first abstraction levels.

Other outcomes which confirm this hypothesis are the positive results of the user agent accessibility assessment obtained when the graphical editor is used (Cont 3, see 6.2.3).

6.2.2. Hypothesis 2

The validation will check if the design solution through models proposed in Chapter 4 could be used by designers with expertise in modeling to design an accessible user agent, in this case, an accessible media player. This validation certifies H2.

This proposal has been accomplished following the Cameleon Reference Framework through a user interface description language. In order to fulfil the proposal, a set of models are provided in Annex II and Annex III depending on if the designer wants to follow a model driven development (see Annex II) or a model-based design (see Annex III). These models are instances of the metamodels of UsiXML or MARIA, depending on the user interface description language used. Therefore, on the one hand, these are well-formed models. And, on the other hand, the models fulfil their objective.

This fulfilment is accomplished through a lab demo which justifies that the approaches could be used in practice and shows both the practical applicability of the approaches, as well as the usefulness of the integration of accessibility requirements in the software development process [153, 154].

In this case, two lab demos have been carried out. The first lab demo is carried out in collaboration with designers with expertise in modeling, thanks to it, some positive outcomes were obtained after developing a model-driven graphical editor [133]. These results demonstrate the suitability of the proposal. Apart from this, the second lab demo is the model-based graphical editor itself which is going to be validated through the following hypothesis (H3).

Moreover, this design solution will be integrated in every interface of a research project[20] in which a media player will be required. In this project, a methodological framework for the development of user-tailored personalised eGovernment services supporting multidevice and taking into account current MDD and MBD technologies is being defined.

6.2.3. Hypothesis 3

The final objective of this proposal is to be used by everybody who needs or wants to design an accessible media player. Therefore, this empirical validation accomplished with the graphical editor, which validates the H3, will be oriented to every professional independently of her/his knowledge about accessibility or modeling.

In order to demonstrate this hypothesis, a exploratory study has been made. This study is going to be described starting from the following parts: participants, stimuli, procedure and method, results obtained and a discussion of the data.

[20] eGovernAbility Project (TIN2014-52665-C2-2-R). A framework for building user tailored accessible services for eAdministration

6.2.3.1.Participants

Fourteen participants accomplished this exploratory study. As a part of the survey which is going to be explained in section 6.2.3.3.1, some personal data of the participants have been gathered. According to these data, the following results have been obtained:

- The age of the participants is between 25 and 40 years.
- Among them, 7 out of the participants have no experience in the use of a graphical editor, the rest of them use or have used it frequently.
- Regarding knowledge about accessibility or WAI standards, two participants have advanced knowledge, another has intermediate knowledge and two others have basic knowledge. The rest of them do not have any knowledge about accessibility or WAI standards.
- In relation with knowledge about modeling or design methodologies based on models or model driven development, one participant has advanced knowledge, another has intermediate knowledge and two others basic knowledge. Likewise as before, the rest of the participants do not have any knowledge about it.
- All the participants have used a media player at least once a week, although the majority of them use the player every day or almost every day. Moreover, all of them access the same content, in this case, audio and video content.

The experimental sessions were carried out in different settings. The majority of the experimental sessions were conducted at participants" office, while the rest of them were made in a Lab of the Computer Science Department of the Universidad Carlos III de Madrid.

The platform used to accomplish the study was Windows 7 in all the sessions.

As far as the personal data gathered are concerned, it can be seen that the participants have different characteristics. For example, there are experts in accessibility, others in modeling and others in the use of a graphical editor. Therefore, this diversity is helpful for the study.

6.2.3.2.Stimuli

The graphical editor presented in Chapter 5 is the stimuli of this exploratory study. This editor is used in order to accomplish the laboratory test (see Figure 45).

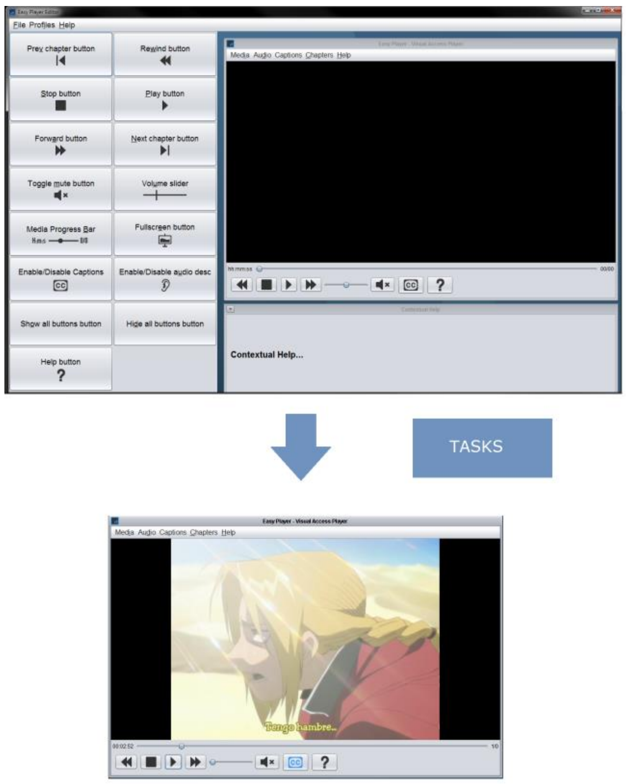


Figure 45 Editor provided and outcome expected

This editor is provided due to the fact that the experiment is going to be accomplished by different users from whom nothing is known about their knowledge or experience in accessibility or modeling.

The interface of the graphical editor has to be user-friendly and intuitive. The reason of that is to help professionals (both experts and non-experts in accessibility) and people without expertise in modeling to design user interfaces, in this case, the interfaces of a user agent which provides video content, in an accessible way. It is difficult to define a mechanism which allows establishing parameters of an experimentation led to measure objects as abstract as the ones which have been defined. This is a generalized problem in Software Engineering [155, 156], when trying to establish the goodness of the methodologies, procedures and tools. There are different techniques of evaluation which allow distinguishing key aspects of products, resources and methodologies in order to select those which the best meet the requirements of efficiency, performance and quality. Every method will be suitable for a particular situation, depending on data which are available and the purpose of the evaluation.

6.2.3.3. Procedure

First, participants were informed about some aspects regarding how to design a media player through the graphical editor and briefly about the three parts which compose the editor. Besides, they signed a consent form.

Second, participants were asked to accomplish four tasks:

1. Task 1: Select the Default Player profile. Use the editor during three minutes.
2. Task 2: Design a player taking into account that the final user to whom the player was directed was visually impaired.
3. Task 3: Design a player to support the largest possible number of users (with or without disability).
4. Task 4: Design a player to be used by elderly who due to their cognitive difficulties should only contain the button that activates the playback and owing to their auditory impairment, the caption button.

The last three tasks were to design three different media players oriented to a specific group of users. Therefore, the goal of every final player was to deliver video content through different alternatives in order to allow a wider range of users to access it.

The participant took the role of designer and was in charge of the media player design considering the necessary requirements to facilitate the final user access. In consequence, the thing which was asked was to design a media player based on models which allows delivering video content and providing access through different types of alternative content taking into account the characteristics of the final user. Moreover, the design had to follow accessibility standards.

Third, a survey method was used to gather information about the user experience, the accomplishment of the tasks and the user itself. The reason of selecting a survey was that this method adjusts perfectly due to the fact that its objective is to register how participants of a project react to new technology, methodology or process which have been used during the development [155]. The survey was elaborated following guides [157, 158]. Starting from the data of the surveys, results based on statistical inference have been obtained.

6.2.3.3.1. Survey

This survey had to be planned and run through a well-defined process in order to take advantages of the outcomes which were going to be obtained. Therefore, it had to accomplish the following activities [157]:

1. Setting specific, measurable objectives.
2. Planning and scheduling the survey.
3. Ensuring that appropriate resources were available.
4. Designing the survey.
5. Preparing the data collection instrument.
6. Validating the instrument.
7. Selecting participants.
8. Administering and scoring the instrument.
9. Analysing the data.
10. Reporting the results.

Taking into account the previous set of activities, the experimentation had the following set of features:

- The main goal of this survey was to check if the third contribution (H3) of this Doctoral Thesis, that is, the graphical editor is known to use. Thanks to this survey, the proposal"s author can know how the respondents accomplished a concrete task and the time which is taken on it.
- The survey was a semi-supervised type. In this case, the survey taker introduced the experiment and answered all the questions which the respondents had.
- It was made an experimental design in order to plan a study to meet the main objective. This objective was obtained in a controlled environment.
- The survey contains three questionnaires:
  - The first one is about user experience. In this questionnaire the survey respondent selected a value from 1 to 7 about 7 qualities (see Annex IV).
  - The second one is about the accomplishment of the tasks. In this questionnaire, participants were interviewed and recorded with a tape recorder. This questionnaire is composed of 11 closed questions, each one with its justification, being one of these questions led only to technical users. Annex IV shows the survey.
  - The last questionnaire is regarding the personal data of the survey respondents. This questionnaire is composed of 8 questions, two of them are led only to people with disabilities.
- The validation of the survey known as content validity was accomplished by experts in modeling and accessibility in order to check if the survey fulfilled the established goals. Besides, the experts considered different parameters about the questions such as the fulfilment of the task or the time which is taken on it.

- The selection of the sample was made among personal contacts. Typically they include between 10 and 20 people who claim to represent a population. This present case was carried out with a group of 14 people, in which 8 out of them were technical users.
- Due to the sample size, it is clear that the results of the evaluation are not statistically conclusive, but, although this assessment is not one the the final objectives of this PhD Thesis, it has allowed to set a starting point for future assessments.

6.2.3.4. Results

In this section a qualitative and a quantitative analysis are going to be presented.

Regarding the qualitative analysis, although it was a semi-supervised survey, the survey participants did not have many doubts. Besides, they were interested in the accomplishment of the tasks, how the survey taker could verify during the reading of the use case and in the accomplishment of Task 1.

In order to gather qualitative data, it is used the justification which appears together with the closed questions. The questions were the followings:

1. Have you been able to accomplish the task completely? If not, give a reason.
2. Have you used some of the established profiles of the editor to accomplish the task? Justify the answer.
3. Do you think the use of the editor is simple? Justify the answer.
4. Do you think the contextual help is useful? Justify the answer.
5. Do you think/Are you sure that the final media player created responds to the final user necessities which have been asked? Justify the answer.
6. Would you recommend the use of this graphical editor? Justify the answer.
7. What do you think about a tool which is able to generate players with accessibility requirements in this platform? Justify the answer.

Questions 1 and 2 are repeated for each of the three tasks (Task 2, 3 and 4) and question 7 is only asked to technical users.

Everybody completed all the tasks, but not all the participants used profiles (except the Default Profile). It is curious the fact that although there are established profiles, they were not used. The reasons of that were different. There were people who do not consider them, but there were survey respondents who knowing their existence preferred to design the media player starting from the default profile.

The majority of the survey respondents did not use the contextual help, the justifications were different, but two of them agreed on the fact that this help is useful for non-technical users. All of them thought that they fulfilled the task in a correct way, that is, responding to the needs of the final user of the media player.

Among the answers obtained, in a nutshell, all the survey respondents agreed on fact that the graphical editor is easy to use, intuitive and a quick way to create a media player, therefore, all of them recommended its use.

Regarding the last question (question 7), in general all the technical respondents agreed on the usefulness of the editor. One of them even said that this editor adds value to overcome accessibility barriers. Another participant said that the editor is innovative and nowadays there is nothing like it on the market. In addition, other respondent thought that it is useful to be able to select what you need.

According to the quantitative analysis, some descriptive statistical results are going to be shown. These results are related to their user experience and the accomplishment of the tasks (the results related to personal data have been explained in section 6.2.3.1).

Except for the Task 1 which has an established time, the time used to accomplish the rest of the tasks depended on the user who carried out them. In order to know the time it took the participant in each task, different statistics have been obtained:

- Figure 46 shows the cumulative time which it took each participant to perform the tasks 2, 3 and 4.

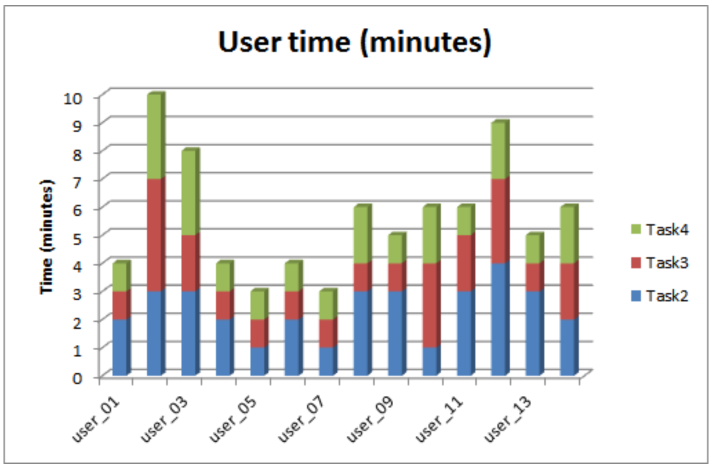


Figure 46 Time (minutes) it takes by each participant in tasks 2, 3 and 4

- Figure 47 presents the percentage of success in the accomplishment of every task considering all the users.

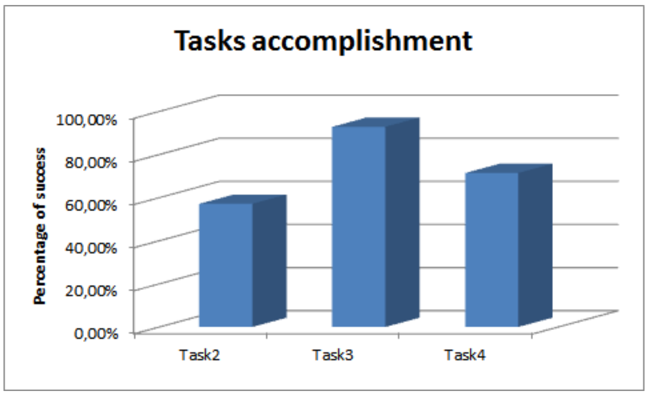


Figure 47 Percentages of success in the accomplishment of the tasks

- And last, Figure 48 compares the time taken by the participant in the accomplishment of every task with the average time taken by all the users in the fulfilment of the corresponding task.

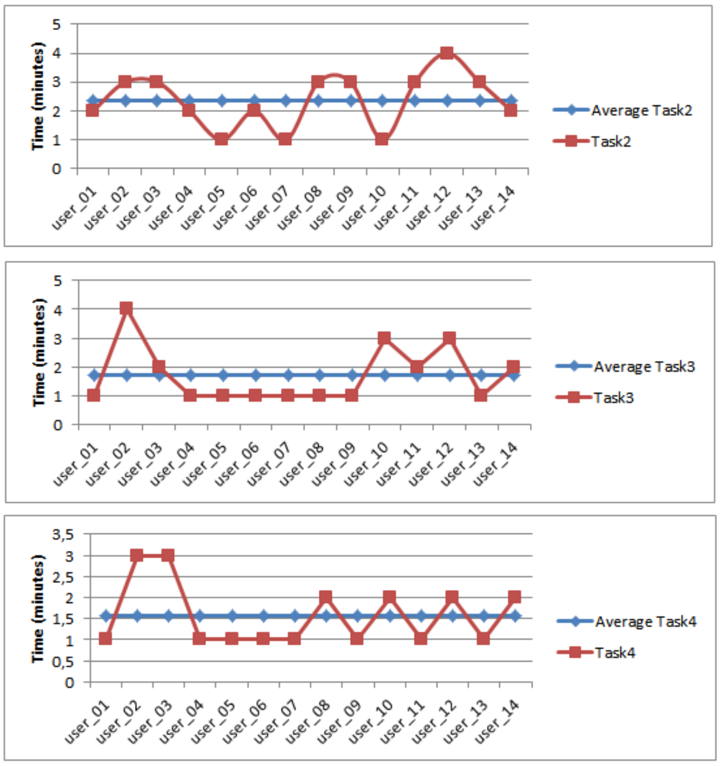


Figure 48 Average time (minutes) by task vs time (minutes) it takes a user in every task

Apart from the time which is taken to fulfil the tasks, the user experience of the participant regarding the graphical editor is also measured. According to Hassenzahl"s model of user experience [159], it has been used pragmatics attributes (relate to the practical usage and functions of the product) and hedonic attributes relate to the user"s psychological well-being to measure the user experience. In order to do that, a Likert scale of 1-7 is used. In this case, seven qualities are measured:

- First quality: Technical or human (value 1 corresponds to technical, value 7 to human).
- Second quality: Complex or simple (value 1 corresponds to complex, value 7 to simple).
- Third quality: Impractical or practical (value 1 corresponds to impractical, value 7 to practical).
- Forth quality: Tricky or direct (value 1 corresponds to tricky, value 7 to direct).
- Fifth quality: Unpredictable or predictable (value 1 corresponds to unpredictable, value 7 to predictable).
- Sixth quality: Confusing or clear (value 1 corresponds to confusing, value 7 to clear).
- Seven quality: Difficult to control or manageable (value 1 corresponds to difficult to control, value 7 to manageable)

Figure 49 shows the results obtained of the analysis of user experience.

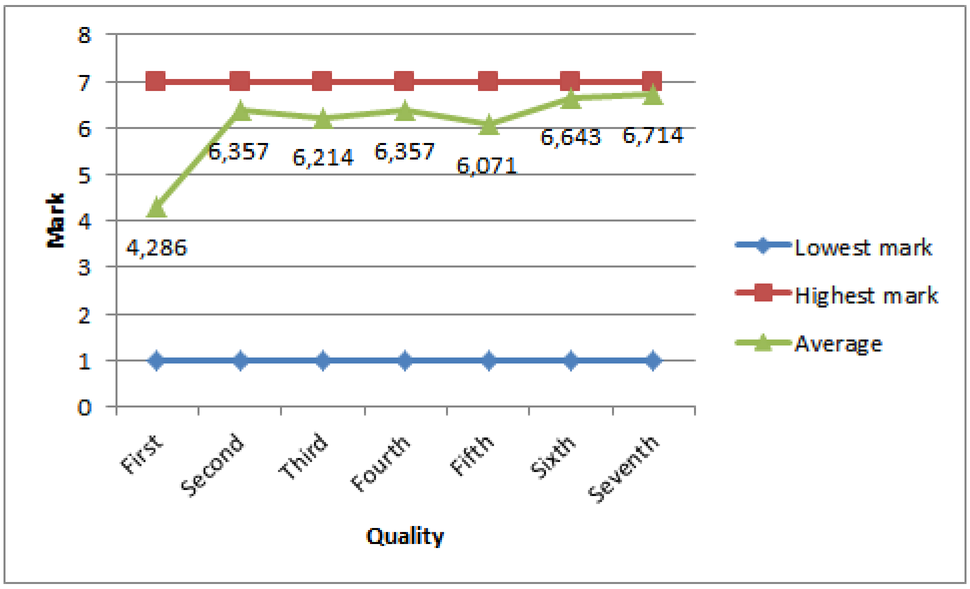


Figure 49 Results of the user experience questionnaire

Thanks to the Likert scale presented in section 6.2.3.4, the emotions of the participant regarding the graphical editor have been assessed [159]. The results obtained of this assessment are the following (see Figure 49), on average, the graphical editor is a slightly more human than technical, more simple than complex, more practical than impractical, more direct than tricky, more predictable than unpredictable, more clear than confusing and more manageable than difficult to control.

6.2.3.5. Discussion

After carrying out the analysis of the results, some conclusions have been obtained.

First of all, concerning the qualitative analysis, it is surprising that the majority of the participants do not use the profiles. Although people read with interest the explanation, after doing it, a great amount of participants developed the media player starting from the Default Profile. One reason would be that some participants have a basic level of English (in spite of the fact that the explanation was in Spanish). Other reason would be that the participants wanted to examine the scope of the tool.

Only one participant considered the contextual help. In this case, one of the reasons would be the same, the language. Other reason was that due to the fact that they did not need it, they did not use it.

The summary of the qualitative analysis is that all the participants consider that the editor is very useful, and they would recommend it because it is simple, intuitive and easy to use. In addition to that, technical participants think that the integration of accessibility requirements in the tool is beneficial as well as a wise decision. Besides, one participant said that the editor is innovative and nowadays there is nothing like it on the market.

Focused on the quantitative analysis, the accomplishment of the design tasks took as much ten minutes (see Figure 46) being the average time taking into consideration the three last tasks almost six minutes. This result allows claiming that the use of the editor decreases the design time considerably because people with low or any expertise in modeling or design can design an accessible media player in few minutes.

Although all the participant thought that they had carried out the different tasks in a correct way, Figure 47 shows the percentages of success in the design process. As it can be seen, more than half of the participants performed successfully Task 2, only one participant failed in the accomplishment of Task 3 and more than two thirds of the participants carried out Task 4 correctly.

Regarding the results obtained from the Likert scale, the participants thought that the graphical editor is more or less in the middle between technical and human, while the rest of qualities studied are clearly classified. Therefore, the outcomes regarding the emotions of the participants agreed with the questionnaire about the accomplishment of the tasks in the fact that the graphical editor is simple, practical, clear and manageable. Besides, they also thought that the graphical editor is direct and predictable.

6.3. Conclusions

After finishing this chapter, it can be concluded that the three hypothesis presented in this chapter have been validated.

As far as H1 is concerned, the requirements and their relationships have been modelled through the four levels of abstraction which form the Cameleon Reference Framework. This hypothesis has been validated checking that the requirements and their relationships are satisfied through the different levels of Cameleon Reference Framework and when the graphical editor is used to design an accessible media player.

According to H2, the models of the two approaches have been presented in order to allow designers with expertise in modeling to design an accessible media player. As it can be seen, this hypothesis has been validated through different lab demos as well as through the integration of this proposal in a research project.

And last, regarding H3, the graphical editor has been presented as a tool to help designers without expertise in modeling neither in accessibility to design an accessible media player. This hypothesis has been validated through the exploratory study in which it has been able to check that a designer without any expertise in modeling or in accessibility can design an accessible media player.

Chapter 7. Conclusions

7.1. Conclusions

Due to the great amount of multimedia content that is delivered on the Web, it is crucial that both the content and the software that provides it are accessible, in this case, for example, a media player should provide synchronized alternatives for content such a captions for deaf people or audio description for blind people. This type of alternatives avoid the exclusion of people with disabilities and elderly people and help them to access all the content delivered.

The integration of accessibility requirements in the user agent is difficult due to its characteristics and the accessibility of the content which the user agent delivers. Owing to this fact, the motivation of this Doctoral Thesis arises to provide technology which can facilitate this integration. This technology is oriented to be used by designers without being experts in accessibility. Besides, this proposal should follow methodological approaches of Software Engineering discipline. Therefore, the objectives of this Doctoral Thesis have been tackled:

- Obj 1. Perform a literature review from which the accessibility requirements in the media player are obtained. This review includes an exhaustive analysis of standards, best practices and related work.
- Obj 2. Create a methodological design approach which provides a workspace for designers to assist in the design of an accessible user agent that provides accessible video content.

And therefore, the following contributions have been obtained:

- Cont 1. A set of accessibility requirements to be included in the design of an accessible media player.
- Cont 2. A modeling design solution which is a conceptual proposal that can follow two approaches, a MDD approach and a MBD approach. Thanks to the models established in this design solution, it is possible to design an accessible media player in different platforms. Besides, this conceptual proposal is oriented to experts in modeling but not in accessibility.
- Cont 3. A concrete and physical design solution of the conceptual proposal. In this case, a graphical editor is used as support in order to design a final accessible media player. This design solution is oriented to designers who are not experts in modeling neither in accessibility.

In order to fulfil the objectives and therefore, obtain the contributions of this Doctoral Thesis, the following subjects have been approached.

After accomplishing the literature review according to accessibility, user agents, media players, user interface design and related works and the accessibility requirements analysis some conclusions have been obtained. First of all, it has not found an analysis of requirements according to the subject treated. Secondly, due to the fact that this proposal aims to be extended to as many people as possible with independence of their knowledge about modeling or accessibility, this proposal provides different manners of design an accessible media player, one directed to experts in modeling and the other one directed to people without expertise in modeling neither in accessibility.

This Doctoral Thesis provides a methodological approach (Chapter 4) which has established the bases of the design model for experts in modeling. In this case, two approaches have been described. It has been used a first architecture (see section 4.3) to guide designers in the design of accessible media players. Once this first aim was achieved and in order to provide adaptive interfaces to the needs and access preferences of users, a second architecture which includes adaptation rules in the final user interface (see section 4.4) has been utilized.

In order to accomplish both proposals, a UIDL has been used. Regarding this language, it is important to highlight that the selection of the UIDL is due to the flexibility and independence that UsiXML and MARIA provide. Besides, these languages do not limit the design in only one platform; they allow users to develop their final interfaces in several platforms in order to obtain an adaptive design.

Apart from these approaches, an authoring tool support (Chapter 5) has also been provided. This is due to the fact that this proposal wants to be extended to people who are not experts in modeling or even in accessibility.

After accomplishing both manners of designing a media player, different assesstments have been fulfilled (Chapter 5 and Chapter 6).

According to Chapter 5, a functional evaluation whose outcomes have been satisfactory and an evaluation of the accessibility which fulfils many of the success criteria have been carried out.

As far as the assessments accomplished in Chapter 6 are concerned, the following assessments have been completed:

- A checking about if the requirements and the semantic established before starting the modeling process remain up to complete the design. In this case, it can be concluded that this validation fulfils the expectations.

- A validation of the workspace which provides a conceptual design proposal following a methodological approach. This validation has been accomplished through lab demos as well as through the integration of this proposal in a reseach project.
- A validation of the graphical editor. This validation has been accomplished through a exploratory study. As has been seen, the results of this study have been very useful.

Once the contributions were obtained, they were the basis of the validation accomplished in Chapter 6. Therefore, thanks to these contributions, this Doctoral Thesis has provided the following artefacts:

Artefact 1. A set of accessibility requirements obtained from Cont 1.

Artefact 2. Models of UsiXML and MARIA established in Annex II and Annex III respectively to design an accessible media player. These models are oriented to designers with expertise in modeling but not in accessibility.

Artefact 3. A graphical editor to give the designer without expertise in modeling or even in accessibility, support to design an accessible media player.

And finally, starting from the contributions and together with the artefacts obtained, three hypothesis have been validated:

Hypothesis 1. The requirements fulfil the success criteria established in the UAAG 2.0 and ISO. These requirements and the semantic established between them are integrated in a UIDL and are transformed through the different abstraction levels.

Hypothesis 2. The models established in every approach (MDD and MBD) allow fulfilling the design of an accessible media player.

Hypothesis 3. The graphical editor used to accomplish the design of an accessible media player is known to use.

In a nutshell, in order to achieve the aim of this Doctoral Thesis, a set of accessibility requirements regarding a media player, a workspace which provides a set of models to fulfil the design of an accessible media player and a graphical editor which allows extending the scope of this Thesis to designers without any expertise in modeling neither in accessibility have been provided.

7.2. Future Work

As far as future works are concerned, the following works can be highlighted (among others):

- A proposal of how to integrate a User-Centered Design approach which allows complementing this Phd proposal from the point of view of the real final users with or without disabilities.
- Incorporate different platforms (Vocal, Mobile, Multimodal Desktop and Multimodal Mobile) with new interaction modalities, such as tactile modality, within the editor.
- Spread the scope of the graphical editor allowing the generation of final user interfaces through other languages and technologies such as the hypertext markup language.
- Update this proposal according to the evolution of accessibility standards.
- Provide an extension of the Doctoral Thesis proposal with an adaptive approach to offer an adaptive generation of final user interfaces in different interaction modalities.
- Incorporate new accessibility requirements to those included in this version of the graphical editor.

7.3. Results' Dissemination

As research results of this Dotoral Thesis, some works have been published in different journals and other works have been presented in different academic and research forums.

On the other hand, this work has been influenced and has been partially supported by the Regional Government of Madrid under the Research Network MA2VICMR [S2009/TIC-1542], by the Spanish Ministry of Education under the project MULTIMEDICA [TIN2010-20644-C03-01], by the European Commission Seventh Framework Programme under TrendMiner project [FP7- ICT287863] and by the Spanish Ministry of Economy and Competitiveness under eGovernAbility project [TIN2014-52665-C2-2-R].

During the accomplishment of this Doctoral Thesis, a stay in the Human-Computer Interaction Research Group at the University of York was completed under the supervision of Christhoper Power and Helen Petrie. The work accomplished during the stay contributed to know and understand how the type of editors known as WYSIWYG editor operates.

The thesis results can be divided into two different groups which correspond to the two objectives of the Doctoral Thesis: Accessibility requirements for a user agent that provides video content and Methodological approach to design an accessible media player.

Regarding Accessibility requirements for a user agent that provides video content, the works which have been published are:

Conf 1. Moreno, L., González-García, M., Martínez, P., and Iglesias, A. 2011. A study of accessibility requirements for media players on the Web. 14th International Conference on Human-Computer Interaction (HCII 2011), Orlando, Florida, USA, July, 2011, Springer Computer Science Editorial, Volume: LNCS 6765, pp. 249-257.

Conf 2. Moreno, L., Martínez, P., Iglesias, A., and González-García, M. 2011. HTML5 support for an accessible user-video-interaction on the Web. 13th IFIP TC13 Conference on Human-Computer Interaction (INTERACT 2011), Lisboa, Portugal, September, 2011, LNCS, Springer, Volume: 6949/2011, pp. 535-539.

Conf 3. González-García, M., Moreno, L., Martínez, P., and Iglesias, A. 2011. Requisitos de accesibilidad web en los reproductores multimedia. XII Congreso de Interacción Persona-Ordenador (Interacción 2011), Lisboa, Portugal, September, 2011, ISBN: 978-84-9281-2, pp. 43-53.

Conf 4. González-García, M., Moreno, L., Martínez, P., and Iglesias, A. 2011. Web accessibility requirements for media players. 13th IFIP TC13 Conference on Human-Computer Interaction (INTERACT 2011), Lisboa, Portugal, September, 2011, LNCS, Springer, Volume: 6949, pp. 669-674.

As far as Methodological approach to design an accessible media player is concerned, the following works have also been published:

Jour 1. González-García, M., Miñon, R., Moreno, L., Martínez, P., and Abascal, J. 2013. A model-based graphical editor to design accessible media players. *Journal of Universal Computer Science*, Volume 19, Nº 18, pp. 2656-2676.

Jour 2. González-García, M., Moreno, L., and Martínez, P. 2015. Approach design of an accessible media player. *Universal Access in the Information Society*, Springer, Volume 14, Number 1, pp. 45-55. DOI: 10.1007/s10209-013-0342-z.

Conf 5. González-García, M., Moreno, L., and Martínez, P. 2014. Integration of Accessibility Requirements in the Design of Multimedia User Agents Interface. The first ACM womENcourage Conference, Manchester, United Kingdom, January, 2014, Volume: In press.

Conf 6. González-García, M., Moreno, L., and Martínez, P. 2012. An approach to User Interface Design of an accessible user agent. 4th International Conference on Software Development for Enhancing Accessibility and Fighting Info-exclusion (DSAI 2012), Douro Region, Portugal, July, 2012, Elsevier, Volume: 14, pp. 254-262.

Conf 7. González-García, M., Moreno, L., and Martínez, P. 2014. Adaptation rules for Accessible Media Player Interface. XV International Conference on Human Computer Interaction (Interacción 2014), Puerto de la Cruz, Tenerife, Spain, September, 2014, ACM, New York, ISBN: 978-1-4503-2, Article Nº 5.

Conf 8. González-García, M., Moreno, L., and Martínez, P. 2015. A Model-Based Tool to develop an Accessible Media Player. 17th International ACM SIGACCESS Conference on Computers & Accessibility (ASSETS '15), Lisboa, Portugal, October, 2015, Volume: In press.

References

1. CISCO. 2012. Cisco Visual Networking Index: Forecast and Methodology, 2012–2017. http://www.cisco.com/c/en/us/solutions/collateral/service-provider/ip-ngn-ip-next-generation-network/white_paper_c11-481360.html.
2. Digiday. 2013. 15 Stats Brands Should Know About Online Video. http://digiday.com/brands/celtra-15-must-know-stats-for-online-video/.
3. Invodo. 2014. Video Statistics: The Marketer"s Summary 2014. http://www.invodo.com/wp-content/uploads/2014/02/Invodo_Video_Statistics_The_Marketers_Summary_2014.pdf.
4. FreeWheel. 2014. Video Monetization Report Q4 2014. http://www.iab.net/media/file/Q4_2014_FreeWheel_Video_Monetization_Report.pdf.
5. W3C. 2012. Web Content Accessibility Guidelines (WCAG). http://www.w3.org/WAI/intro/wcag.
6. WAI. 2010. Web Accessibility Initiative. http://www.w3.org/WAI/.
7. Moreno, L., Martínez, P. and Ruiz-Mezcua, B. 2008. Disability Standards for Multimedia on the Web. October. IEEE Multimedia, IEEE Computer Society, 2008, ISSN: 1070-986X, Vol: 15, N: 4, pp. 52-54.
8. W3C. 2015. WAI Guidelines and Techniques. http://www.w3.org/WAI/guid-tech.html.
9. W3C. 2005. User Agent Accessibility Guidelines. http://www.w3.org/WAI/intro/uaag.
10. Lazar, J., Dudley-Sponaugle, A. and Greenidge, K. 2004. Improving Web Accessibility: A Study of Webmaster Perceptions. Computers and Human Behavior. Vol. 20 No. 2, pp. 269-288.
11. Moreno, L., Valverde, F., Martínez, P. and Pastor, O. 2013. Supporting accessibility in Web engineering methods: a methodological approach. Journal of Web Engineering, RINTON PRESS, INC, January, 2013, ISSN: 1540-9589, Vol.12, No.3&4.
12. ISO. 2010. International Standards for Business, Government and Society (ISO). Ergonomics of human-system interaction -- Part 210: Human-centred design for interactive systems ISO DIS 9241-210.
13. Koch N. 2006. Transformation Techniques in the Model-Driven Development Process of UWE. Workshop proceedings of the sixth international conference on Web engineering (ICWE"06), Palo Alto, California, July, 2006, Article nº 3.
14. ISO. 2008. International Organization for Standardization, ISO 9241-171:2008, Ergonomics of human-system interaction (Guidance on software accessibility). http://www.iso.org/iso/catalogue_detail.htm?csnumber=39080.
15. W3C. 2012. How People with Disabilities Use the Web: Overview. http://www.w3.org/WAI/intro/people-use-web/Overview.html.
16. UN. 1948. United Nations Universal Declaration of Human Rights. http://watchlist.org/wordpress/wp-content/uploads/Universal-declaration-of-human-rights.pdf.
17. ISO. 1947. International Organization for Standardization. http://www.iso.org/.
18. ISO. 2012. Information technology -- W3C Web Content Accessibility Guidelines (WCAG) 2.0.

http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=58625.

19. W3C. 2008. Web Content Accessibility Guidelines 2.0. http://www.w3.org/TR/WCAG20/.
20. PowerMapper. 2012. Government Accessibility Standards and WCAG 2.0. http://blog.powermapper.com/blog/post/Government-Accessibility-Standards.aspx.
21. Section 508. 1986. http://www.hhs.gov/web/508/index.html.
22. BITV 2.0. 2011. Bundesministerium der Justiz,Barrierefreie-Informationstechnik-Verordnung 2.0. http://www.gesetze-im-internet.de/bitv_2_0/BJNR184300011.html.
23. RGAA. 2009. République Française, Référentiel Général d'Accessibilité pour les Administrations. http://references.modernisation.gouv.fr/introduction-au-rgaa-0.
24. AODA. 2005. Ontario, Accessibility for Ontarians with Disabilities Act. http://www.e-laws.gov.on.ca/html/statutes/english/elaws_statutes_05a11_e.htm.
25. AENOR. 2012. Asociación Española de Normalización y Certificación, UNE 139803:2012, Web content accessibility requirements, Norma UNE 139803:2012. http://www.aenor.es/aenor/normas/normas/fichanorma.asp?tipo=N&codigo=N0049614.
26. W3C. 2014. User Agent Accessibility Guidelines 2.0. http://www.w3.org/TR/UAAG20/.
27. W3C. 2015. Authoring Tool Accessibility Guidelines 2.0. http://www.w3.org/TR/ATAG20/.
28. W3C. 2015. Guidance on Applying WCAG 2.0 to Non-Web Information and Communications Technologies (WCAG2ICT). http://www.w3.org/TR/wcag2ict/.
29. CVAA. 2010. Federal Communications Commission, Twenty-First Century Communications and Video Accessibility Act. http://www.fcc.gov/guides/21st-century-communications-and-video-accessibility-act-2010.
30. ETSI. 2014. EN 301 549 V11.1, Accessibility requirements suitable for public procurement of ICT products and services in Europe. http://www.etsi.org/deliver/etsi_en/301500_301599/301549/01.01.01_60/en_301549v010101p.pdf.
31. Moreno, L., González-García, M., Martínez, P. and Iglesias, A. 2011. A study of accessibility requirements for media players on the Web. 14th International Conference on Human-Computer Interaction (HCII 2011). 6th International Conference on Universal Access in Human-Computer Interaction, Orlando, Florida, USA, July, 2011, Springer Computer Science Editorial, Volumen: LNCS 6765, pp. 249-257.
32. Easy Youtube player. 2009. http://hiantonia.com/journal/2009/06/17/easy-youtube-player-making-it-easier/.
33. eWEEK. 2012. http://www.eweek.com/cloud/youtube-expands-video-captioning-in-6-more-languages/.
34. ACCESS: Youtube. 2013. http://accessyoutube.org.uk/.
35. W3C. 2014. HTML5, A vocabulary and associated APIs for HTML and XHTML.

http://www.w3.org/TR/html5/.

36. Pfeiffer, S. and Green, T. 2015. Using and Manipulating HTML5 Video and Audio Elements. Beginning HTML5 Media. pp. 25-66. http://link.springer.com/chapter/10.1007/978-1-4842-0460-3_2.

37. Pfeiffer, S. and Green, T. 2015, Accessibility, Internationalization, and Navigation. Beginning HTML5 Media. pp 129-189. http://link.springer.com/chapter/10.1007/978-1-4842-0460-3_4.

38. W3C. 2014. Introduction to Model-Based User Interfaces. http://www.w3.org/TR/mbui-intro/.

39. Schlungbaum, E. 1996. Model-based User Interface Software Tools. Current state of declarative models. GVU Technical Report; GIT-GVU-96-30. Georgia Institute of Technology. http://hdl.handle.net/1853/3516.

40. W3C. 2012. Concur Task Trees (CTT). http://www.w3.org/2012/02/ctt/.

41. MacDonald, A., Russel, D. and Atchison, B. 2005. Model-driven Development within a Legacy System: An industry experience report. In Proceedings of the Australian Software Engineering Conference (ASWEC"05).

42. MDA, Model Driven Architecture. http://www.omg.org/mda/.

43. OMG, Object Management Group. http://www.omg.org/.

44. Truyen, F. 2006. WhitePaper: The Fast Guide to Model Driven Architecture. The Basics of Model Driven Architecture. http://www.omg.org/mda/mda_files/Cephas_MDA_Fast_Guide.pdf.

45. Kardoš, M. and Drozdová, M. 2010. Analytical Method of CIM to PIM Transformation in Model Driven Architecture (MDA). Journal of Information & Organizational Sciences, Volume 34, Issue 1, pp. 89-99.

46. Sharifi, H. R., Mohsenzadeh, M. and Hashemi, S. M. 2012. CIM to PIM Transformation: An Analytical Survey. International Journal of Computer Technology & Applications, Vol. 3, Issue 2, pp. 791-793.

47. Merilinna, J. 2005. A Tool for Quality-Driven Architecture Model Transformation. Doctoral Thesis, VTT Electronics Software architecture group, VTT Publications 561, Otamedia Oy, Espoo 2005, Oulu. http://www.vtt.fi/inf/pdf/publications/2005/P561.pdf.

48. Bodart, F., Hennebert, A. M., Leheureux, J. M., Sacre, I. and Vanderdonckt, J. 1993. Architecture Elements for Highly-Interactive Business-Oriented Applications. Lecture Notes in Computer Science, Vol. 153, L. Bass, J. Gornostaev & C. Unger (éds.), Springer-Verlag, Berlin, pp. 83-104.

49. Bodart, F. and Vanderdonckt, J. 1994. On the Problem of Selecting Interaction Objects. Proc. of BCS Conf. HCI'94 "People and Computers IX", Glasgow, 23-26 août 1994, G. Cockton, S.W. Draper & G.R.S. Weir (éds.), Cambridge University Press, Cambridge, pp. 163-178.

50. Bodart, F., Hennebert, A. M., Leheureux, J. M., Provot, I., Sacre, B. and Vanderdonckt, J. 1995. Towards a Systematic Building of Software Architectures: the Trident methodological guide. Proc. of 2nd Eurographics Workshop on Design, Specification, Verification of Interactive Systems DSV-IS'95, Toulouse, 7-9 juin 1995, Ph. Palanque & R. Bastide (éds.), Springer-Verlag, Vienne, pp. 262-278.

51. Limbourg, Q., Vanderdonckt, J. and Souchon N. 2000. The Task-Dialog and Task-Presentation Mapping Problem: Some Preliminary Results. In Proceedings

of the 7th international conference on Design, specification, and verification of interactive systems DSV-IS 2000, pp. 227-246.

52. Bouillon, L., Vanderdonckt, J. and Souchon, N. 2002. Recovering Alternative Presentation Models of a Web Page with VAQUITA. Proceedings of CADUI"2002, pp. 311-322.

53. Nunes, N. and Cunha, J. 2000. Wisdom: a UML based Architecture for Interactive Systems. In Interactive Systems: Design, Specification, and Verification. 7th International Workshop DSV-IS, Limerick, Ireland, June, 2000. Ph. Palanque and F. Paternò (Eds.). LNCS Vol. 1946, Springer, 2000.

54. Nunes, N. 2001. Object Modeling for User-Centered Development and User Interface Design: The Wisdom Approach. Tesis doctoral, Universidad de Madeira, Abril, 2001.

55. Nunes, N. and Cunha, J. 1998. Case Study: SITINA – A Software Engineering Project Using Evolutionary Prototyping. In CaiSE"98/IFIP 9.1 EMMSAD"98 Workshop.

56. Nunes, N. and Cunha, J. 2000. Wisdom: A Software Engineering Method for Small Software Development Companies. Software, IEEE, Volume: 17 , Issue: 5, Sept.-Oct., 2000, pp. 113-119. http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6156721.

57. Lozano, M.D. 2001. Entorno Metodológico Orientado a Objetos para la Especificación y Desarrollo de Interfaces de Usuario. Tesis Doctoral, Universidad Politécnica de Valencia.

58. Lozano, M., González, P., Montero, F., Molina, J. P. and Ramos, I. 2002. Integrating Usability within the User Interface Development Process of Web Applications. In Proc. of 2nd Int. Workshop on Web Oriented Software Technology (IWWOST'02), Málaga, pp. 134-147.

59. Letelier, P., Ramo, I., Sánchez. P. and Pastor, O. 1998. OASIS version 3: A Formal Approach for Object Oriented Conceptual Modeling . UPV, Spain.

60. Calvary, G., Coutaz, J., Bouillon, L., Florins, M., Limbourg, Q., Marucci, L., Paternò, F., Santoro, C., Souchon, N., Thevenin, D. and Vanderdonckt, J. 2002. The CAMELEON Reference Framework, deliverable 1.1. http://giove.isti.cnr.it/projects/cameleon/pdf/CAMELEON%20D1.1RefFramework.pdf.

61. Sukaviriya, P., Foley, J.D. and Griffith, T. 1993. A second generation user interface design environment: the model and the runtime architecture. In Proceeding of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems (CHI '93). pp. 375-382.

62. Szekely, P., Sukaviriya, P. Castells, P., Muthukumarasamy, J. and Salcher. E. 1996. Declarative interface models for user interface construction tools: the MASTERMIND approach. In: L. Bass, C. Unger (Eds.): Engineering for Human-Computer Interaction. Proceedings of the IFIP TC2/WG2.7 working conference on engineering for human-computer interaction, Yellowstone Park, August 1995, London: Chapman & Hall, 1996, pp. 120-150.

63. Lonczewski, F. and Schreiber, S. 1996. Generating User Interfaces with the FUSE-System.

64. UIDL, User Interface Description Languages. http://www.uidl.net.

65. Delgado, A., Estepa, A., Troyano, J.A. and Estepa, R. 2008. Un sistema de desarrollo de interfaces web basado en modelos para aplicaciones de gestión. IX

Congreso Internacional Interacción Persona-Ordenador (Interacción 2008), Albacete, España, Junio, 2008, ISBN: 978-84-691-3871-7, pp. 313-318.

66. UsiXML. 2011. User Interface Extensible Markup Language (UsiXML). http://www.usixml.org/.

67. Stanciulescu, A., Limbourg, Q., Vanderdonckt, J., Michotte, B. and Montero, F. 2005. A transformational approach for multimodal web user interfaces based on UsiXML. In Proceedings of the 7th international conference on Multimodal interfaces (ICMI '05). pp. 259-266.

68. Puerta, A. and Eisenstein, J. 2002. XIML: a common representation for interaction data. In Proceedings of the 7th international conference on Intelligent user interfaces (IUI '02). pp. 214-215.

69. Ali, M.F., Pérez-Quiñones, M.A., Abrams, M. and Shell, E. 2002. Building Multi-Platform User Interfaces with UIML. Computer-Aided Design of User Interfaces III. pp. 255-266.

70. MarcAbrams, M. and Phanouriou, C. 1999. UIML: An XML Language for Building Device-Independent User Interfaces. In Proceedings of XML‘99, Philadelphia. December 1999.

71. Fabio Paternò, F., Santoro, C. and Spano, L.D. 2012. MARIA (Model-based lAnguage foR Interactive Applications). http://www.w3.org/wiki/images/3/36/MARIA.pdf

72. XUL Tutorial. 1999. https://developer.mozilla.org/en-US/docs/XUL/Tutorial.

73. OpenLaszlo. 2006. An Open Architecture Framework for Advanced Ajax Applications. http://www.openlaszlo.org/whitepaper/LaszloWhitePaper.pdf.

74. Crease, M., Gray, P. and Brewster, S. 2001. A toolkit Mechanism and Context Independent Widgets. Interactive Systems Design, Specification, and Verification, Lecture Notes in Computer Science Volume 1946, 2001, pp. 121-133.

75. Berti, S., Correani, F., Paternò, F. and Santoro, C. 2009. The TERESA XML Language for the Description of Interactive Systems at Multiple Abstraction Levels. http://giove.cnuce.cnr.it/projects/cameleon/pdf/workshop-avi.pdf.

76. Liang, Y.D. 2014. Introduction to Java Programming, Prentice Hall, ISBN-13: 978-0-13-359220-7.

77. Python. 2014. https://www.python.org/.

78. Standard ECMA-334. 2006. C# Language Specification.

79. Anuradha, P. 2000. Automated Circulation System using Visual Basic 6.0. Annals of Library Science and Documentation, 47, 1, pp. 23-40.

80. Castro, E. 1998. Perl and CGI for the World Wide Web (Visual QuickStart Guides), Peachpit Press, ISBN-13: 9780201353587.

81. Moreno, L., Gálvez, M.C., Ruiz, B. and Martínez, P. 2008. Inclusion of Accessibility Requirements in the Design of Electronic Guides for Museums. Computers Helping People with Special Needs (ICCHP 2008), 11th International Conference, Linz, Austria, July 09-11, 2008, LNCS, Springer, 5105, pp. 1101-1108. DOI = http://link.springer.com/chapter/10.1007%2F978-3-540-70540-6_165.

82. Brunet, P., Feigenbaum, B.A., Harris, K., Laws, C., Schwerdtfeger, R. and Weiss, L. 2005. Accessibility requirements for systems design to accommodate

users with vision impairments. BM Systems Journal. 44, 3 (2005), pp. 445-466. DOI = http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=5386679.

83. Rosas Villena, J.M., Goularte, R. and Pontin M. Forter, R. 2014. A User Test with Accessible Video Player Looking for User Experience. Universal Access in Human-Computer Interaction. Design for All and Accessibility Practice. 8516, pp. 623-633. http://link.springer.com/chapter/10.1007%2F978-3-319-07509-9_59.

84. Peissner, M., Schuller, A., Ziegler, D., Knecht, C. and Zimmermann, G. 2014. Requirements for the Successful Market Adoption of Adaptive User Interfaces for Accessibility. Universal Access in Human-Computer Interaction. Design for All and Accessibility Practice. 8516, pp. 431-442. http://link.springer.com/chapter/10.1007%2F978-3-319-07509-9_41.

85. Federico, M. and Furini, M. 2012. Enhancing learning accessibility through fully automatic captioning. Proceedings of the 9th International Cross-Disciplinary Conference on Web Accessibility (W4A'12), Lyon, France, ISBN: 978-1-4503-1019-2.

86. Wald, M. 2011. Crowdsourcing Correction of Speech Recognition Captioning Errors. In Proceedings of the 8th International Cross-Disciplinary Conference on Web Accessibility (W4A'11), Hyderabad, Andhra Pradesh, India.

87. Hughes, C. J., Armstrong, M., Jones, R. and Crabb, M. 2015. Responsive design for personalised subtitles. Proceedings of the 12th Web for All Conference (W4A"15), Florence, Italy, Article No. 8. ISBN: 978-1-4503-3342-9. http://dl.acm.org/citation.cfm?id=2745555.2746650.

88. Lim, W., Jang, I. and Ahn, C. 2014. Dynamic Subtitle Authoring Method Based on Audio Analysis for the Hearing Impaired. Computers Helping People with Special Needs. 8547, pp. 53-60. http://link.springer.com/chapter/10.1007%2F978-3-319-08596-8_9.

89. Hong, R., Wang, M., Yuan, X., Xu, M., Jiang, J., Yan, S. and Chua, T. 2011. Video accessibility enhancement for hearing-impaired users. ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM). 7S, 1 (2011), Article No. 24. http://dl.acm.org/citation.cfm?id=2037681.

90. Lasecki, W., Miller, C.D., Kushaklnagar, R. and Bigham, J.P. 2013. Legion Scribe: Real-Time Captioning by Non-Experts. Proceedings of the 10th International Cross-Disclipinary Conference on Web Accessibility (W4A"13), Rio De Janerio, Brazil. Article No. 22. DOI= http://dx.doi.org/10.1145/2461121.2461151.

91. Stinson, M.S., Francis, P., Elliot, L.B, and Easton D. 2014. Real-time caption challenge: C-print. Proceedings of the 16th International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS"14), Rochester, New York, USA, pp. 317-318, ISBN: 978-1-4503-2720-6. http://dl.acm.org/citation.cfm?id=2661337&CFID=683472779&CFTOKEN=43951259.

92. Hansen, E.G., Forer, D.C. and Mang, L.S. 2000. Making Movies On the Web Accessible to People With Disabilities. In Proceedings of International Conference on Mathematics / Science Education and Technology 2000, pp. 197-201.

93. Kobayashi, M., Fukuda, K., Takagi, H. and Asakawa, C. 2009. Providing Synthesized Audio Description for Online Videos. Proceedings of the 11th

international ACM SIGACCESS conference on Computers and accessibility (Assets '09), New York, USA, 2009, ISBN: 978-1-60558-558-1, pp. 249-250.

94. Chapdelaine, C. and Gagnon, L. 2009. Accessible videodescription On-Demand. Proceedings of the 11th international ACM SIGACCESS conference on Computers and accessibility (Assets'09), New York, USA, 2009, ISBN: 978-1-60558-558-1, pp. 221-222.

95. Debevc, M., Kosec, P. and Holzinger, A. 2011. Improving multimodal web accessibility for deaf people: sign language interpreter module. Multimedia Tools and Applications archive. Volume 54 Issue 1, August 2011, pp. 181-199.

96. Saray Villamizar, J.F., Encelle, B., Prié, Y. and Champin, P. 2011. An adaptive videos enrichment system based on decision trees for people with sensory disabilities. Proceedings of the 8th International Cross-Disciplinary Conference on Web Accessibility (W4A'11), Hyderabad, Andhra Pradesh, India, pp. 1-4.

97. Encelle, B., Ollagnier-Beldame, Mm, Pouchot, S. and Prié, Y. 2011. Annotation-based Video Enrichment for Blind People: A Pilot Study on the Use of Earcons and Speech Synthesis, The 13th International ACM SIGACCESS Conference on Computers and Accessibility (Assets'11), Dundee, Scotland, UK, pp. 123-130.

98. González-García, M., Moreno, L., Martínez, P. and Iglesias, A. 2011. Web accessibility requirements for media players. 13th IFIP TC13 Conference on Human-Computer Interaction (INTERACT 2011), Lisboa, Portugal, September, 2011, LNCS, Springer, Volume: 6949, pp. 669-674.

99. González-García, M., Moreno, L., and Martínez, P. 2012. An approach to User Interface Design of an accessible user agent. 4th International Conference on Software Development for Enhancing Accessibility and Fighting Info-exclusion (DSAI 2012), Douro Region, Portugal, July, Elsevier, Volume: 14, pp. 254-262.

100. Nishimura, K. and Cohen, M. 2012. Media Players for Accessibility. Proceedings of the 2012 Joint International Conference on Human-Centered Computer Environments (HCCE"12), New York, USA, 2012, ISBN: 978-1-4503-1191-5, pp. 184-189.

101. Niederl, F., Bußwald, P., Tschare, G., Hackl, J. and Philipp, J. 2012. Dubbing of Videos for Deaf People – A sign Language Approach, Proceedings of the 13th International Conference on Computers Helping People with Special Needs (ICCHP 2012), Linz, Austria, pp. 225-228.

102. Mourouzis, A., Partarakis, N., Doulgeraki, C., Galanakis, C. and Stephanidis, C. 2008. An Accessible Media Player as a User Agent for the Web. Proceedings of the 11th international conference on Computers Helping People with Special Needs (ICCHP '08), Linz, Austria, pp. 474-481.

103. Rosas Villena, J.M., Costa Ramos, B., Pontin M. Forter, R. and Goularte, R. Web Videos – Concerns About Accessibility based on User Centered Design. Proceedings of the 5th International Conference on Software Development and Technologies for Enhancing Accessibility and Fighting Info-exclusion (DSAI 2013), Procedia Computer Science, 27 (2014), pp. 481-490.

104. Hanson, L. and Crayne, S. 2005. Personalization of Web browsing: adaptations to meet the needs of older adults. Universal Access in the Information Society. Volume 4 Issue 1, September 2005, pp. 46–58.

105. Stanciulescu, A. Limbourg, Q, Vanderdonckt, J, Michotte, B. and Montero, F. 2005. A transformational approach for multimodal web user interfaces based on UsiXML In Proceedings of the 7th international conference on Multimodal

interfaces (ICMI '05), Trento, Italy, October 04-06, 2005, ACM, New York, NY, pp. 259-266. DOI= http://dl.acm.org/citation.cfm?doid=1088463.1088508.

106. Stanciulescu, A. 2008. A Methodology for Developing Multimodal User Interfaces of Information Systems. Doctoral Thesis. SIMILAR PhD Registration Number: ISBN 978-2-87463-114-6. Copyright registration D/2008/9964/8. EAN 9782874631146. Université catholique de Louvain.

107. Link, S., Schuster, T. Hoyer, P. and Abeck, S. 2008. Focusing Graphical User Interfaces in Model-Driven Software Development. Advances in Computer-Human Interaction The First International Conference on Advances in Computer-Human Interaction, Sainte Luce, Martinique, February 10-15, 2008, ACHI2008. pp. 3-8. DOI= http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=4455950.

108. Lu, X. and Wan, J. 2007. Model Driven Development of Complex User Interface. In MDDAUI.

109. Huang, J., Voeten, J., Groothuis, M., Broenink, J. and Corporaal, H. 2007. A model-driven design approach for mechatronic systems. En Seventh International Conference on Application of Concurrency to System Design (ACSD 2007), pp. 127-136.

110. Santoro, C. 2004. A Task Model-Based Approach for the Design and Evaluation of Innovative User Interfaces. Presses univ. de Louvain.

111. Chesta, C., Paternò, F. and Santoro, C. 2004. Methods and Tools for Designing and Developing Usable Multi-Platform Interactive Applications. PsychNology Journal, 2, 1 (2004), pp. 123-139.

112. Melchior, J., Vanderdonckt, J. and Van Roy, P. 2011. A model-based approach for distributed user interfaces. In Proceedings of the 3rd ACM SIGCHI symposium on Engineering interactive computing systems. pp. 11-20.

113. Rodríguez, M. L., Garrido, J. L., Hurtado, M. V. and Noguera, M. 2007. An approach to the model-based design of groupware multi-user interfaces. In Groupware: Design, Implementation, and Use. pp. 157-164.

114. Puerta, A. R. 1997. A model-based interface development environment. Software, IEEE, Volume 14, Nº 4, pp. 40-47.

115. Saleh, E., Khazem, R. and Kamel, A. 2007. Model-Based Approaches for Multi-Device User Interface Design and Development. The 3rd International Conference on Information Technology (ICIT"2007), Amman, Jordan, pp. 211-221. http://www.zuj.edu.jo/conferences/ICIT07/PaperList/Papers/411RafaEman.pdf.

116. Meixner, G., Paternò, F. and Vanderdonckt, J. 2011. Past, Present, and Future of Model-Based User Interface Development. In i-com, 10, Nº. 3, pp. 2-11.

117. Hanumansetty, R. G. 2004. Model based approach for context aware and adaptive user interface generation. Doctoral Thesis, Virginia Polytechnic Institute and State University.

118. Miñón, R., Paternò, F. and Arrue, M. 2013. An environment for designing and sharing adaptation rules for accessible applications. In Proceedings of the 5th ACM SIGCHI symposium on Engineering interactive computing systems (EICS '13), London, United Kingdom, June 24-27, 2013, ACM, New York, NY, pp. 43-48. DOI= http://dl.acm.org/citation.cfm?doid=2494603.2480323.

119. Peissner, M., Häbe, D., Janssen, D. and Sellner, T. 2012. MyUI: generating accessible user interfaces from multimodal design patterns. In Proceedings of the 4th ACM SIGCHI symposium on Engineering interactive computing systems (EICS'12), Copenhagen, Denmark, June 25–28, ACM, New York, NY, pp. 81-90. DOI = http://dl.acm.org/citation.cfm?doid=2305484.2305500.

120. Ranaweera, G.D.S. and Withanage, D.K. 2013. Adaptive user interface concept for personal desktop. International Journal of Information Technology and Business Management, 17, 1 (September. 2013), pp. 43-48. http://www.jitbm.com/17th%20Volume/4%20Information%20Technology.pdf.

121. Giani, G., Paternò, F., Santoro, C. and Spano, L.D. 2012. A Set of Languages for Context-Aware Adaptation. In Proceedings of the Workshop on Context-Aware Adaptation of Service Front-Ends (CASFE 2012), Pisa, Italy, November. http://ceur-ws.org/Vol-970/paper10.pdf.

122. Serenoa Project. 2013. http://www.serenoa-fp7.eu/.

123. Mori, G., Paternò, F. and Santoro, C. 2004. Design and Development of Multidevice User Interfaces through Multiple Logical Descriptions. IEEE Transactions On Software Engineering, 30, 8 (August 2004), pp. 507-520. DOI= http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=1316868.

124. Montenegro Marín, C.E., Gaona García, P.A., Cueva Lovelle, J.M. and Sanjuan Martínez, O. 2011. Aplicación de Ingeniería Dirigida por Modelos (MDA), para la construcción de una herramienta de Modelado de Dominio Específico (DSM) y la creación de módulos en Sistemas de Gestión de Aprendizaje (LMS) independientes de la plataforma. Dyna, 2011, Nro. 169, pp. 43-52, ISSN; 0012-7353.

125. Eclipse. 2001. www.eclipse.org.

126. Ayotte, D., Vass, J., Mitchell, J. and Treviranus, J. Personalizing Interfaces Using an Inclusive Design Approach. 2014. Universal Access in Human-Computer Interaction. Design and Development Methods for Universal Access, Volume 8513, pp. 191-202. http://link.springer.com/chapter/10.1007%2F978-3-319-07437-5_19.

127. Creissac Campos, J. and Alves Mendes, S. 2011. FlexiXML A portable user interface rendering engine for UsiXML. Proceedings of UIDL'2011 - Software Support for User Interface Description Language - Interact'2011 Workshop, Lisboa, Portugal, 2011, (Eds.). ISBN 978-2-9536757-1-9.

128. Miñón, R., Moreno, L. and Abascal, J. 2013. A graphical tool to create user interface models for ubiquitous interaction satisfying accessibility requirements. Universal Access in the Information Society (UAIS), Springer, ISSN: 1615-5297, Volume: 12, Issue 4, pp. 427-439.

129. Abascal, J., Aizpurua, A., Cearreta, I., Gamecho, B., Garay-Vitoria, N. and Miñón, R. 2011. Automatically generating tailored accessible user interfaces for ubiquitous services. In The proceedings of the 13th international ACM SIGACCESS conference on Computers and accessibility (ASSETS '11). ACM, New York, NY, USA, pp. 187-194.

130. Kanai, S., Higuchi, T. and Kikuta, Y. 2009. 3D digital prototyping and usability enhancement of information appliances based on UsiXML. International Journal on Interactive Design and Manufacturing (IJIDeM), 2009, Volume 3, Issue 3, pp. 201-222.

131. Smith, A.C., Francioni, J.M. and Matzek, S.D. 2000. A Java programming tool for students with visual disabilities. Proceeding of th fourth International

ACM Conference on Assistive Technologies, ACM New York, pp. 142-148, ISBN: 1-58113-313-8. http://dl.acm.org/citation.cfm?id=354356.

132. W3C. 2012. ConcurTaskTrees. http://www.w3.org/2012/02/ctt/.

133. González-García, M., Miñon, R., Moreno, L., Martínez, P. and Abascal, J. 2013. A model-based graphical editor to design accessible media players. Journal of Universal Computer Science, 19, 18 (December 2013), pp. 2656-2676. DOI= http://www.jucs.org/doi?doi=10.3217/jucs-019-18-2656.

134. Limbourg, Q., Vanderdonckt, J., Michotte, B., Bouillon, L. and López, V. 2005. UsiXML: a Language Supporting Multi-Path Development of User Interfaces, in: Bastide, R., Palanque, P., Roth, J. (eds.) Engineering Human Computer Interaction and Interactive Systems. Springer, Heidelberg, LNCS, vol. 3425, pp. 200-220.

135. W3C. 2009. UsiXML. http://www.w3.org/2005/Incubator/model-based-ui/wiki/UsiXML.

136. Filament Group. 2009. http://filamentgroup.com/lab/update_jquery_ui_slider_from_a_select_element_now_with_aria_support/.

137. WAI_ARIA. 2011. http://www.w3.org/WAI/intro/aria.

138. jQuery UI. 2013. http://jqueryui.com.

139. CTTE. 2013. ConcurTaskTrees Environment. http://giove.isti.cnr.it/tools/CTTE/home.

140. Paternò, F., Santoro, C. and Spano, L.D. 2010. ConcurTaskTrees and MARIA languages for authoring service-based applications. Retrieved from http://www.w3.org/2010/02/mbui/soi/paterno-1.pdf.

141. Mori, G., Paterno, F. and Santoro, C. 2004. Design and development of multidevice user interfaces through multiple logical descriptions. Software Engineering, IEEE Transactions on, 30, 8 (2004), pp. 507-520. DOI= 10.1109/TSE.2004.40.

142. W3C. 2012. MARIA, Model-based lAnguage foR Interactive Applications. W3C Working Group Submission 3. http://www.w3.org/wiki/images/3/36/MARIA.pdf.

143. González-García, M., Moreno, L., and Martínez, P. 2014. Adaptation rules for Accessible Media Player Interface. XV International Conference on Human Computer Interaction (Interacción 2014), Puerto de la Cruz, Tenerife, Spain, September, 2014, ACM, New York, ISBN: 978-1-4503-2, Article No 5. http://dl.acm.org/citation.cfm?id=2662258.

144. W3C. 2013. WAI, Diversity of Web Users. http://www.w3.org/WAI/intro/people-use-web/diversity.

145. Papamarkos, G., Poulovassilis, A. and Wood, P.T. 2006. Event-Condition-Action Rule Languages for the Semantic Web. Current Trends in Database Technology (March 2006). EDBT 2006. Lecture Notes in Computer Science, 4254, pp. 855-864. DOI = http://link.springer.com/chapter/10.1007/11896548_64.

146. eWEEK. 2011. Java Use Increases Among Developers Worldwide: Survey. http://www.eweek.com/c/a/Application-Development/Java-Use-Increases-Among-Developers-Worldwide-Survey-844137/.

147. IEEE Spectrum. 2014. Top 10 Programming Languages. http://spectrum.ieee.org/computing/software/top-10-programming-languages.

148. Tiobe software. 2014. December Headline: R and Swift candidate languages of the year. http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html.

149. JAAPI. 2014. Package javax.accessibility. https://docs.oracle.com/javase/8/docs/api/javax/accessibility/package-summary.html.

150. Oracle. 2014. Java SE Desktop - Java Accessibility Utilities. http://www.oracle.com/technetwork/java/javase/downloads/downloads-jsp-138220.html.

151. O'Reilly & Associates. 2001. Chapter 1. The Java Foundation Classes. http://docstore.mik.ua/orelly/java-ent/jfc/ch01_01.htm.

152. Oracle. 2014. Java SE Desktop Accessibility. http://www.oracle.com/technetwork/java/javase/tech/index-jsp-136191.html.

153. Wieringa, R.J. 2009. Design Science as Nested Problem Solving. In Proceedings of the 4th International Conference on Design Science Research in Information Systems and Technology (DESRIST 2009), Philadelphia, May, 2009, ACM, ISBN: 978-1-60558-408-9, pp. 1-12.

154. Wieringa, R.J. 2010. Relevance and problem choice in design science. In the 5th International Conference on Global Perspectives on Design Science Research (DESRIST 2010), St. Gallen, Switzerland, June, 2010, LNCS 6105, Springer Verlag, ISBN: 978-3-642-13334-3, pp. 61-76.

155. Pfleeger, S.L, Atlee, J.M. 2006. Software Engineering: Theory and Practice. Second Edition, Prentice Hall PTR, Upper Saddle River, NJ.

156. Wohlin, C., Runeson, P., Höst, M. 2000. Experimentation in Software Engineering: An Introduction. Springer.

157. Pfleeger, S.L., Kitchenham, B.A. 2001. Principles of survey research part 1: turning lemons into lemonade. ACM SIGSOFT Software Engineering Notes, November, 2001, Volume 26, Issue 6, pp. 16-18.

158. Kitchenham, B.A., Pfleeger, S.L. 2002. Principles of Survey Research: Part 2: Designing a Survey. ACM SIGSOFT Software Engineering Notes, January, 2002, Volume 27, Issue 1, pp. 18-20; Part 3: Constructing a Survey Instrument. ACM SIGSOFT Software Engineering Notes, March, 2002, Volume 27, Issue 2, pp. 20-24; Part 4: Questionnaire Evaluation. ACM SIGSOFT Software Engineering Notes, May, 2002, Volume 27, Issue 3, pp. 20-23; Part 5: Populations and Samples. ACM SIGSOFT Software Engineering Notes, September, 2002, Volume 27, Issue 5, pp. 17-20; Part 6: Data Analysis. ACM SIGSOFT Software Engineering Notes, March, 2003, Volume 28, Issue 2, pp. 24-27.

159. Hassenzahl, M. 2004. The Interplay of Beauty, Goodness, and Usability in Interactive Products. Journal Human-Computer Interaction, December, 2004, Volume 19, Issue 4, pp. 319-349. DOI: 10.1207/s15327051hci1904_2.

Annex I

This work is based on the W3C Working Draft 25 September 2014.

W3C: Guideline 1.1 of User Agent Accessibility Guidelines (UAAG) 2.0

Guideline 1.1 - Provide access to alternative content [Reference for 1.1]

Summary: The user can choose to render any type of alternative content available (1.1.1) with an indicator that the alternative content is present (1.1.2) or a placeholder replacing the non-text content (1.1.3) . It's recommended that users can also choose at least one alternative, such as alt text, to be displayed by default (1.1.4). It's recommended that caption text or sign language alternative cannot obscure the video or the controls (1.1.5) and that the user can configure the size and position of media alternatives (1.1.6).

1.1.1 Render Alternative Content:

The user can choose to render any type of recognized alternative content that is present for a content element. (Level A)

- *Note*: It is recommended that the user agent allow the user to choose whether the alternative content replaces or supplements the original content element.

Reference for 1.1.1

1.1.2 Indicate Unrendered Alternative Content:

The user can specify that indicators be displayed along with rendered content when recognized unrendered alternative content is present. (Level A)

Reference for 1.1.2

1.1.3 Replace Non-Text Content:

The user can request a placeholder that incorporates recognized text alternative content instead of recognized non-text content, until explicit user request to render the non-text content. (Level A)

Reference for 1.1.3

1.1.4 Provide Configurable Alternative Content Defaults:

The user can specify which type(s) of alternative content to render by default for each type of non-text content, including time based media. (Level AA)

Reference for 1.1.4

1.1.5 Facilitate Clear Display of Alternative Content for Time-based Media:

For recognized on-screen alternative content for time-based media (e.g. captions, sign language video), the following are all true: (Level AA)

- **Don't obscure controls:** Displaying time-based media alternatives doesn't obscure recognized controls for the primary time-based media.
- **Don't obscure primary media:** The user can specify that displaying time-based media alternatives doesn't obscure the primary time-based media.
- **Use configurable text:** The user can configure recognized text within time-based media alternatives (e.g. captions) in conformance with 1.4.1.
- *Note*: Depending on the screen area available, the display of the primary time-based media may need to be reduced in size to meet this requirement.

Reference for 1.1.5

1.1.6 Allow Resize and Reposition of Time-based Media Alternatives:

The user can configure recognized alternative content for time-based media (e.g. captions, sign language video) as follows: (Level AAA)

- **Resize:** The user can resize alternative content for time-based media up to the size of the user agent's viewport.
- **Reposition:** The user can reposition alternative content for time-based media to two or more of the following: above, below, to the right, to the left, and overlapping the primary time-based media.

- *Note 1:* Depending on the screen area available, the display of the primary time-based media may need to be reduced in size or hidden to meet this requirement.
- *Note 2:* Implementation may involve displaying alternative content for time-based media in a separate viewport, but this is not required.

Reference for 1.1.6

W3C: Guideline 1.3 of User Agent Accessibility Guidelines (UAAG) 2.0

Guideline 1.3 - Provide highlighting for selection, keyboard focus, enabled elements, visited links [Reference for 1.3]

Summary: The user can visually distinguish between selected, focused, and enabled items; and recently visited links (1.3.1); with a choice of highlighting options that at least include foreground and background colors, and border color and thickness (1.3.2).

1.3.1 Highlighted Items:

The user can specify that the following classes be highlighted so that each is uniquely distinguished: (Level A)

- Selection
- Active keyboard focus (indicated by focus cursors and/or text cursors)
- Recognized enabled input elements (distinguished from disabled elements)
- Recently visited links
- Found search results

Reference for 1.3.1

1.3.2 Highlighting Options:

When highlighting classes specified by 1.3.1 Highlighted Items, the user can specify highlighting options that include at least: (Level AA)

- Foreground colors
- Background colors
- Borders (color, style, and thickness)
- Size when the indicator is an image
- Blink rate (where implemented)

Reference for 1.3.2

Annex II

UsiXML Meta-models

Task Model

This meta-model represents the task decomposition view of the application in the Tasks & Concepts layer of the UsiXML Framework. Figure 50 shows the Task meta-model.

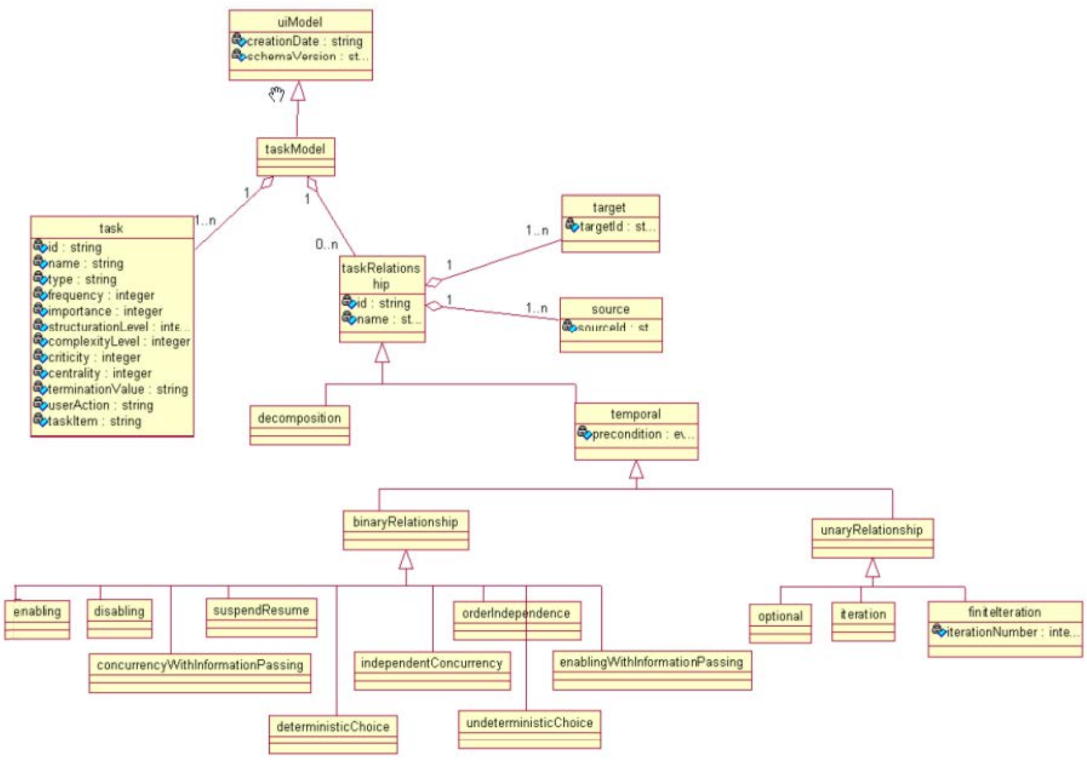


Figure 50 Task meta-model of UsiXML

As it can be seen in Figure 24, the task model of the proposal presented in this Doctoral Thesis is composed of two types of tasks, interaction and abstract, two types of unary relationships, optional and iteration, and two types of binary relationships, enabling and independent concurrency.

Therefore, so as to replicate the Task model presented in this proposal, not all of the design primitives of this meta-model are going to be used. Figure 51 shows the design primitives used in this proposal.

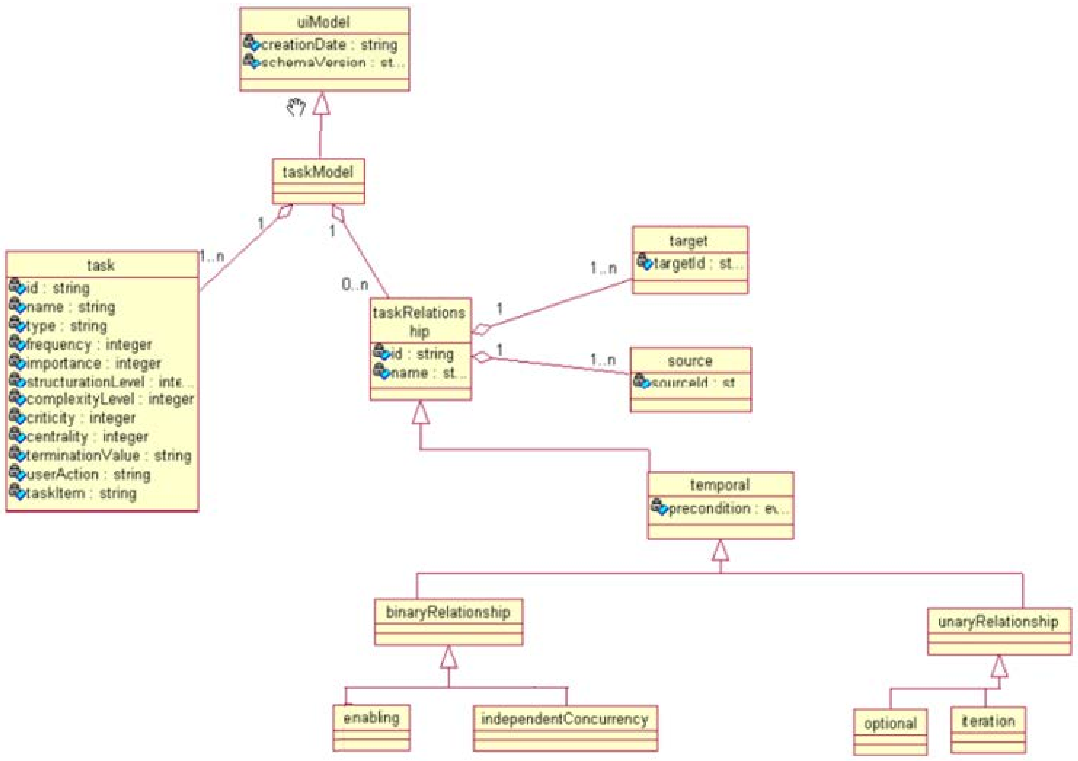


Figure 51 Restricted Task meta-model of UsiXML

Abstract User Interface Model

This meta-model is an expression of the UI in terms of interaction spaces (or presentation units), independently of which interactors are available and even independently of the modality of interaction (graphical, vocal, haptic,..). Being an interaction space a grouping unit which supports the execution of a set of tasks connected in a logical manner (see Figure 52).

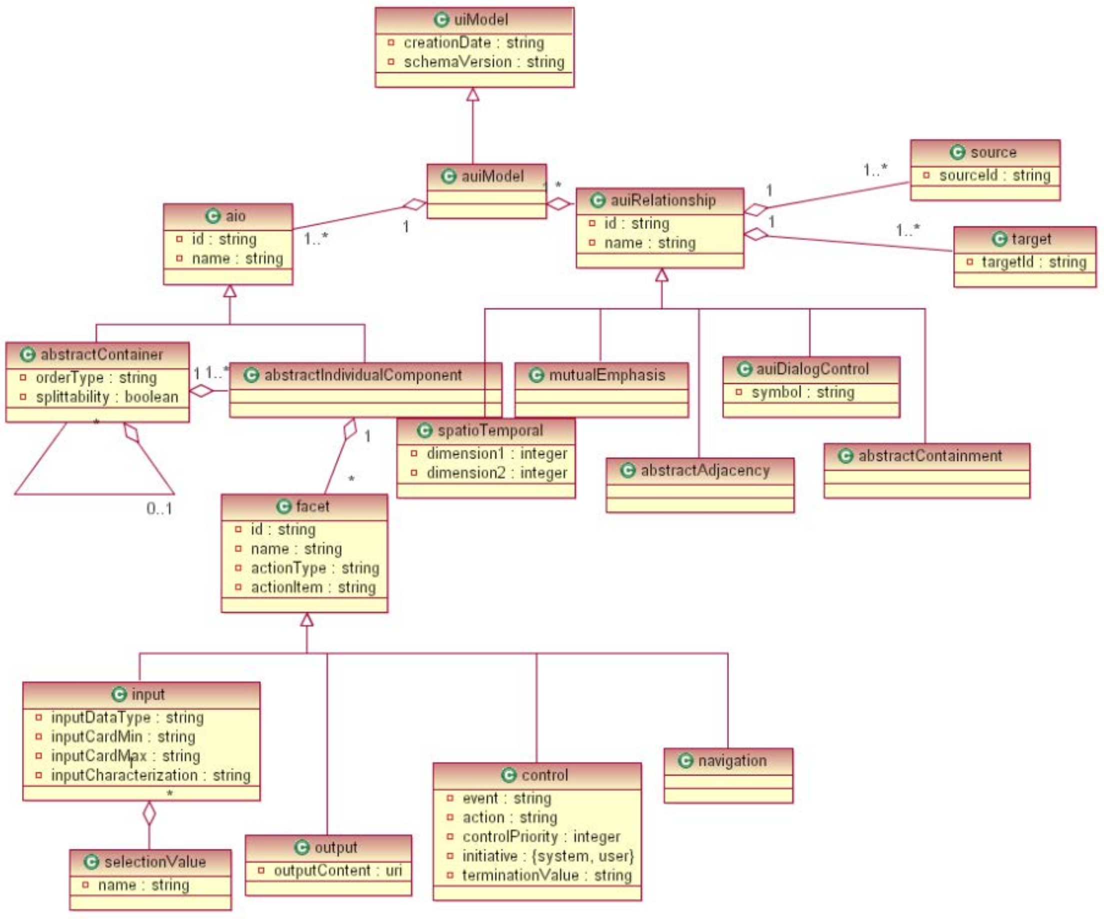


Figure 52 Abstract User Interface meta-model of UsiXML

The same as before, in the case of the Abstract User Interface, the model used in this proposal only contains Abstract Containers (ACs), Abstract Individual Components (AICs) and the abstract relationships. Then, Figure 53 shows all the compoments of the UsiXML Abstract User Interface used in this Doctoral Thesis.

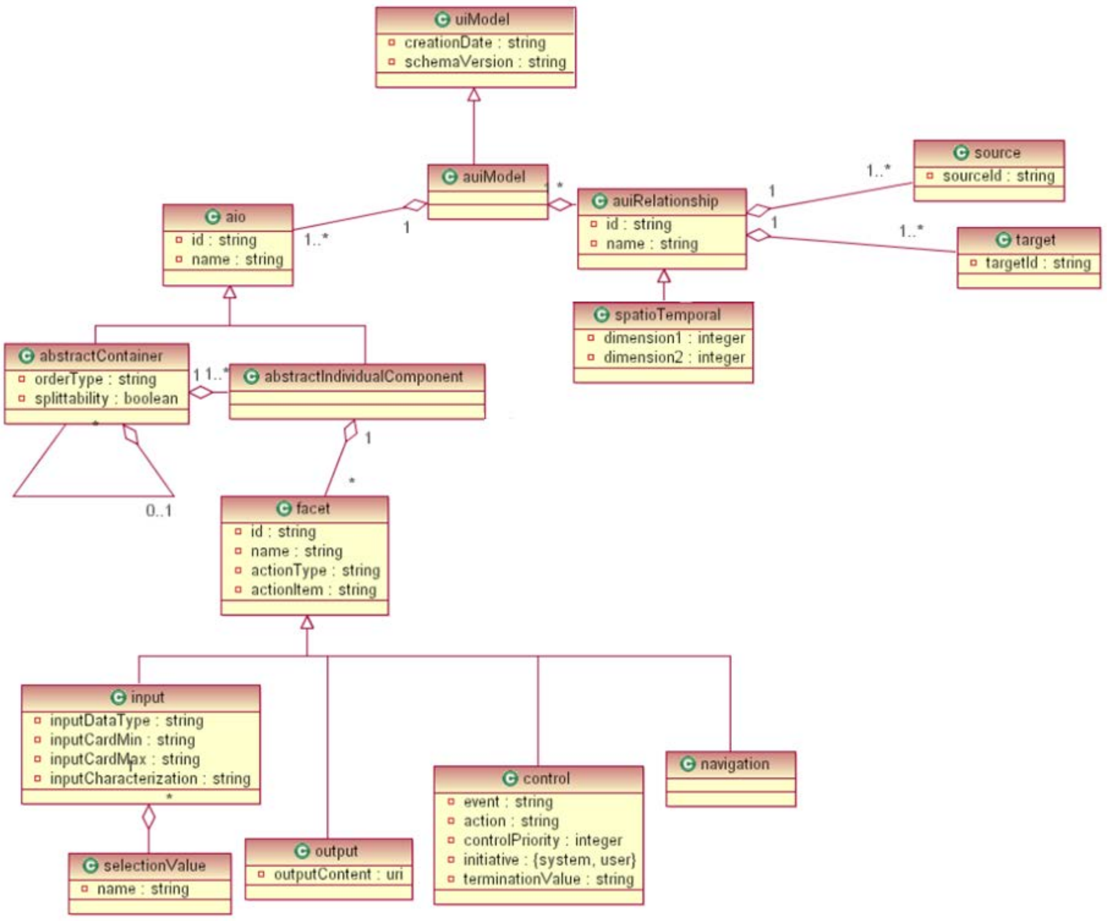


Figure 53 Restricted Abstract User Interface meta-model of UsiXML

Concrete User Interface Model

This meta-model is an expression of the UI in terms of "concrete interaction units" which depend on the type of platform and media available, besides, it has a set of attributes which define more concretely how it should be perceived by the user.

Figure 54 shows the first part in which this model has been divided.

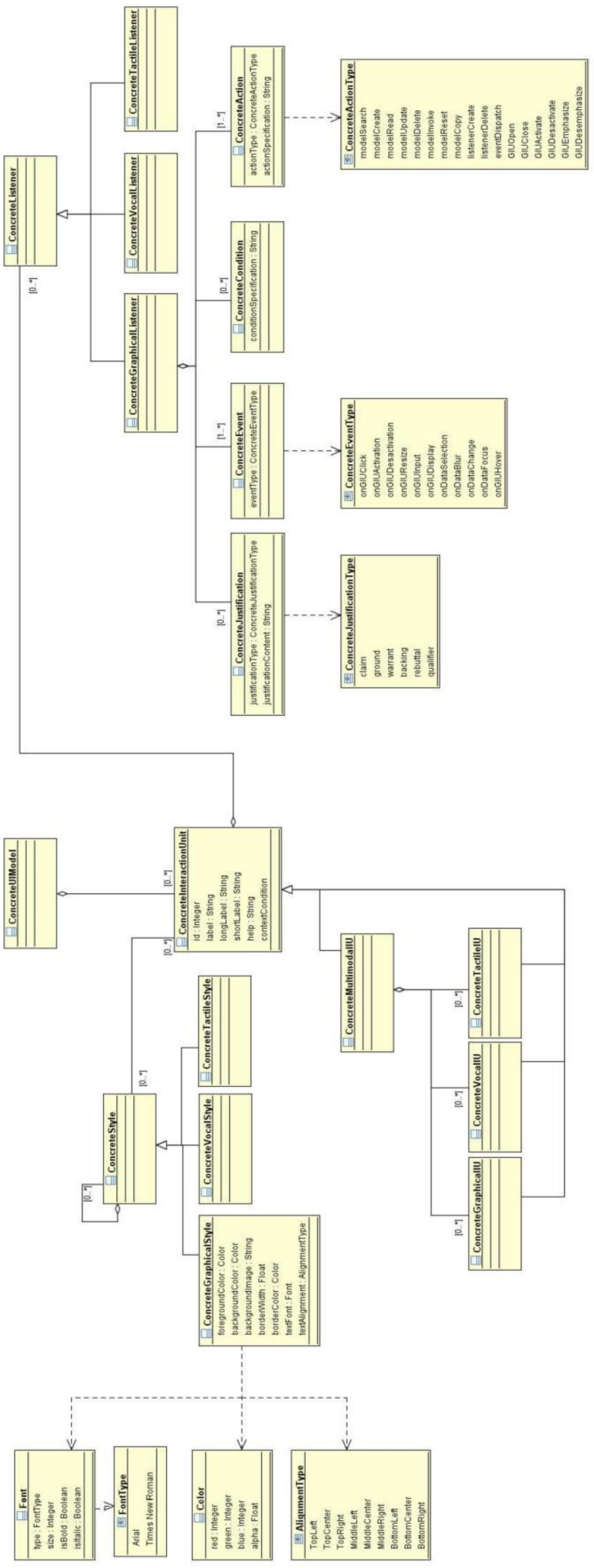


Figure 54 Concrete User Interface overview – Part I

In Figure 55, the second part of this meta-model is shown.

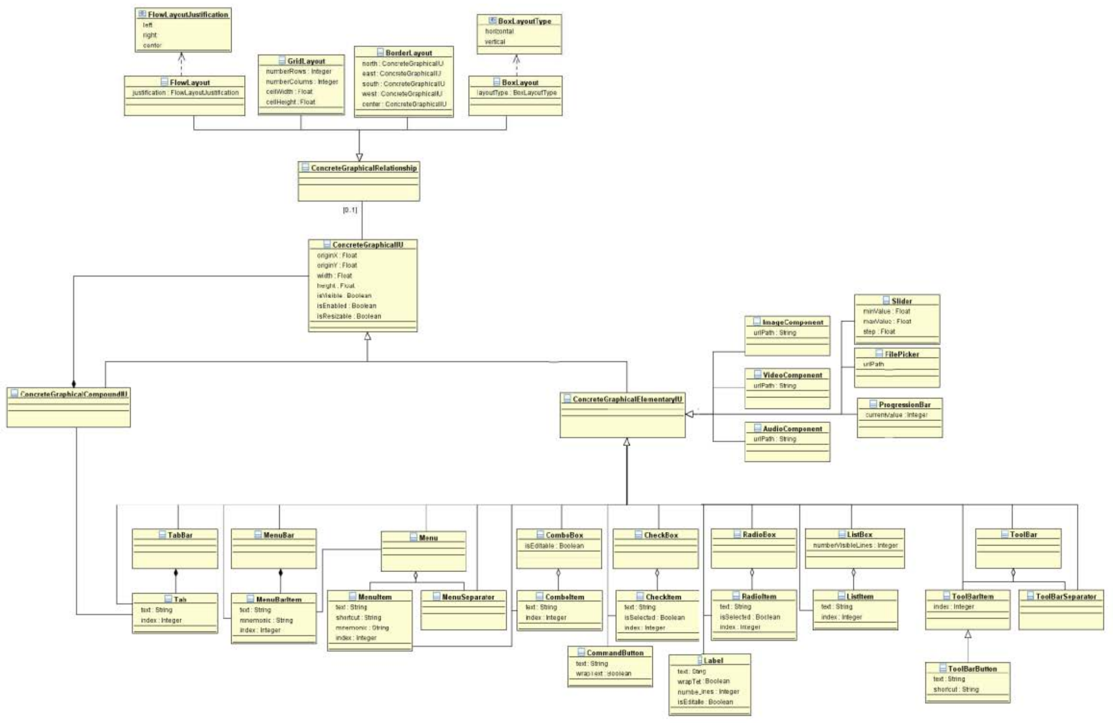


Figure 55 Concrete User Interface overview – Part II

Due to the fact that in this proposal, it has been used a restricted CUI, Figure 56 and Figure 57 show this CUI.

On the one hand, Figure 56 shows the specific elements of the CUI meta-model, such as the listeners or the style. Apart from these elements, this restricted meta-model also presents the modality selected (the graphical modality) to accomplish the user interface.

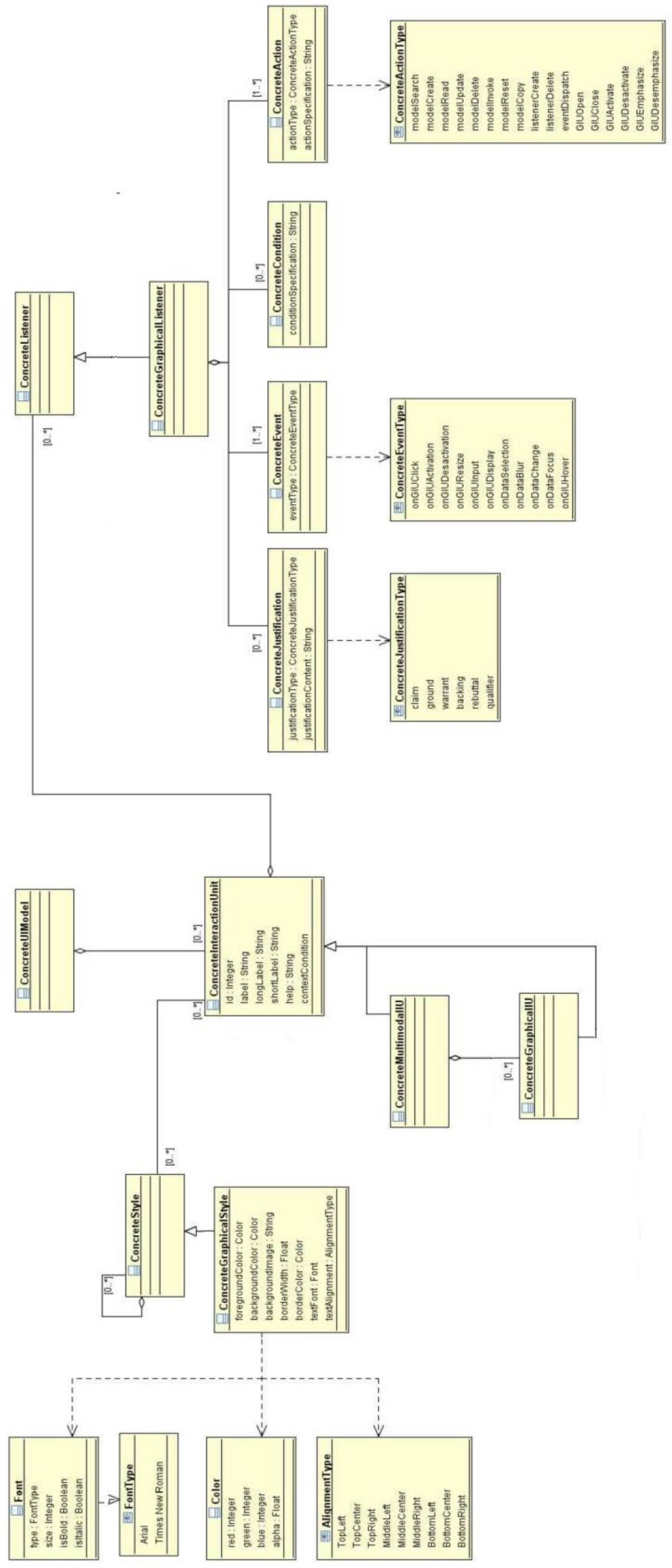


Figure 56 Part I of the Restricted Concrete User Interface meta-model of UsiXML

On the other hand, Figure 57 shows the elements used to replicate this proposal taking into account the chosen modality. Therefore, only the graphical elements which are going to be included in the graphical editor are presented.

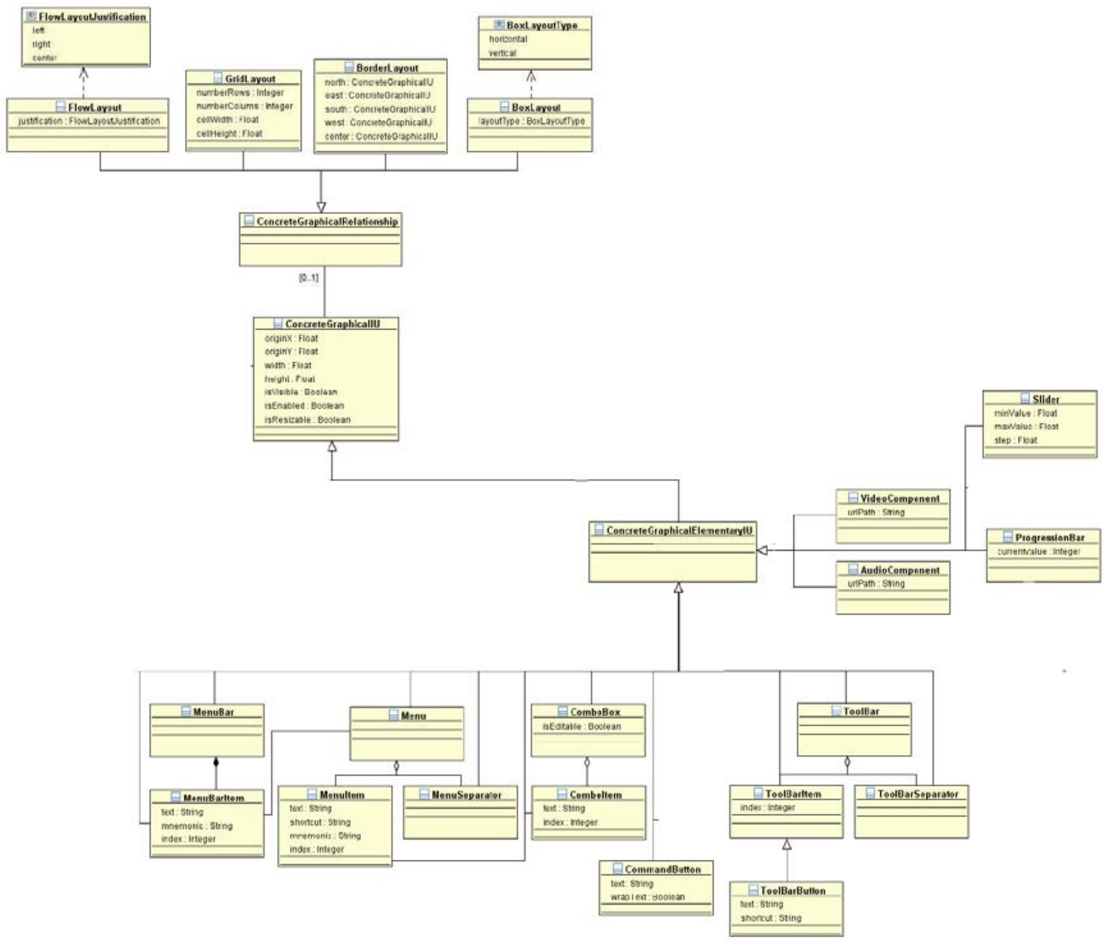


Figure 57 Part II of the Restricted Concrete User Interface meta-model of UsiXML

Annex III

MARIA Meta-models

Task Model

This meta-model represents the same task decomposition which it can be seen in the Task meta-model of UsiXML shown in Figure 51 in Annex II.

Abstract User Interface Model

This meta-model describes a UI by just referring to the semantics of the interaction, without considering a particular device capability, interaction modality or implementation technology (see Figure 58).

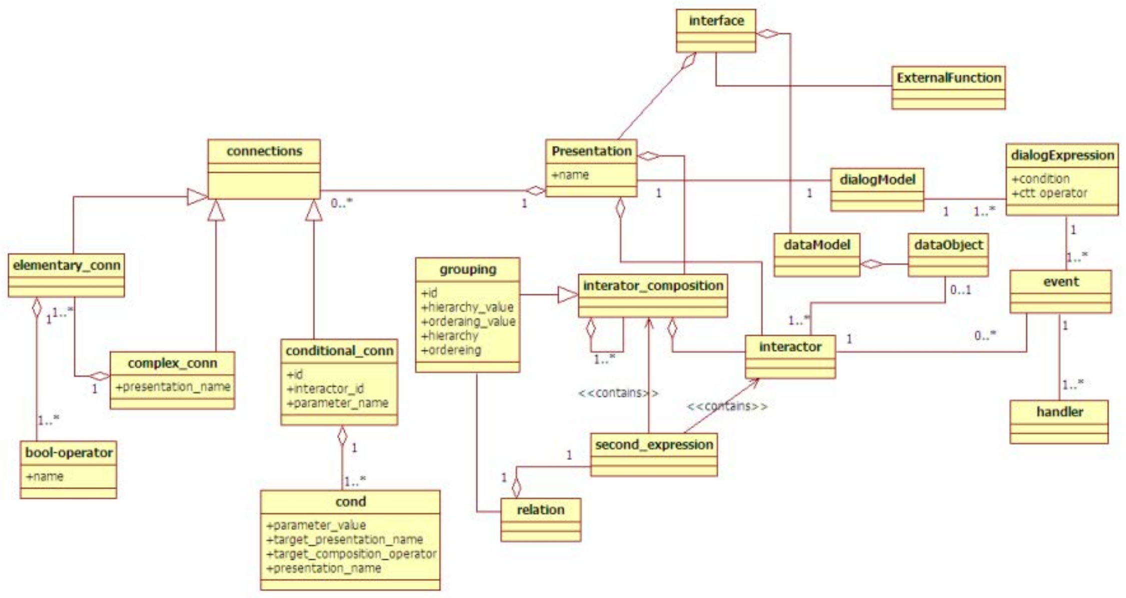


Figure 58 MARIA Abstract User Interface meta-model

Figure 58 shows the complete MARIA AUI meta-model. In the case of this Doctoral Thesis, not all the components of this meta-model are used, therefore, Figure 59 presents the group of components used in order to develop the AUI with MARIA.

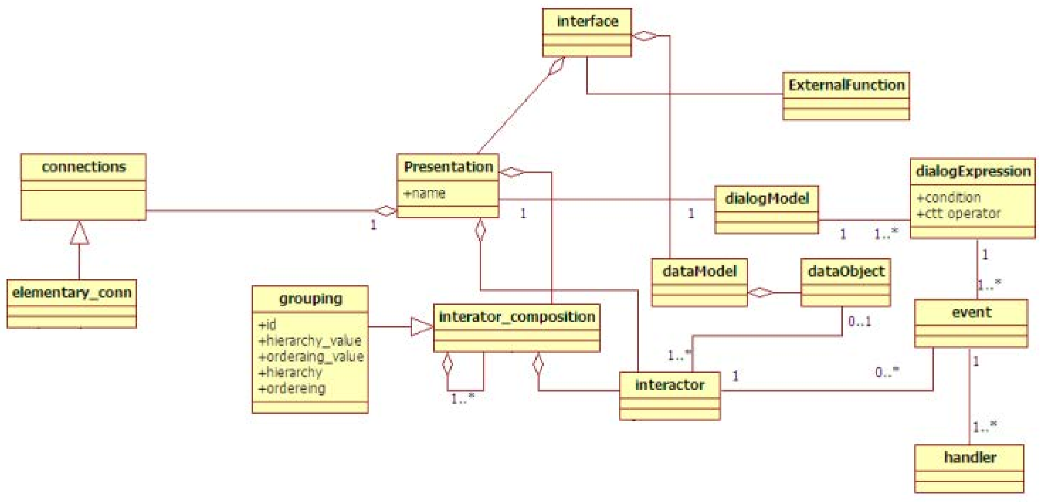


Figure 59 Restrictred Abstract User Interface meta-model of MARIA

Annex IV

Use Case: design of an accessible media player

USER IDENTIFIER: ________________

Introduction

It is wanted to design an accessible media player which delivers accessible video content. In order to do that, a graphical editor which generates accessible players is going to be used.

This graphical editor presents two manners of accomplishing the design. On the one hand, the designer can use a default player and introduces components from the editor´s palette (Default Player profile in the editor). On the other hand, the designer can choose one of the profiles previously established and provided by the editor.

Among the established profiles, four profiles are distinguished (besides the default profile):

- *Simplified Player*: the user interface presents only those elements essential to access playback.
- *Visual Access Player*: the user interface presents basic controls and elements regarding captions. This profile is oriented to be used by deaf or hearing impaired users. Apart from users that due to their preferences want to access video with captions.
- *Auditory Access Player*: the user interface presents basic controls and elements regarding audio description. This profile is oriented to be used by blind or visually impaired users.
- *General Play*er: the user interface presents all the elements placed in the palette.

Proof of concept

With the proof of concept, it is wanted to assess some aspects regarding the use of this graphical editor.

The user has to adopt the designer rol. It is going to ask her/him to accomplish some tasks using the graphical editor. These tasks are the design and generation of an accessible media player according to a set of features of the final user.

Task accomplishment

-----------------------------RUN THE EDITOR---------------------------

TASK 1

- Select the Default Player profile. Use the editor during three minutes.

TASK 2

1) Write down the hour: ________
2) Design a player taking into account that the final user to whom the player is directed is visually impaired.
3) Save the created editor. In order to do that, you have to access menu Profile→Save Custom Profile…. Two dialog box (two windows) are going to appear, in both cases, the file name has to be Task2_user_identifier.
4) Once the task has been finished, write down the hour: _______

TAREA 3

1) Write down the hour: ________
2) Design a player to support the largest posible number of users (with or without disability).
3) Save the created editor. In order to do that, you have to access menu Profile→Save Custom Profile…. Two dialog box (two windows) are going to appear, in both cases, the file name has to be Task3_user_identifier.
4) Once the task has been finished, write down the hour: _______

TAREA 4

1) Write down the hour: ________
2) Design a player to be used by elderly people who due to their cognitive difficulties should only contain the button that activates the playback and owing to their auditory impairment, the caption button.
3) Save the created editor. In order to do that, you have to access menu Profile→Save Custom Profile…. Two dialog box (two windows) are going to appear, in both cases, the file name has to be Task4_user_identifier.
4) Once the task has been finished, write down the hour: _______

----------------------------- TASK END ---------------------------

Thank you for your colaboration. Tell the interviewer that the tasks have been accomplished. Next, a questionnaire about your user experience using this graphical editor has to be filled.

QUESTIONNAIRE (1): USER EXPERIENCE

USER IDENTIFIER: ___________________

In this survey you are asked to select a value from 1 to 7 about 7 qualities. You will find two opposite words each one associated with one end of the scale, that is, the first word with value 1 and the second word with value 7. The average between them would be value 4.

Rate from 1 to 7 the following qualities of the graphic editor:

First quality: Technical or human (value 1 corresponds to technical, value 7 to human)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7

Second quality: Complex or simple (value 1 corresponds to complex, value 7 to simple)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7

Third quality: Impractical or practical (value 1 corresponds to impractical, value 7 to practical)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7

Forth quality: Tricky or direct (value 1 corresponds to tricky, value 7 to direct)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7

Fifth quality: Unpredictable or predictable (value 1 corresponds to unpredictable, value 7 to predictable)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7

Sixth quality: Confusing or clear (value 1 corresponds to confusing, value 7 to clear)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7

Seven quality: Difficult to control or manageable (value 1 corresponds to difficult to control, value 7 to manageable)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7

And last, a brief interview is going to be carried out to obtain your opinion about the tasks accomplishment and the graphical editor use.

QUESTIONNAIRE (2): TASKS ACCOMPLISHMENT

USER IDENTIFIER: _________________

TASK 2

2.1 Have you been able to accomplish the task completely?			
YES		NO	
If not, give a reason.			
2.2 Have you used some of the established profiles to accomplish the task?			
YES		NO	
Justify the answer.			

TASK 3

3.1 Have you been able to accomplish the task completely?			
YES		NO	
If not, give a reason.			
3.2 Have you used some of the established profiles to accomplish the task?			
YES		NO	
Justify the answer.			

TASK 4

4.1 Have you been able to accomplish the task completely?			
YES		NO	
If not, give a reason.			
4.2 Have you used some of the established profiles to accomplish the task?			
YES		NO	
Justify the answer.			

Do you think the use of the editor is simple?			
YES		NO	
Justify the answer.			

Do you think the contextual help is useful?			
YES		NO	
Justify the answer.			

Do you think/Are you sure that the media player created responds to the final user necessities which have been asked?			
YES		NO	
Justify the answer.			

Would you recommend the use of this graphical editor?			
YES		NO	
Justify the answer.			

-------- (only to be asked to technical participants) --------

What do you think about a tool which is able to generate players with accessibility requirements in this platform? Justify the answer.	
Give a scoring from 1 to 5	

QUESTIONNAIRE (3): USER IDENTIFICATION

USER IDENTIFIER: ________________

1. How old are you? ____________
2. Do you have any expertise in the use of a graphical editor? (for example, Microsoft Visio, diagramming programs, etc.)

    □ No expertise
    □ I have used it once or twice
    □ I use or have used it frequently
    □ I am an expert in using it

3. Do you have any knowledge about accessibility and Web Accessibility Initiative (WAI) standards?

    □ No knowledge
    □ Basic knowledge
    □ Intermediate knowledge
    □ Advanced knowledge

4. Do you have any knowledge about modeling and design methodologies based on models (MBD) and model driven development (MDD)?

    □ No knowledge
    □ Basic knowledge
    □ Basic knowledge, but not technical
    □ Intermediate knowledge
    □ Advanced knowledge

5. Have you ever used a media player? (for example Windows Media, Real Player, Quicktime, VLC, etc or web sites with players such as Youtube, Vimeo, etc.)

    □ Never
    □ Ever
    □ Once a week
    □ Several times a week
    □ All or almost every day

6. If you have used a player, what type of content do you listen to or watch?

    □ Only audio content
    □ Only video content
    □ Audio and video content
    □ I do not use any type of player

-------- (only to be completed by people with disabilities) --------

7. If you have a disability, please indicate which is/are:

   ____

   ____

8. If you need assitive technology, please indicate which is/are:

   ____

   ____