Widening Access to Virtual Educational Scenarios

562463-EPP-1-2015-1-UK-EPPKA2-KA

D3.1 Installation and Maintenance Guidelines

Deliverable number	D3.1
Delivery date	June, 2018
Status	Final
Authors	Martin Adler (Instruct), Panagiotis Antoniou (AUTH), Andrzej Kononowicz (KI), Sheetal Kavia (SGUL), Supriya Krishnan (SGUL), Dimitris Spachos(AUTH), Natalia Stathakarou (KI), Daniel Schwarz (MU), David Topps (UC), Luke Woodham (SGUL).

Funded by the Erasmus+ Programme of the European Union

TABLE OF CONTENTS

1. INTRODUCTION ... 4
1.1. Rationale ... 4
1.2. Scope and Structure of this Deliverable Report ... 4
2. OVERVIEW OF THE WAVES TECHNICAL TOOLKIT ... 5
2.1. Usability and Accessibility Toolkit Inventory Table ... 5
2.2. Additional Information ... 6
3. DETAILED INSTALLATION AND MAINTENANCE GUIDELINES ... 7
3.1. CASUS GUI Translation Tool ... 7
3.1.1. Repository ... 7
3.1.2. What is this repository for? ... 7
3.1.3. Technical Installation Instructions ... 7
3.1.4. Main functionality ... 8
3.1.5. Contribution Guidelines ... 8
3.2. Internationalisation Support for CASUS ... 8
3.2.1. Online Repository ... 8
3.2.2. Online Demonstration ... 8
3.2.3. What is in the Box? ... 8
3.2.4. Contribute Guidelines ... 9
3.3. Working VS Player Mobile Theme for Open Labyrinth ... 9
3.3.1. Repository ... 9
3.3.2. Online Demonstration ... 9
3.3.3. Description ... 9
3.3.4. Technical Installation Instructions ... 9
3.3.5. Development Server ... 11
3.3.6. Code Scaffolding ... 11
3.3.7. Additional Tools ... 11
3.3.8. What is in the Box? ... 11
3.3.9. Tools ... 11
3.3.10. Libraries ... 12
3.3.11. Contribution guidelines ... 12
3.4. New VS Pathway Review Mechanism for Learners ... 12
3.4.1. Repository ... 12
3.4.2. Description ... 12
3.4.3. Technical installation instructions ... 12
3.5. Heuristic Evaluation of Open Labyrinth 3 Usability ... 13
3.5.1. Repository ... 13

3.5.2. Description ... 13
3.5.3. Contribution Guidelines ... 13
3.6. Prototype of guided tour mechanism for Open Labyrinth 3 ... 13
3.6.1. Repository ... 13
3.6.2. Description ... 13
3.6.3. Online Demonstration ... 13
3.6.4. Technical Installation Instructions ... 13
3.7. Summary of Recommendations and Improvements of Accessibility ... 14
3.7.1. Example for the CASUS case selection page: ... 16
3.8. Challenges of Export/Import using the MedBiquitous Virtual Patient ANSI/MEDBIQ VP.10.1-2010 Standard ... 17
3.8.1. What is the MedBiquitous Virtual Patient ANSI/MEDBIQ VP.10.1-2010 Standard? ... 17
3.8.2. Why do the Virtual Scenarios in the WAVES project use a Standard called Virtual Patients? 17
3.8.3. What VS Systems Support the Standard? ... 17
3.8.4. What can the Standard do? ... 17
3.8.5. What Aspects of a VS does the standard Include? ... 18
3.8.6. Where can I find examples of the standard? ... 18
3.8.7. How can I use the standard to play a Virtual Scenario? ... 18
3.8.8. What can I expect when I import a VS? ... 18
4. APPENDIX ... 20

1. INTRODUCTION

1.1. Rationale

The end-user specification for the WAVES user requirements is presented in D1.1, in the form of user stories along with the importance of the requirements and its relevance to the objectives of the project. These specifications have been summarized to the following user needs (as presented in D1.1):

- “As a learner I want Scenario-Based Learning (SBL) to be delivered through realistic means and media with impactful consequences so that it is engaging and useful for real life situations.”
- “As an educator I want an easy to use and powerful Scenario-Based Learning (SBL) authoring toolkit that can integrate realistic authentic means of interaction for impactful SBL educational episodes.”

From these pedagogy focused aspects, some purely technical requirements emerged and these have been pivotal during the development of the technical implementations and specifications. Again, summarizing from D1.1:

- Learners wish SBL to be free from technical barriers, so that focus can remain on learning.
- Educators need SBL to be easily “integrate-able” to existing platforms and Learning Management Systems (LMSs) in order to have a transparent authoring environment and to be able to integrate SBL for building complex eLearning scenarios and environments. They also consider desirable the SBL content to be accessible across platforms and media (PC, tablet, mobile).

With these general aims in mind, the WAVES consortium developed a technical toolkit, described its specifications and delivered exemplar implementations. This document provides the necessary installation and maintenance guidelines for all software tools developed within the WAVES project.

1.2. Scope and Structure of this Deliverable Report

This deliverable provides the necessary guidelines for the installation and maintenance of the technical toolkit developed within the WAVES project. The technical toolkit facilitates access to scenario-based learning and thus contributes the core objectives of the project. This is achieved both by proposing and by implementing improvements in the usability of existing virtual scenario delivery systems and other relevant tools. It is also achieved through the development of new tools that can both be integrated in the existing technical virtual scenario infrastructure and can stand alone as specifications and implementations that facilitate user engagement with the topic of virtual scenarios.

By its nature, the toolkit is not an integrated piece of software. It consists of several components and implementation specifications. As such, the installation and maintenance of each one is covered individually in its own section. The two virtual scenario systems used within this the project are OpenLabyrinth (OL) and CASUS.

2. OVERVIEW OF THE WAVES TECHNICAL TOOLKIT

2.1. Usability and Accessibility Toolkit Inventory Table

Section	Description	Contribution	VS tool	Outcome/Form	Repository
3.1	CASUS GUI translation tool	Instruct	CASUS	Java code (JSF web app)	https://bitbucket.org/gulpipvt/i18properties
3.2	Internationalizati-on support for CASUS	UJ, MU, Isfahan, Instruct	CASUS	Translation files + Java code improvements (JSF web app)	Language files: Access on request. Polish translation maintained by UJ - for improvements contact: support@casus.net Tested converters RTL CSS: http://rtlcss.com CSS-Flip: https://github.com/twitter/css-flip
3.3	Working VS player mobile theme for Open Labyrinth	AUTH	OL	HTML, CSS + JavaScript	https://github.com/wavesnetwork/mobileplayer
3.4	New VS pathway review mechanism for OL 3.1	KI	OL	HTML, CSS + JavaScript	https://github.com/wavesnetwork/vispath
3.5	Heuristic evaluation of OLab 3 usability	AUTH	OL	Report	https://github.com/wavesnetwork/heuristic-evaluation
3.6	Prototype of guided tour	MU	OL	PHP+JavaScript	https://github.com/wav

	mechanism for OL 3.5			(Intro.js)	esnetwork/guidedtour
3.7	Summary of recommendatio n-ns and improvements of accessibility (for users with visual impairments) which are required by law to make the system accepted for use in public institutions				
3.8	Challenges of Export/Import using the MedBiquitous Virtual Patient ANSI/MEDBIQ VP.10.1-2010 standard				https://github.com/wavesnetwork/mvp-standard

2.2. Additional Information

A brief description of the toolkit can be found online at the WAVES project website:

http://wavesnetwork.eu

and in the GitHub project repository:

https://wavesnetwork.github.io

All available tools are available under the following MIT license:

3. DETAILED INSTALLATION AND MAINTENANCE GUIDELINES

The two exemplar Virtual Scenario authoring systems used in WAVES have been developed and running for several years. The installation and maintenance guidelines for the current version of OpenLabyrinth can be found in the appendix of this deliverable. As the CASUS authoring tool is a commercial product which is maintained and serviced by Instruct AG there are not installation and maintenance guidelines for this.

3.1. CASUS GUI Translation Tool

3.1.1. Repository

https://bitbucket.org/gulpipvt/i18properties/

3.1.2. What is this repository for?

Minimal Translate Tool as a WEB-APP tested within a TOMCAT 8 container
The tool is in version: 0.1.

3.1.3. Technical Installation Instructions

Clone this project repository

bash git clone https://bitbucket.org/gulpipvt/i18properties/

Directory containing the property files, have to be external, please configure in WEB-XML context-param entries for:

```
properties_path: directory
properties_name: primary name of i18 properties
```

For Google Translation support, you'll need to keep the Google libraries ("xx-translate-1.0-jar-with-dependencies.jar"), otherwise delete this. Additionally you'll need a Google account resulting in a APIU key, On JAVA startup this key can be activated by a -DGOOGLE_API_KEY=<KEY> directive

3.1.4. Main functionality

- The file without language is the reference
- Keeps order of properties keys
- Google translate add-on (account required)

3.1.5. Contribution Guidelines

End users are free to contribute through pull requests in the project repository.

3.2. Internationalisation Support for CASUS

3.2.1. Online Repository

https://github.com/wavesnetwork/gui-translation

3.2.2. Online Demonstration

More information about how to require a demonstration:

WAVES Internationalisation support for CASUS Demo

Polish translation of the GUI is ready and part of the CASUS web site

http://krakow.casus.net

3.2.3. What is in the Box?

The CASUS Learning System has at this moment about 2,400 labels and text blocks per languages. In the past CASUS focused on:

- English
- German

As part of WAVES project the following languages were added:

- Polish

We classified the languages in the following way:

- Core languages, which are maintained by the CASUS team
- Additional languages maintained with support from partners

The language specific resources are JAVA property files that consist of key and value pairs; the key is a defined label used in the internationalised templates, the value is the actual translation. Direct work on these property files is possible but not really useful for non-technical persons. The objective of this task was to experiment with a tool, providing an easy to use interface to translate such JAVA property files.

3.2.4. Contribute Guidelines

You can contribute through pull requests. You can either:

1. Improve existing translations
2. Suggest new translations

You can use the free GUI-translation toll from Martin Adler, also part of the WAVE project. The tool sources are available in an open bitbucket repository:

https://bitbucket.org/gulpipvt/i18properties

Main configuration can be done by setting the directory of the property files, access control can be achieved by using standard JAVA webapp access methods, so can also be integrated into other systems.

3.3. Working VS Player Mobile Theme for Open Labyrinth

3.3.1. Repository

https://github.com/wavesnetwork/mobileplayer

3.3.2. Online Demonstration

An online demonstration is available online at:

http://medresearch2.med.auth.gr/waves-mob

3.3.3. Description

A mobile player for OpenLabyrinth (OL) virtual scenarios. Although the OL application is a client-server application and, thus, can be run using a mobile browser, the user experience (UX) suffers in small screens.

We followed a bottom-to-up approach, designing the player not only to fit in small screens, but also to maximise the user experience, when played in a mobile device.

3.3.4. Technical Installation Instructions

1. Clone the following repository

```
git clone https://github.com/wavesproject/OL-mobile
```

2. Go to the project folder and install dependencies:

```
npm install
```

3. Launch development server, and open localhost:4200 in your browser:

```
npm start
```

Project structure

```
dist/                        compiled version
docs/                        project docs and coding guides
e2e/                         end-to-end tests
src/                         project source code
|- app/                      app components
|   |- core/                 core module (singleton services and single-use
components)
|   |- shared/               shared module  (common components, directives and
pipes)
|   |- app.component.*       app root component (shell)
|   |- app.module.ts         app root module definition
|   |- app-routing.module.ts app routes
|   +- ...                   additional modules and components
|- assets/                   app assets (images, fonts, sounds...)
|- environments/             values for various build environments
|- theme/                    app global scss variables and theme
|- translations/             translations files
|- index.html                html entry point
|- main.scss                 global style entry point
|- main.ts                   app entry point
|- polyfills.ts              polyfills needed by Angular
+- test.ts                   unit tests entry point
reports/                     test and coverage reports
proxy.conf.js                backend proxy configuration
```

Task automation is based on NPM scripts.

Tasks	**Description**
npm start	Run development server on `http://localhost:4200/`
npm run build [-- --env=prod]	Lint code and build app for production in `dist/` folder
npm test	Run unit tests via Karma in watch mode
npm run test:ci	Lint code and run unit tests once for continuous integration
npm run e2e	Run e2e tests using Protractor
npm run lint	Lint code
npm run translations:extract	Extract strings from code and templates to `src/app/translations/template.json`
npm run docs	Display project documentation

When building the application, you can specify the target environment using the additional flag `--env <name>` (do not forget to prepend -- to pass arguments to npm scripts).

The default build environment is `prod`.

3.3.5. Development Server

This project was generated with ngX-Rocket version 1.3.3

Run `npm` start for a dev server. Navigate to `http://localhost:4200/`. The app will automatically reload if you change any of the source files. You should not use ng serve directly, as it does not use the backend proxy configuration by default.

3.3.6. Code Scaffolding

Run `npm run generate -- component <name>` to generate a new component. You can also use `npm run generate -- directive|pipe|service|class|module`.

If you have installed angular-cli globally with `npm install -g @angular/cli`, you can also use the command `ng generate` directly.

3.3.7. Additional Tools

Tasks are mostly based on the angular-cli tool. Use ng help to get more help or go check out the Angular-CLI README.

3.3.8. What is in the Box?

The app template is based on HTML5, TypeScript and Sass. The translation files use the common JSON format.

3.3.9. Tools

Development, build and quality processes are based on angular-cli and NPM scripts, which includes:

- Optimized build and bundling process with Webpack
- Development server with backend proxy and live reload
- Cross-browser CSS with autoprefixer and browserslist
- Asset revisioning for better cache management
- Unit tests using Jasmine and Karma
- End-to-end tests using Protractor
- Static code analysis: TSLint, Codelyzer, Stylelint and HTMLHint
- Local knowledge base server using Hads

3.3.10. Libraries

- Angular 2
- Bootstrap 4
- Font Awesome
- RxJS
- ng-bootsrap
- ngx-translate
- Lodash

3.3.11. Contribution guidelines

End users are free to contribute through pull requests in the project repository.

3.4. New VS Pathway Review Mechanism for Learners

3.4.1. Repository

https://github.com/wavesnetwork/vispath

3.4.2. Description

The aim of this task was to enhance learners' awareness of the decisions made in an OpenLabyrinth case. For that purpose when a student who is solving a case reaches the end of the case is able to view the whole structure of the scenario in a flowchart diagram that illustrates all nodes of the scenario and the selected options (visited nodes). By that the learner will be able to gain a clear overview of the virtual scenario, reflect on the total available options or required decisions and on the choices made. This prototype implementation is based on the OpenLabyrinth 3.1 system.

3.4.3. Technical installation instructions

To extend OpenLabyrinth 3.1 by including the pathway review mechanism overwrite the Renderlabyrinth.php and basic.php found in the path:

```
../application/classes/controller/renderlabyrinth.php
```

and

```
 ../application/views/labyrinth/skin/basic/basic.php
```

respectively with the files placed in Github:

https://github.com/wavesnetwork/vispath

3.5. Heuristic Evaluation of Open Labyrinth 3 Usability

3.5.1. Repository

https://github.com/wavesnetwork/heuristic-evaluation

3.5.2. Description

This report provides a detailed analysis of the heuristic evaluation process used to evaluate Open Labyrinth version 3.3 (OL). The evaluation itself was performed using the heuristic evaluation usability method, based on heuristics provided by Jakob Nielsen[1]. This method consists of evaluators comparing a pre-defined set of usability principles to an application or website while attempting to complete a system task.

For this project, nine heuristics were used, focusing on the core functionalities of OL: displaying, creating and playing virtual patients (VPs). The goal of this evaluation was to identify major usability flaws within the OL interface through the application Nielsen's heuristics.

3.5.3. Contribution Guidelines

Users can contribute through pull requests. You can either:

1. Improve existing usability evaluation
2. Expand the evaluation in newer versions of OL

3.6. Prototype of guided tour mechanism for Open Labyrinth 3

3.6.1. Repository

https://github.com/wavesnetwork/guidedtour

3.6.2. Description

The goal of this tool was to enrich the virtual scenario delivery systems with contextualized help/tutorial system which would make navigation for users easier. The basic idea was to avoid annoying users with wordy manuals and instead offer them clear help entries for each non-trivial UI element to be used at the moment they may need it. This prototype demonstrates how a “guided tour mechanism”can be embedded into the OL virtual scenario delivery system, particularly OpenLabyrinth 3.5.

3.6.3. Online Demonstration

An online demo is available at https://olab.wavesnetwork.eu

3.6.4. Technical Installation Instructions

1. Create database in MySQL from `database.sql`, with/without test data in INSERT INTO…

[1] Nielsen, Jakob. "Heuristic evaluation." *Usability inspection methods* 17.1 (1994): 25-62

2. Deploy database backend `editor.php` and insert your own test data

3. Add `_introjs_hook.php`, modify its header for IP filtering (or change laoding IntroJS[2] from CDN to local source) and mysql connection settings:

```
<?php
include_once("iplib.php");
$filter = new IPFilter( array(
      '.....',   // add to IP adress to filter
));

if ($filter->check($_SERVER["REMOTE_ADDR"]) === true):
..

// --- connection to dedicated
$config["db_server"] = "localhost";
$config["db_name"] = "adddbname";
$config["db_user"] = "adddbuser";
```

4. Modify Labyrinth ver.3 source codes

```
home.php, ../views/labyrinth/skin/*.php -- add:

<?php
require_once('_introjs_hook.php');
?>
```

5. For exact jquery style DOM selecting/matching is important to add more ids or classes into skins -- `../views/labyrinth/skin/*.php`, for example id="iba_node_title":

```
<h4 id="iba_node_title"><?php echo Arr::get($templateData, 'node_title'); ?></h4>
```

3.7. Summary of Recommendations and Improvements of Accessibility

For the optimization of CASUS for screen readers, required for a project, we used the tool “WAVE” available at wave.webaim.org. The webaim.org website also includes other useful resources for optimizing own websites/web applications. The WAVE tools are entirely online, so no installation is needed.

While the tool “WAVE” is very easy to use for static websites, using it for a dynamic webapp required some tricks which we would like to share in this section.

Once you go to wave.webaim.org the only available interface element is a input field for a URL which you have to fill with the page you would like to analyze and press return. webaim.org fetches the typed in page and will give you the results:

[2] Intro.js, http://introjs.com.

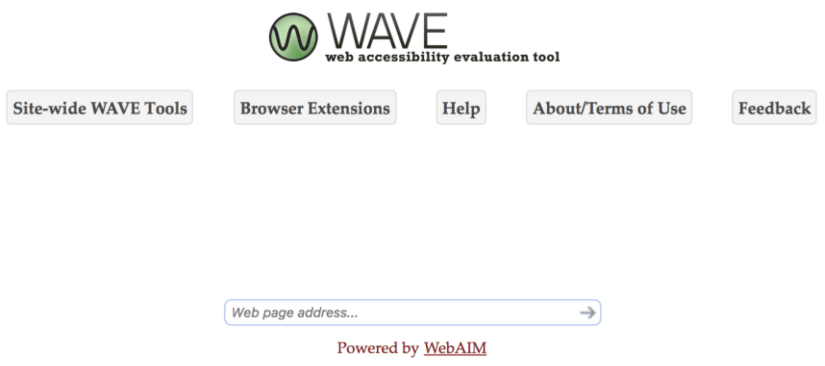
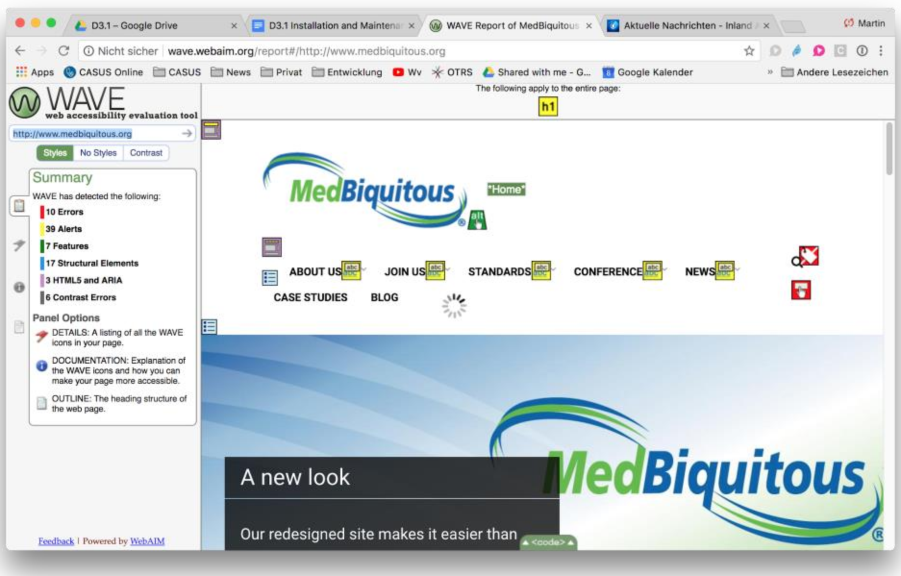


Image: Start screen of waves.webaim.org check application



The following apply to the entire page:

WAVE has detected the following:

- 10 Errors
- 39 Alerts
- 7 Features
- 17 Structural Elements
- 3 HTML5 and ARIA
- 6 Contrast Errors

DETAILS: A listing of all the WAVE icons in your page.

DOCUMENTATION: Explanation of the WAVE icons and how you can make your page more accessible.

OUTLINE: The heading structure of the web page.

A new look

Our redesigned site makes it easier than

Image: wave.webaim.org: sample result screen

The summary contains number of “errors”, “alerts”, “features, “structural elements”, “html5 and ARIA” and “contrast errors”

Errors: critical structural issues, which prevent screen readers from being able to read the content in a meaningful way, examples: missing labels for input fields. to get more than the summary you can go to the “details” section in the left menu, for each issue you can even get more explanations and hints how to fix these with “i” icon next to the issue category.

Alerts: even though not critical for reading the page, improvements would help the screen readers to process the page better, examples are broken headline structures, not meaningful links names like “read more” without more context.

Features: same as alerts but even less critical, like alternative texts of images

Structural Elements: scan for headlines, lists, etc. which help screen readers

Html5 and ARIA: checks whether additional context tags like <header>, <nav> are present in the scanned document

Contrast Errors: These refer to the additional “contrast” section of WAVE, which does address not the screen reader opportunities, but whether the page has issues for users with low vision, but not severe enough to need screen readers

The tips and tricks linked within WAVE are highly recommended and give developers basic concepts for their work.

For the CASUS analysis the goal was to have no errors at all + limiting alerts to an uncritical minimum. The changes were addressed by template and/or CSS changes, no backend programming was needed.

As written before WAVE reads the typed in URL an scans the page on their system and present results. To analyse deeper dynamic pages after login we used a trick:

CASUS is a web application installed on Apache Tomcat webapp container: Tomcat transparantely allows Http session coded with Cookies or URLs, the URL coding of the http session is

<protocol>://<host>;<session_cookie_name>=<session_id>/<path_and_page>

session_cookie_name := usually “jsessionid” unless differently configured in the application container

session_id := is the same as the cookie!

So we had 2 browsers windows open in, one we used CASUS as we would use it normally (logged in triggered navigation events) before we copy and pasted the actual URL into the WAVE tool, we grabbed the cookie with common web developer tools, like the built in tools in Firefox, copied the value of the present “JSESSIONID” Cookie (Standard http session cookie name in Tomcat unless differently configured) and so constructed the URL encoded tracking with adding e.g. ;jsessionid=FA22C9A397876F716B1A909A9F42C8BF to the plain url:

3.7.1. Example for the CASUS case selection page:

Original URL: https://player.casus.net/pmw2/app/player2/player_selection2.html

URL encoded http session:
https://player.casus.net;jsessionid=FA22C9A397876F716B1A909A9F42C8BF/pmw2/app/player2/player_selection2.html

With this technique we were able to address issues in all relevant pages for the course participant player within CASUS. If the web app container or framework does not support such a transparent way for coding http session data in cookies, you might need to add similar functionality work arounds in you app.

WAVE is a very powerful tool, we highly recommend checking your pages constantly, you very soon get a sense on important things to work on for helping users with low or no visuals. Most of the changes were fairly easy and not hard to fix. Nevertheless we also recommend additionally to WAVE trying out screen readers to get a sense how you application feels for such users. On Mac a basic functionality is already built in, another tool is the NVDA screen reader available on a donation basis on https://www.nvaccess.org/ for Windows. The standard is commercial JAWS, pricy though: https://www.freedomscientific.com/Products/Blindness/JAWS

3.8. Challenges of Export/Import using the MedBiquitous Virtual Patient ANSI/MEDBIQ VP.10.1-2010 Standard

This text is also available online in the GitHub repository at https://github.com/wavesnetwork/mvp-standard and will be regularly updated as new questions emerge. These questions and their answers may also be contributed by the wider community.

3.8.1. What is the MedBiquitous Virtual Patient ANSI/MEDBIQ VP.10.1-2010 Standard?

The MedBiquitous Virtual Patient ANSI/MEDBIQ VP.10.1-2010 standard was developed by MedBiquitous, a consortium that develops health professions technology standards. The Virtual Patient (VP) standard defines an XML schema for describing VP resources which allows VP resources to be exported and imported into standard-compliant systems. The standard builds upon the existing Healthcare Learning Object Metadata and SCORM 2004 4th Edition standards.

Effective VPs are widely recognised to be difficult to create, so the goal of the standard is to enable the exchange of these resources between individuals and systems, allowing wider access to examples of these resources, and to allow educators to share and repurpose resources for their learners. The standard (sometimes known as MedBiq VP or even MVP) was used by the EC-funded eViP program to facilitate the development of a shared bank of 340 VPs, and that project contributed to the development of the standard.

3.8.2. Why do the Virtual Scenarios in the WAVES project use a Standard called Virtual Patients?

Simply because, in most cases, there is no difference between the two! A Virtual Patient (VP) is simply a Virtual Scenario (VS) that has a setting in a healthcare context. Most VS can be represented using the VP standard.

The primary reason why the standard is designed for VPs is that those who developed it were working in medicine and healthcare education, so they chose to use the word "patient". However, it soon became apparent that this technology can be used for more than just healthcare. There are some healthcare-specific elements in the VP standard, but they can simply be omitted when a scenario does not need to use them.

3.8.3. What VS Systems Support the Standard?

There is no definitive list of the systems that support the standard. However, the systems used in the WAVES project, Casus and OpenLabyrinth/OLab, comply with the standard.

3.8.4. What can the Standard do?

The standard allows standards-compliant VS systems to export the cases as content packages in the form of a zip file that is structured so as to adhere to the standardised specification. This means that another standard-compliant VS system can read and import the file and will know where it can retrieve the relevant information from the content package.

As a result, the standard means that you can take a case from one system, and move it to another system. These systems can be different installations of the same system (i.e. if you are moving a VS from one installation of OpenLabyrinth to another) or different systems (i.e. moving a VS for

OpenLabyrinth to Casus). You could even use the standard to export and import a VS back into the original system, thus creating a duplicate case that you can edit.

3.8.5. What Aspects of a VS does the standard Include?

The structure of the standard, and how it describes a VS in technical detail is beyond the scope of this document. The files and schema which make up the standard can be found in full on the MedBiq VP GitHub page. The exact elements of a VS supported will vary from system to system, so you should consult the documentation for your system.

However, in general the standard can include the text-based content and media files, any decision-making or branched options, and any metadata. It may include any MCQs or questions, and mechanisms for scoring or updating values such as blood pressure or heart rate throughout the VS. If your case is created using a visual editor, the standard may also include some details of the layout of your case map.

3.8.6. Where can I find examples of the standard?

You can find examples of VS cases to play on the web on the WAVES website, many of which will also include downloadable content packages that use the standard. Similarly, there are a lot of VS cases freely available to download from the eViP programme website. All these VS are made available under Creative Commons licences that allow them to be freely used and shared subject to the specific conditions of that licence.

3.8.7. How can I use the standard to play a Virtual Scenario?

You will need to have access to a VS system, such as OpenLabyrinth or Casus. In most systems, when you are logged into them with "Author" permissions you will find an option in the system interface to import a scenario using the "MedBiq VP" standard. Select the content that you wish to import, and follow the instructions provided by the system. An imported case can then be played in the same way as any other case.

Please note that you cannot play a case directly from the standardised file.

3.8.8. What can I expect when I import a VS?

When you import a VS case into a system, you should not expect it to be fully playable and completely organised in the new system immediately. Some of the system-specific features of a VS case may have been lost as a result of using the standard, and different systems may interpret the standard slightly differently. This is true of all technology standards; imagine the situation when you open a web page in different browsers and they each work slightly differently. Although the web page is standardised (using the HTML standard) each of Chrome, Firefox, Safari and Internet Explorer are required to interpret the information, and will likely do so differently. The standard does however ensure that the fundamentals of the information provided remain intact and readable.

When you first import a VS you should try to play it, first to see if it works at all, and then to identify if there are any modifications that need to be made in order for it to make sense in its new setting. These modifications can be made using the authoring tools of the system to which the case was imported, and may include relinking some media files, resetting the starting point of a case, or changing aspects of the scenario to account for any features that are no longer present in the new system.

The number of changes needed post-import will depend upon the similarity of the importing and exporting system. If the two systems are the same, then you can expect to have relatively few changes required. However, if the two systems are very different in nature, there may be more work involved in adapting the scenario in the new system so that it is logical and coherent to the learner.

4. APPENDIX

OpenLabyrinth Installation Guide and FAQ

Table of Contents

Version 2
Introduction to this document 3
Frequently Asked Questions 4
- What will the installation guide cover? 4
- What is OpenLabyrinth about? 4
- What license governs OpenLabyrinth? 4
- What's the latest version of OpenLabyrinth? 4
- Will there be a new version of OpenLabyrinth? 4
- What do I need to deploy OpenLabyrinth in my institution? 4
- Did you develop OpenLabyrinth from the ground up? 4
- Can I use a third-party login system in OpenLabyrinth? 4
- Can I get custom reports? 4
- Where can I get support? 5
- Troubleshooting: After installation I try to login but I get 404 error 5
- Troubleshooting: After installation, I get blank pages 5

Step-by-step Installation Guidelines 6
- Before installation 6
- Copy OpenLabyrinth to your server 6
  - Installation using Git 6
  - Manual Installation 7
- Configure your web server and grant file permissions 7
- Run the in-browser installation sequence 8

Appendices 10
- Technical Settings: Pre-Installation Check 10
- Technical Settings: File Permissions 11

Version

Version Number	Date	Author	Details
1.0	Dec 2015	Luke Woodham & Lazoros Ioannides	Initial version for ePBLNet Training
1.1	July 2018	Luke Woodham	Updated version for WAVES D3.1

Introduction to this document

OpenLabyrinth is a Virtual Patient/Virtual Scenario system specifically designed to deliver interactive learning activities with branching choices. The system is web-based and open-source, so in order to run the system you will need to install the system onto a web server. This can be challenging for those without a background in installing and maintaining such servers.

The aim of this document is to provide detailed instructions for those who wish to install and run an installation of OpenLabyrinth. The instructions are designed to be as easy to follow as possible, but some technical expertise and knowledge of servers is assumed.

The instructions are based in part upon the installation instructions provided by the OpenLabyrinth Development Consortium at https://github.com/olab/Open-Labyrinth/wiki/Installing-Open-Labyrinth.

Frequently Asked Questions

What will the installation guide cover?

The intended outcomes of the guide are:

- To understand the OpenLabyrinth system architecture, and the prerequisites for installation
- To know how to install and maintain an instance of OpenLabyrinth

The guide will NOT cover the specifics of a setting up or maintaining a network or a networked machine.

What is OpenLabyrinth about?

OpenLabyrinth is an open-source platform for creating and playing virtual patients.

What license governs OpenLabyrinth?

It's a GNU-GPL3 license - in short, we reserve:

- your freedom to use the software for any purpose,
- your freedom to change the software to suit your needs,
- your freedom to share the software with your friends and neighbors, and
- your freedom to share the changes you make.

What's the latest version of OpenLabyrinth?

As of April 2014, it is 3.3 stable, with further additions coming from develop and release-candidate versions. A version 3.4 is available, and this includes some features that may not be fully developed.

Will there be a new version of OpenLabyrinth?

Apart from maintenance fixes there will be no more development on OpenLabyrinth v3. All development is focused on the new version OLab4 https://github.com/olab/OLab4 which uses an entirely new codebase and has been completely rewritten.

What do I need to deploy OpenLabyrinth in my institution?

You will need a web server that can run PHP scripts and a compatible relational database. We mostly use Apache-PHP-MySQL, so this is the safest and fastest way to go.

Did you develop OpenLabyrinth from the ground up?

Yes and no. OpenLabyrinth v3.x was developed from scratch (v2.x and older were developed for ASP) and it is built on the Kohana MVC framework.

Can I use a third-party login system in OpenLabyrinth?

OpenLabyrinth supports the use of OAuth, an open-source, open-standard authentication protocol. We also plan to support WebID and Shibboleth in a later version.

Can I get custom reports?

You can get custom reports by using the semantic features and the SPARQL endpoint.

Where can I get support?

If you have questions about joining the software development group or wish to contribute to case development, we would happy to hear from you. Contact us by email: info@openlabyrinth.ca. If you have technical issues, leave a bug report / feature request at https://github.com/olab/Open-Labyrinth/issues

Troubleshooting: After installation I try to login but I get 404 error

You need to allow overriding apache settings in OpenLabyrinth folder: see here: https://github.com/olab/Open-Labyrinth/wiki/After-installation-I-try-to-login-but-I-get-404-error

Troubleshooting: After installation, I get blank pages

Check your Apache logs and the OpenLabyrinth logs (application/logs). Especially if there are no OpenLabyrinth logs, it is probable that the appropriate folders are not writeable.

Step-by-step Installation Guidelines

Installing OpenLabyrinth requires you to do a number of key steps.

1. Before installation, set up your server
2. Download the OpenLabyrinth files to your server and put them in the correct folder for your web browser
3. Configure your web server and grant file permissions
4. Run the in-browser installation sequence

Please note that some of the commands, and the location on your server that the files should be placed will vary depending upon your setup. For the purposes of this guide we will assume that you are using an Apache web server, with the document root (the folder in which your web pages must be placed) being at /var/www.

Before installation

To install OpenLabyrinth you need to provision a server on your network that satisfies all the requirements in the pre-installation checklist detailed in the next section. It is beyond the scope of this guide to describe the steps needed to do this, but there are many resources available on the web.

OpenLabyrinth can be installed upon machines running different web servers (Apache, IIS) or operating systems (Microsoft Windows, Mac OS X, Linux). It can also be run using different database engines (MySQL, Microsoft SQL Server, SQLite). However, for ease of maintenance it is recommended that a LAMP (Linux, Apache, MySQL, PHP) stack is used.

Copy OpenLabyrinth to your server

OpenLabyrinth is developed and hosted using the Git version control system, and is hosted as an open source project on the GitHub website. There are two available methods of installation: manual installation, or using the functionality of Git to retrieve the latest version hosted on GitHub. Your preferred method of installation will depend upon what level of access to the server you have.

If you have full root administrative access to the server, or if you access the server only through a terminal window over SSH, it is recommended to install OpenLabyrinth using Git.

If you have limited access to the server (i.e. you do not have the right to install packages or have only permissions to the directories accessible to the web server), or if you primarily access the server using an FTP client, it is recommended to use a manual installation.

Installation using Git

1. Access the terminal on the server.

Switch to root user using the following command:

```
sudo -su
```

(This command may vary depending upon the distro of Linux used. The above command is used by Ubuntu and its variants.)

2. If you already have Git client on your server, then skip this step. Install Git client:

```
apt-get install git-core
```

3. Go to "www" folder:

```
cd /var/www/
```

4. Clone OpenLabyrinth from GitHub

```
git clone https://github.com/olab/Open-Labyrinth.git
```

This will copy the OpenLabyrinth files to your current folder on the web server.

Manual Installation

If you do not have Git installed on your server and do not have permissions to do so, or if you have only FTP access to your web servers directories, you can instead download OpenLabyrinth as a zip file from the website, and copy it to the server using FTP.

The latest version of OpenLabyrinth can be obtained as a Zip file from https://github.com/olab/Open-Labyrinth/releases.

1. Download the zip file from the site.
2. Unzip the package.
3. Use FTP to copy the package across to the correct directory on the server (i.e. the directory used by your web server).

Configure your web server and grant file permissions

Please note that many of these steps will require you to have root administration permissions on the server. If you do not have these credentials, you will need to seek assistance from members of your network administration team. Again, for the purposes of these instructions we are assuming that you are using an Apache server with the default folders and settings.

5. The next step requires you to configure your Apache hosts to make sure that the system can correctly identify your OpenLabyrinth homepage when people come to

your site. This involves editing one of the Apache configuration files. The name of your text editor may vary, but the command to do so in Ubuntu is as follows:

```
nano /etc/apache2/sites-available/000-default.conf
```

This will open a text file called 000-default.conf. You should edit the line which specifies DocumentRoot to read

```
DocumentRoot /var/www/Open-Labyrinth/www
```

Save the file and using the terminal, restart Apache

```
service apache2 restart
```

6. You must then make sure that the OpenLabyrinth system can write to its own folders, so that it can correctly install, make changes to files and upload media. First we need to designate Apache as the “owner” of the folders, by running the following command

```
sudo chown -R apache:apache /var/www/Open-Labyrinth/
```

Please note that web server user and group can change depending upon your configuration. It may be 'www-data:www-data' instead of 'apache:apache' or something similar, so you should consult the documentation for your web server.

We then need to set the permissions on the folder by running the command

```
sudo chmod -R 0775 /var/www/Open-Labyrinth/
```

This means that all users can execute and read the files on that location, but only the apache process (i.e. your web server) can write to the folder.

Run the in-browser installation sequence

7. Open OpenLabyrinth in the browser. If your browser is running on the server rather than your own PC, this will be http://localhost. If it is a different machine, then you will need to type the URL or IP address for your server that should have been provisioned in setting up the server before installing OpenLabyrinth.
8. This will begin an installation process, displayed in the browser. Complete the installation following the on-screen prompts (you will need MySQL database credentials, which will have been created when you installed MySQL. If you do not know these, please speak with your network administrators.)

The interface will show whether or not your system meets the specified prerequisites to run Openlabyrinth. If your system does not meet these requirements, steps will need to be taken in order to ensure that it does. Issues will mostly relate to installing or enabling packages and components on the server.

If the system indicates that you need to enable mod_rewrite, run the following command from the terminal

```
a2enmod rewrite
```

You will then need to restart apache

```
service apache2 restart
```

The system installation will run through the necessary tasks to set up a database and install OpenLabyrinth. You will be asked to populate the username and password for the MySQL admin user.

If you have some issues with files permissions during installation, then read this page:
- https://github.com/olab/Open-Labyrinth/wiki/How-do-I-make-files-and-folders-writable-for-the-web-server%3F

If you have 404 error after installation, then read this page:
- https://github.com/olab/Open-Labyrinth/wiki/After-installation-I-try-to-login-but-I-get-404-error

After installation has completed, you will be redirected to the home page (login page). Here you can use the credentials you created during the installation. Having logged in you can go to "Import" module and import all labyrinths that you want.

Appendices

Technical Settings: Pre-Installation Check

Requirement	WHY?	TO DO
Apache 2.2+ or IIS	kohana 3.2 requirement	Install a compliant Web server
PHP 5.3.3 or newer	kohana 3.2 requirement	Install PHP 5.3.3
MySQL 5.1+ or Microsoft SQL Server or SQLITE 3	kohana 3.2 requirement	Install a compliant database server
Mod-rewrite	To enable kohana's friendly URLs	Install mod-rewrite for apache and enable it
System Directory	The kohana system files are in-place	Make sure you copied or cloned the repository correctly
Application Directory	The open labyrinth application is in-place	Make sure you correctly copied or cloned the repository
PCRE UTF-8	Utf-8 regular expressions matching (export/import, semantic features, inline tags replacement)	(should be available by default – re-check your PHP version)
SPL Enabled	PHP class auto-loading for kohana	(should be available by default – re-check your PHP version)
Reflection Enabled	Kohana requirement	(should be available by default – re-check your PHP version)
Filters Enabled	External data (user input) sanitizer and validator	(should be available by default – re-check your PHP version)
Iconv Extension Loaded	Character set conversion (needed for import/export	Should be enabled by default, check your php.ini
Mbstring Not Overloaded	Kohana requirement (multibyte strings)	If you need mbstring overloading for another site, please disable it, locally, in OpenLabyrinth's .htaccess
Character Type (CTYPE) Extension	Character type checks required by kohana	(should be available by default – re-check your PHP version)
URI Determination	Tests if kohana correctly recognizes paths with subdirectories	(should be available by default – re-check your PHP version)

Technical Settings: File Permissions

File / folder	How long?	Why?
/olab/www/application/bootstrap.php	Only for installation	Holds configuration for the kohana application
/olab/www/install.php	Only for installation	It gets deleted afterwards, for security and integrity reasons
/olab/www/installation	Only for installation	It gets deleted afterwards, for security and integrity reasons
/olab/www/application/cache	All time	It is used for caching
/olab/www/application/logs	All time	It is used for logging
/olab/www/application/config	Only for installation	Holds configuration files for the kohana application
/olab/www/avatars	All time	Avatars are stored here
/olab/www/css/skin	All time	Skins are stored here
/olab/www/files	All time	Uploaded files are stored here
/olab/www/scripts/fileupload	All time	Uploaded files are stored here
/olab/www/tmp	All time	Temporary files are stored here
/olab/www/updates	Only for installation / updates	Database Updates are stored here and get deleted afterwards