Introducing Accessibility Features in an Educational Game Authoring Tool

The <e-Adventure> Experience

Javier Torrente¹, José Ángel Vallejo-Pinto², Pablo Moreno-Ger¹, Baltasar Fernández-Manjón¹,³

¹Dpt. ISIA Universidad Complutense de Madrid ²Dpt. of Comp. Science, University of Oviedo, Asturias, Spain ³Lab of Computer Science, MHG-Harvard University

{jtorrente, pablom}@fdi.ucm.es vallejo@uniovi.es bfernandezmanjon@partners.org

Abstract—The introduction of new technologies in education raises accessibility issues, which can endanger the universal right to education for students with disabilities. Moreover, in education there is a shift to provide more dynamic and interactive content to the students and video games have been proposed to promote the development of new skills. In this paper we discuss the main challenges of introducing accessibility in game-based educational software and how they are being addressed within the <e-Adventure> authoring tool for educational games. The goal is to turn <e-Adventure> into a holistic solution for the development of accessible educational games considering both the accessibility of the games developed and the authoring tool itself.

Keywords—accessibility, <e-Adventure>, e-learning, distance learning, game authoring tools, game-based learning, online learning, videogames.

I. INTRODUCTION

During the last years new technologies and media have been proposed for education in order to prepare students for the challenges that this changing society demands. In this regard one of the proposals is to provide more dynamic and interactive content for students such as video games, which have been pointed out because of their potential as learning tools [1]. Nonetheless the application of video games in educational settings is very broad and diverse, including experiences where students interact with the video games but also other experiences where students learn in a creative way by designing their own video games [2].

As the academic interest in video games as educational materials grows, several proposals have emerged to facilitate the creation of educational video games and simplify their application in educational settings. One of the proposals is using authoring tools to produce educational games (e.g. Game Maker or Unity) since they reduce the production costs and facilitate the game development by people who do not have a technical background [3]. In this manner video games are brought closer to the educational system as teachers and students can develop their own games.

However, the accessibility of video games is an open issue, and it is even worse with game authoring tools. This problem becomes more evident when the games are introduced in e-Learning environments, where the accessibility of traditional content is reasonably covered by initiatives that facilitate access to the Web like WAI (which includes well-know lines of work such as WCAG or the ATAG - Authoring Tool Accessibility Guidelines and assistive technology that facilitates the use to major authoring tools for educational content (e.g. Microsoft Word ™). The purpose of this paper is to analyze the challenges posed by the introduction of accessibility in game authoring environments and educational video games. From this analysis we present how these problems are being addressed in the <e-Adventure> authoring tool, a platform for the creation of educational point-and-click games [4]. The aim is to provide an accessible video game authoring tool that produces accessible games without requiring a significant increase in the development cost.

II. ACCESSIBILITY IN GAMES AND DESKTOP AUTHORING TOOLS

In this section we analyze the major accessibility initiatives for both types of systems which are of particular relevance for this work: video games and desktop authoring tools.

A. Accessibility in Video Games

Most video games are not fully accessible for people with disabilities. This issue may be related to a lack of awareness of game developers and an unclear return of investment when the extra adaptation effort and the potential additional users are compared. The development of many accessibility features require technical solutions that increase the total development cost (e.g. speech synthesis, voice recognition). This issue is even more relevant in the field of educational games, where budgets are more constrained and accessibility is even more important. Therefore it is necessary to find new ways of reducing the cost of improving games accessibility.

Nevertheless there are different approaches to improve the accessibility of video games [5]. Focusing on hardware, an approach is to develop game pads adapted for people with different disabilities, new interaction devices such as systems that track the movement of the eyes or the head, or special keyboards. There are other cases where games are made compatible with or even integrate their own support tools such as screen readers, magnifiers, speech synthesizers, voice recognizers or virtual keyboards.

Another approach is to provide a multimodal design of the game interaction, providing various input and output channels for interacting with the game. For example it is typical to combine images with captions, sounds and voices [6], include support for voice commands [7], etc. However,
such approaches are often developed ad-hoc, providing
particular solutions for each game, which makes difficult to
reuse efforts. More general solutions, including frameworks
and other development tools are needed, especially for small
projects.

A general accessibility requirement for video games is to
let the player configure some of game settings. Certain
visual, hearing and motor problems can be dealt with by
making minor adjustments in the game such as the size of the
texts, the time thresholds for replying the attack of an enemy,
etc. [5].

B. Accessibility in Desktop Authoring Tools

The accessibility of desktop authoring tools is a complex
issue that has not received enough attention yet. Most
desktop applications that include accessibility features allow
the adaptation of the Graphical User Interface (GUI) by
configuring some parameters, such as the size and font color,
which can solve some common problems. In this context, but
focused on the Web, the ATAG initiative focuses on
providing guidelines for developing authoring tools for
accessible Web content. Between the guidelines provided in
ATAG, the accessibility of the authoring tool is also
considered as a strong point.

Another common strategy is to seek compatibility with
external assistive tools like screen readers (e.g. JAWS). This
is the approach that has been applied, for example, to
increase the accessibility of well-known tools like Microsoft
Word™ [8] or Microsoft Excel™ [9].

However, this approach is insufficient to make accessible
other complex tools such as Paint™, Adobe PhotoShop™ or
Adobe Flash™. First, these tools are usually focused on the
creation of highly visual content (e.g. images, photographs or
maps). Second, the amount of editable information that the
tools need to provide to the user is very high, which makes
navigation through the elements more difficult for people
with visual disabilities. Third, these tools usually implement
interaction modes that are not accessible because they
require eye-hand coordination, such as *drag & drop* [10]. In
this type of tools, the development of accessibility to target
visual disabilities can become an arduous and expensive
task. In this sense the tools that have achieved greater
success in accessibility terms are the ones which combine
support from external assistive technologies with the
introduction of facilities to simplify the edition of content.
For example, the templates that Microsoft Power Point™
includes facilitate the creation of presentations by visually
disabled authors as they do not have to deal with the spatial
organization of the contents of each slide, which is specified
by the template.

Game authoring tools are a good example of these highly
visual applications. Games are a rich and complex medium,
and therefore game authoring tools need to present a lot of
information and features that the user has to configure to
compose the final game, such as characters, objects,
conversations, cut-scenes or scenes. The simplification of the
tools, removing or hiding configuration options that are not
indispensable could address this issue, at least partially.
Nevertheless, this approach is not always feasible as an
oversimplification of the tool could limit the expressive
resources available to create the video games, leading to the
production of poor video games in terms of engagement and
educational value. Therefore one of the main challenges is
how to structure the information that the tool presents so all
users can navigate through the elements of the GUI using
different input devices (e.g. mouse, keyboard or a
microphone).

III. THE <e-ADVENTURE> PLATFORM

<e-Adventure> is an authoring tool for creating
educational video games and simulations [4]. The aim of <e-
Adventure> is to facilitate the development of video games
for people without a technical background (i.e. mainly
teachers or students). In addition <e-Adventure> is
multiplatform (developed in Java), allowing the games to run
on Windows, Linux, Mac or even the web.

A. The <e-Adventure> Games

As a strategy to reduce the development costs and the
complexity of the authoring tool, <e-Adventure> focuses on
the genre of *point-and-click* adventure games, like the
Monkey Island™ or Myst™ sagas.

![Figure 1](image1.png)

Figure 1. Screenshot of a third-person <e-Adventure> game. It shows a
two-button contextual menu with the available interactions over an object.
The cursor changes when the mouse is over an interactive element.

Therefore <e-Adventure> games are composed by scenes
(structured over a background image), which are
interconnected (as defined by the game author) resulting in a
2D navigational environment. In this environment the player
has to solve the challenges that game author defines.

In <e-Adventure> games, in order to discover the
interactive elements, players need to explore the scene using
the mouse. When the mouse moves over an interactive
element the cursor changes and a short text is displayed,
providing in this manner visual feedback (Figure 1).

When players click the mouse on an interactive element a
contextual menu appears, including the different actions that
can be performed with it (Figure 1).

B. The <e-Adventure> Authoring Tool

As Figure 2 depicts, the <e-Adventure> authoring tool is
a visual editor structured in three different areas: a toolbar on
the upper section, a panel to access the main elements

![Figure 2](image2.png)
that form a game on the left section (e.g. the scenes that represent the virtual world, objects and characters, etc.) and an edition panel on the central section to configure the properties of the element selected on the left panel. The edition panel is different for each kind of element, but is generally made up of tabs, tables and nested sub-panels in which the editable properties of each element are grouped. Therefore, navigation through this interface using the keyboard is hard as all the elements of the GUI must be wandered in a linear sequence.

The configuration of the scenes is done through several drag & drop panels like the one displayed in Figure 2. When a scene is selected the central panel is loaded with the background image of the scene. Upon this image, the game author can drag characters and items onto the scene. Following a similar idea the game author can define interactive regions on the scene (i.e. portions of the background image delimited by polygons) which are placed using the mouse.

IV. INTRODUCING ACCESSIBILITY IN <E-ADVENTURE> GAMES

Our first approach to the introduction of accessibility in <e-Adventure> was focused on the games. As we describe in [11], once the game is designed the author of the game can enable a set of input and output modules and other support tools (e.g. a screen magnifier) that are already bundled in the platform.

Currently <e-Adventure> has been powered with three different input layers: a Mouse Interface Layer (MIL), a Voice Interface Layer (VIL) and a Keyboard Interface Layer (KIL). MI provides the classic point-and-click interaction through the mouse. VI and KI provide new interaction mechanisms that are especially suitable for people with motor or visual disabilities. Users who cannot move a computer mouse can interact with the game through voice commands using the VI. Users who are blind or have a severe visual disability (i.e. those that are used to interact with computers using the keyboard) profit from the KI. Both interfaces recognize the same command patterns in natural language but using different input devices. A typical command matches the structure Verb + Article (optional) + Element + Conjunction (optional) + Element2 (optional). For instance, valid commands are “Describe the scene” or “Use key with locker”.

Additionally <e-Adventure> games can also be enhanced with auditive feedback for people with visual disabilities, including a speech synthesizer that transforms all text lines that present information in the game to voice and descriptive sounds (e.g. beeps) for common actions (e.g. enter/exit the game menu).

Furthermore, to deal with cognitive disabilities <e-Adventure> includes a game adaptation engine that can modify the game experience according to characteristics of the player (e.g. by skipping some initial levels, avoid puzzles that are too challenging, etc.) [12]. To deal with needs of users with motor or cognitive disabilities <e-Adventure> allows configuring some game parameters, like the double click gap.

All these accessibility features can be introduced in the games using the game authoring tool, which reduces the cost of introducing accessibility in educational games. Nevertheless we are also improving the accessibility of the authoring tool following the ideas of the ATAG.

V. INTRODUCING ACCESSIBILITY IN THE <E-ADVENTURE> AUTHORING TOOL

The <e-Adventure> editor exemplifies many of the features that hinder the introduction of accessibility in this type of software. Among the most common problems, we
highlight the configuration of visual elements that make up the virtual world of the game, and the way to structure the editable information.

First, the configuration of visual elements poses a serious accessibility problem, especially for people with visual disabilities since the game author must be able to perceive the virtual world and the position of the elements that compose the world as a whole, (not in isolation) without the support of vision. One way to facilitate the configuration of the game universe is to use a 2D representation instead of 3D. However, it is still necessary to solve the problem of defining accessible mechanisms for placing the elements on the scenes of the game. This problem occurs recurrently in the <e-Adventure> editor, as well as in other editors, where the panels that allow the configuration of the scene are based on drag-and-drop or point-and-click interaction.

Second, how to structure the large amount of information that is needed to configure the game in a way that is accessible through the keyboard is a major issue. In this section we describe how we are addressing them in the <e-Adventure> editor.

A. Introducing GUI Settings

The first strategy that we have followed to improve the accessibility of the <e-Adventure> editor is to allow the configuration of some parameters of the Graphic User Interface (e.g. the text size). As discussed in section II this is one of the most common strategies. However, there are several approaches for configuring the user interface.

On the one hand, one option is to give the user the possibility to choose the settings manually. On the other hand there is the option to automatically load the settings according to the user profile that is stored in the configuration of the operative system.

As <e-Adventure> is multipurpose we initially chose the first alternative, adding an “Accessibility” drop-down menu in the toolbar where the user can select the font size (normal or large) and the display mode (normal or high contrast). Moreover, in this menu we also include the options that allow the configuration of the accessibility features of the games, as described in section IV.

B. Structure and Navigation through the Information

To cope with the need of structuring the information of the editor in an accessible way we have followed two main strategies. First, we have reduced the number of nested elements of the editor (e.g. replacing tables by more accessible components). This allows navigating throughout the interface more easily using the keyboard. Second, we have implemented a tree structure of “focus loops” that improves even further the navigation using the keyboard.

A focus loop sets the order in which the elements of a specific part of the interface can be inspected with the keyboard. We have adopted the Tab key to cycle between elements within the active focus loop (tree level). When one of the interface elements consists of a larger group of components, the user can go down in the tree and start exploring the components of this element using the ctrl + Tab combination, changing the active focus loop accordingly. To return the focus to the upper level the combination ctrl + shift + Tab can be used.

However, since navigation through the entire focus tree is complex, we have also included keyboard shortcuts to go directly to certain points of the “focus tree”. This option is useful when users want to edit an element that is far in the focus loop tree from the active focus loop. For instance, when users are placing the elements on a specific scene they can jump directly to the left panel using ctrl+left arrow and start editing other game elements, instead of going backwards in the tree using ctrl + shift + tab. Additionally typical shortcuts to access the menus of the toolbar are available using the convention alt + letter.

C. Multimodal Feedback

One of the major problems that pose the inclusion of accessibility for people with visual disabilities is to provide an alternative (multimodal) return of information. For this purpose, we have implemented a speech synthesis module using Microsoft’s Speech Synthesis engine (Speech API 5.4), which is wrapped with the Com4J library that allows invoking Microsoft’s COM interfaces from Java code.

Through this system the user can configure the speech synthesis module using his/her own support tools, like the accessibility features included in Windows Vista (or higher) or the popular screen reader JAWS.

This system is used to solve the common “where-am-I” problem. That is, complex navigation structures make it hard to remember what piece of information is being edited (i.e., what is the active focus cycle). As a solution, we added a special command that indicates via speech synthesis a summary of the current location within the focus tree.

In addition we have also included special sounds associated with relevant system events, such as “Image loaded successfully (or not)”, “Image linked successfully” (when you configure the navigation through scenes) or “Element referenced in the scene” when an object or character has been successfully placed.

D. Setting the Game Scenes

Another relevant problem is how configure the visual elements that make up the virtual world in an accessible way for people with visual disabilities. To achieve this we have implemented an alternative to typical drag & drop interaction.

The goal is to provide aids for people with visual disabilities that allow them to place elements in the scene as a whole. In this way a visually disabled user could easily place an element in the scene taking into account both the position of other interactive elements and what the background image of the scene looks like. For instance, if there is a door drawn in the right side of the scene, the user should be able to place an interactive rectangle over this area of the scene and link it to other scene.

We have taken several actions to facilitate this issue. First of all, elements in drag & drop panels are also accessible using the keyboard as they have been added to the focus loop tree described in V.B. Once the drag & drop panel that contains the elements of the scene is reached, users can add
and remove elements and modify the location of the elements in the scene using the arrow keys.

Second, we have added two special aids that facilitate the placement of elements in the context of a specific scene. On the one hand, we have developed a tagging system to attach additional accessibility meta-data to the background images that will be used to configure the scenes of the game. A tag is a special mark that is attached to a background image and indicates the location of the elements that compose the image (see Figure 2). The idea is that the resource provider can annotate the position of relevant elements in the art resources using a special tool, thus allowing visually disabled students to participate in game creation activities. Besides a general alternative description of the scene for the speech synthesizer can be also provided with the tool, which will generate a XML document with all the meta-data specified for the scene.

The <e-Adventure> authoring tool allows users to locate elements in the scene in relation to the position of the tags and other elements of the scene using special keyboard shortcuts. For example, in Figure 2 users could execute actions such as “place element on the left of the door” or “place element behind the bed”.

On the other hand, we have implemented a system of nine imaginary regions that divide the scene (Figure 2). The 9 regions are fixed for all scenes and games. Therefore they can be used to place elements in the scene when no tags have been provided.

Finally, when users have selected a specific element of the scene (e.g. a character, an item, an interactive area, etc.) they can use a special command (Ctrl+Shift+R) that provides the position of the element in relation to other elements through the speech synthesizer (e.g. the bookcase is on the left side of the exit door but on the right of the fishbowl, table and bed).

VI. DISCUSSION AND FUTURE WORK

The introduction of new technologies in education must always take into account the needs of both students and instructors with disabilities. There are still many open accessibility issues to be addressed to introduce highly visual and interactive technologies in the educational system in an inclusive manner. We have addressed these accessibility problems in the video games and game authoring tools by introducing accessibility features in <e-Adventure>.

The introduction of accessibility in an authoring tool as <e-Adventure> raises two different challenges. First, to make accessible the <e-Adventure> editor, which is a highly visual, drag & drop desktop authoring tool. Second, to make the content produced with the tool (i.e. the games) accessible by including easily configurable features in the platform that facilitate the introduction of accessibility in the games with a marginal extra cost.

Despite the work done, the methodologies and solutions proposed have various limitations. On the one hand, the introduction of accessibility in <e-Adventure> has focused on a small subset of primarily visual disabilities. Next versions of the platform will pay special attention to hearing impairment, motor and cognitive skills.

On the other hand, <e-Adventure> is a multiplatform tool that can be run on Windows, Mac, Linux and even on the Web (only the games, not the editor). From a technical perspective, it is still uncertain how to develop accessibility for Java applications that run on different systems. Therefore, accessibility features only run on Windows and other approaches should be considered for other platforms.

Nonetheless many of the problems and solutions proposed for the <e-Adventure> editor are applicable to other desktop authoring applications with similar characteristics (drag & drop interaction, highly visual content, etc.). One of our lines of future work is the generalization of our approach to other tools and paradigms.

ACKNOWLEDGMENT

The Spanish Committee of Science and Technology (TIN2010-21735-C02-02) and the Ministry of Industry (TSI-020110-2009-170, TSI-020312-2009-27) have partially supported this work, as well as the Complutense University of Madrid and the Regional Government of Madrid (research group 921340 and project e-Madrid S2009/TIC-1650), and the PROACTIVE EU project (505469-2009-LLP-ES-KA3-KA3MP) and the GALA EU Network of Excellence in serious games (FP7-ICT-2009-5- 258169). Special thanks to Elisa Sanz and Technostie (ONCE group) for their contributions.

REFERENCES