Contract Number **619347**	Document Title **AUTOMATIC ITEM GENERATOR, Final Version**	Deliverable Nature **P** Dissemination level **PU** Version **1.0**

EAGLE

EnhAnced Government LEarning

www.fp7-eagle.eu

FP7-ICT-2013-11

Objective 8.2 Technology-enhanced learning;

Target outcome c): Holistic learning solutions for managing, reaching and engaging learners in the public administrations

Deliverable 6.11

Automatic Item Generator (Final version)

WP 6 – OER SERVICES – OER Public Administration Services
Lead Participant: LIST

Approval Panel	Name / Partner short name	Department / Function	Date
Author	Eric Tobias – LIST Olivier Pedretti – LIST Eric Ras – LIST Muriel Foulonneau – LIST	Engineer Engineer Group Leader R/T Associate	13/07/16
Reviewer	Sabine Möbs – DHBW	Professor	27/04/16

This project has received funding from the European Union's Seventh Framework Programme for research, technological development and demonstration under grant agreement N°310806.

Table of Content

List of Figures

Software Title	EAGLE Automatic Item Generator
Version	3 (software version number 0.8.0)
Release Date	2016-07-30
License	Apache 2.0
Languages	PHP 5.6 , Java (EE) 1.8

1 Introduction

EAGLE's main objective is to equip employees in rural local government with a holistic training solution based on Open Educational Resources (OER) and Open Source (OS) tools, supporting the development of critical transversal EAGLE competences such as Information literacy (IL), ICT/Digital literacy (DL), and Change management (CM).

Work package 6 (WP6) for which this deliverable has been prepared, aims to develop specific services and tools support for OER usage.

This deliverable describes the recent work to develop the automatic item generator (AIG) v3.0. In this deliverable we present the overall AIG component and some of the newer components developed during the final iteration. We also describe the use of the component from a user perspective. We will put some focus on the integration with the EAGLE platform to show the intricacies of the connection between the existing and developed tools and the final product.

2 Overview of the prototype

D6.7 updated D6.3 and describes the second iteration of the Automatic Item Generation (AIG) component. D6.11 is based on D6.7 and present the final version the AIG as an evolution from its base (D6.3) and 2nd release (D6.7). It implements the model, specifically the template model fully, to support a greater number of items and, thus, extends its scope beyond the initial set of items.

In D6.7 components that were developed for assessing the quality of the generated items and providing valuable metadata were loosely integrated into the AIG process and have been readied for full integration for this final iteration. For the final version of AIG, several extensions in TAO have been developed or improved upon to support the delivery of tests and the general EAGLE workflow as defined by the platform.

The overall AIG component will generate tests from a composition of assessment items. During the final iteration, the test generation component was improved, using the template-based item generation as defined in D4.5 to its fullest and its final version is now properly integrated with TAO and subsequently into the EAGLE platform.

3 The Automatic Item Generator at a Glance

3.1 Problem(s) addressed by the Automatic Item Generator

The attitude of learners is a critical component. The EAGLE platform supports users to make a successful learning experience. 21st Century skills, including digital literacy are very important for the appropriation of the platform content and features by learners. Information literacy skills help to find, understand, create and use the information of the platform in a daily working context. We claim that formative assessment is an efficient means to support learning of information literacy. Nevertheless, it requires that a learner is continuously assessed and that tests are made available. Automatic item generation helps to solve this challenge by generating tests from different resources automatically.

3.2 Solution proposed by the Automatic Item Generator

In order to support platform users in making the EAGLE experience an active experience, the EAGLE platform will provide a self-assessment component where a learner can take short tests to improve the knowledge and skills. Such a test is considered an OER in the EAGLE platform. First, the EAGLE platform will allow users to manually create tests. Second, it will also provide a mechanism to generate tests to support users in providing OER to the platform. The generated tests will focus on information literacy skills. Nevertheless, EAGLE platform users will be able to manually create tests on any learning topic which is of relevance. While users are browsing the EAGLE platform, the assessment functionalities will support the learning process.

Authoring assessment items and tests is a complex process that requires thinking about the learning outcomes and relevant competence models. In addition, the creation of a test is an effort in itself. Typically, employees in the public administration lack competences to author assessment resources. Therefore, the assessment generation component aims to support these users in creating assessment resources through the generation of tests which can be manually revised by the users. In order to prevent certain quality loopholes and biases in a test, metrics on the linguistic and semantic content of the assessment items are provided, which ease the selection of the best items by the unexperienced test author.

3.3 Innovative aspects of the Automatic Item Generator

There is no other multilingual automatic item generation mechanism. While automatic item generation has been mainly used for generating assessment items for mathematics and language learning skills, information literacy skills have not been considered yet in the context of AIG. Today, there is no other solution that provides metrics on the linguistic and semantic content of test items.

3.4 Prerequisites to use the Automatic Item Generator

The Automatic Item Generation component works optimally with a well formatted domain model in RDFS/OWL. Textual resources should be formatted in a way that semantic domain models can be extracted as a basis for the AIG process. In addition, by following provided authoring guidelines, semantic relevant information can be easily extracted from the resources.

3.5 Integration of the Automatic Item Generator into the EAGLE platform

The Automatic Item Generation component is integrated into the EAGLE platform through a portlet that gives access to the assessment functionalities provided by the TAO software component. Tests are presented on the platform thanks to identifiers and usage information exposed through a TAO related Web service.

4 The Automatic Item Generator from a User Perspective

In EAGLE five personas have been developed as well an overall scenario. This section describes first more abstract user stories and second instantiates than one of these user stories by using one specific persona. This scenario describes the use of the AIG for a specific work example.

4.1 User stories

User stories have been designed for the main components of the EAGLE platform to facilitate the technical review of the different software deliverables in WP5 and WP6.

All user stories have been defined in a standardised way to ease their understanding. Each user story starts with a type of user, a well-defined task, and the effect or impact it will bring.

Furthermore, feedback can be given by naming the corresponding user story. The following template was used.

As <a type of user>, I want to <task to be performed with EAGLE>, so that <>.

The following user stories have been implemented in the first and second prototypes of the EAGLE argumentation tool:

AIG users story 1:

As user, I want to automatically generate a test related to an OER I just consumed, so that I can assess my knowledge and skills by taking the generated test, or that I can save time to create a test related to an OER I created before and which I can share with other users.

AIG users story 2:

As user, I want to manually create an OER of the type test, so that I can reflect about what I just learned or know, and that I can share this test with other who want to assess their knowledge and skills.

AIG users story 3:

As a user, I want to take a test related to a specific OER, so that I can assess whether I learned from the OER I just consumed.

4.2 Example for user story: Persona – Louisa

Being an experienced ICT user, Louisa was able to familiarise herself quickly with the more basic OER authoring formats, such as text. Since her personal interest is also to help colleagues with using ICT, Louisa has already authored several Wiki pages, which describe how to use specific ICT tools in municipalities. While doing so, she followed consciously the authoring guidelines provided on the EAGLE platform for Wiki page authoring. She decided to provide a glossary of common features of the device for printing ID cards and terms uses in the formal process of issuing ID cards. In the past, her help desk, for which she is

responsible, was called often to solve issues about print ID cards and questions about the underlying process.

After consuming an available tutorial about the automatic item generation component, she felt ready to generate a test, which was meant to be published as an additional OER. The goal of this OER was to enable her colleagues to assess their knowledge with regard to issuing a new ID card.

5 System Architecture

While for any user the system design as separate independent parts working in unison will be transparent, it is important to stress both views on the system. On one hand the end user will see the EAGLE portal with all of its moving pieces as one webpage he uses with one account. On the other hand the service providers that are thought to host the portal need to be aware of all the pieces that they might need to install, deploy, and maintain. In this section we will provide the latter view in different levels of abstraction to present to the reader the overall design of the system with a focus on the AIG sub-system.

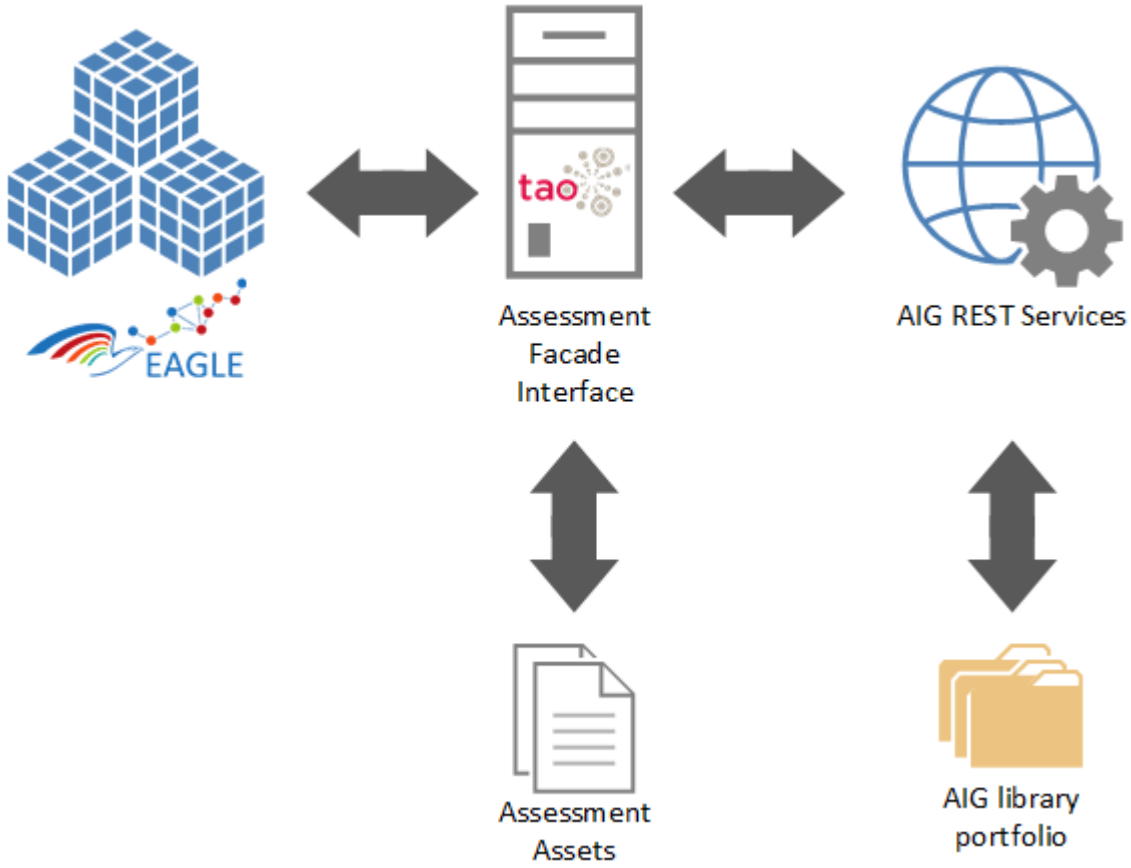

FIGURE 1: A TOP-LEVEL VIEW OF THE MAJOR COMPONENTS WITH A FOCUS ON AIG

The EAGLE portal will have one access point to the AIG component, that is, an embedded instance of a customized TAO. The instance will run in an iFrame and usable from within the portal. All TAO features will be available to users but navigation to the instance might redirect to specific views of the interface to provide the users the feature s/he wants to access without the need to navigate the TAO instance by themselves.

Behind the customized TAO instance are a couple of newly developed and customized modules. The most important module is the Assessment Façade Interface (also called the Façade or, more technically, the taoAigFacade). It provides functionality to the customized inbuilt modules and provides I/O to other services such as the AIG library. In addition it exposes some core TAO functionality as a REST interface.

Other TAO modules modify TAO and provide, for example, test generation facilities (Test Generation module) or metrics on items (Item Feature Analysis module). A top-level view on the different TAO modules is shown in Figure 2.

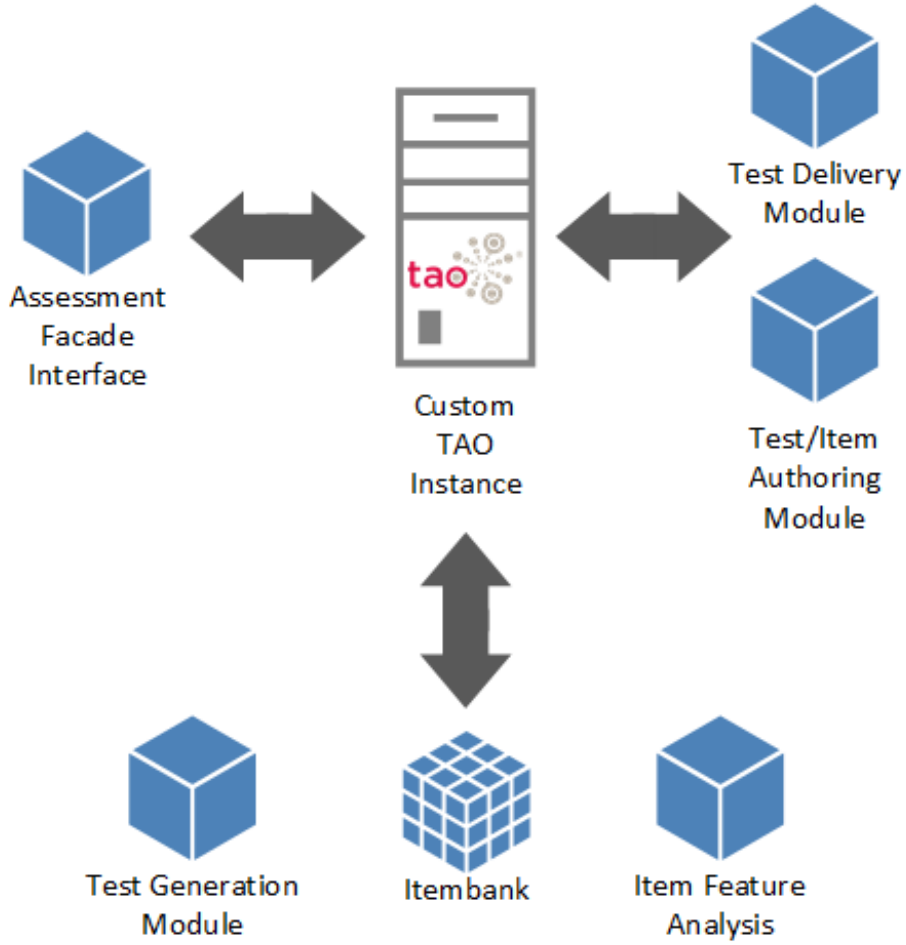

FIGURE 2: A TOP-LEVEL VIEW ON THE CUSTOM TAO INSTANCE AND ITS MODULES

As shown in Figure 1, the Façade does not only provide access to assessment assets such as tests or items, it also communicates with the AIG REST Services (also known as SIREN) in order to launch automated item generation or information extraction services provided by the Automatic Item Generation library.

Figure 3 shows a top-level view the AIG ecosystem. SIREN provides a front-end to several core AIG services and is used as a single point of entry using REST to request any resource processing or item generation.

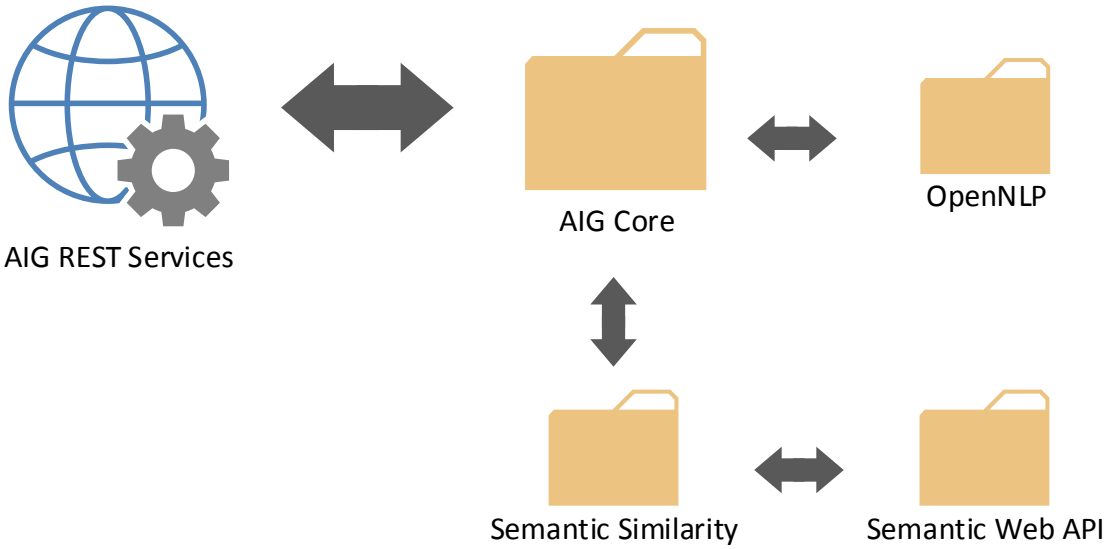

FIGURE 3: TOP-LEVEL VIEW OF THE AIG ECOSYSTEM

The current version features a reworked and more stable version of the Semantic Similarity library used to generate metadata on items and their contents as well as an implementation of the templates as described in D4.5. In order to process resources that have been generated natural language, the AIG Core uses Apache's OpenNLP library[1].

SIREN, while an integral part of the ecosystem, has been rendered transparent to the EAGLE platform which will only converse with the Façade (see Figure 1). While the services provided by SIREN remain largely unchanged and can still be accessed directly, it is not recommended that any adopter do so. As resources need to be synchronized between the item bank (TAO) and the EAGLE platform, it is advised that the server configuration where the AIG ecosystem is installed reject any calls from third parties and only allows calls from the Façade.

[1] https://opennlp.apache.org.

6 Accessibility

With the Automatic Item Generator functioning in the backend, its accessibility concerns lie with the content it produces; with test items and tests. Items are specified using the Question and Test Interoperability (QTI) notation. They are then stored in the item back and delivered as tests through TAO.

The Automatic Item Generator itself does not inject any accessibility elements as defined by IMS' Accessible Portable Item Protocol (APIP)[2]. However, the generator can easily be adapted to support APIP as the elements specific to the accessibility can be held by the template and resolved normally during the generation process.

TAO, in version 3 as used for this project, supports the QTI standard in version 2.1 which includes accessibility as defined by APIP[3]. Furthermore, TAO, as mentioned on their feature page[4] supports delivering tests to mobile devices.

[2] https://www.imsglobal.org/apip/index.html.
[3] http://www.imsglobal.org/question/qtiv2p1/imsqti_infov2p1.html#section10104.
[4] http://www.taotesting.com/get-tao/features.

7 Source Code

Sources are readily found on the EAGLE Git hosted by LIST**:** https://git.list.lu/groups/eagle. Most sources are readily available to consortium members which can request a Git account at any moment. The specific sources of the projects mentioned above can be found in:

- https://git.list.lu/eagle/SIREN**,**
- https://git.list.lu/eagle/AIG**,**
- https://git.list.lu/eagle/tao-for-eagle**.**

The latter contains the sources of all modified and custom-made TAO modules delivered as a complete custom TAO distribution specific for the EAGLE project ready for installation within the scope and limits detailed in this document.

Once the project is complete, all sources will be made public, that is, all users can access the resources freely. The consortium will remain in control of the resources themselves and monitor all requests for changes to guarantee all interested parties can access the same resources.

8 Installation Guide

This part of the document aims to provide all the necessary technical details for users versed in the field to install and subsequently use the AIG system and related components. The following sections detail each of the technical corner pieces of the AIG system infrastructure.

8.3 Prerequisites

8.3.1 TAO and custom modules

TAO[5] is an open source e-Testing platform developed by Open Assessment Technologies S.A.[6]. The requirements listed below are for the recommended respectively minimum versions required to run TAO (3.0) as used in the EAGLE project:

- PHP 5.6,
- Apache 2.4.7,
- MySQL 5.5.

On the client-side, TAO can be run in any modern web browser running JavaScript. Mobile devices are supported as well but the quality of the user experience depends on screen size. This in turn also holds true for displaying TAO in an iFrame which sometimes sees clipping issues. For specific client-side requirements we recommend to check the OAT requirements page[7].

8.3.2 AIG services and libraries

SIREN is packaged as a web application archive (WAR). You will need a Java Application Server to run SIREN. You can install a new webserver or reuse an existing one. We have actively tested the application on Apache Tomcat 8. The WAR itself requires Java 8.

SIREN built as a WAR contains all dependencies and does not feature any other requirement. However, users can tweak some parameters by altering some of the properties set in a properties file. The file contains documentation but we discourage that users modify these without a good grasp of the consequences.

8.4 Installation

8.4.3 TAO and custom modules

The team at OAT has compiled an installation guide for TAO on Ubuntu which we would like to recommend. It details the installation steps and specificities for Apache, PHP, and MySQL as well as their configuration. For this project, we have used Ubuntu 14.04.2 LTS (64 bit). Visit http://forge.taotesting.com/projects/tao/wiki/InstallUbuntuApacheMySQL for a complete guide.

[5] https://en.wikipedia.org/wiki/TAO_%28e-Testing_platform%29.
[6] http://www.taotesting.com.
[7] http://www.taotesting.com/get-tao/system-requirements.

The provided custom TAO package for EAGLE contains all extensions to existing modules and newly developed modules. To install these modules, it is sufficient to install the modules from the TAO Extension manager once TAO has installed successfully. For the more technical users, extensions can also be installed by an installation script. To read more we recommend http://forge.taotesting.com/projects/tao/wiki/How_to_manage_your_Extensions.

As some of the custom modules need to address SIREN, please be aware that you need to provide the modules with the location of SIREN. The *taoAigFacade* folder which contains the Façade module code contains a *constants.php* file located under *taoAigFacade/includes.* The properties *TAO_AIG_FACADE_SIREN_HOST* and *PORT* respectively will need to be modified with the correct address and port number to be able to address the deployed SIREN endpoint.

The Item Feature Analysis module needs to address several AIG REST services. The addresses of these services can be configured by modifying related constants in the *itemFeatureAnalysis/includes/constants.php* file. The constants names are suffixed by *_URL*. In a next iteration these services should be regrouped in one service.

8.4.4 AIG services and libraries

To install the AIG services it suffices to compile SIREN as a WAR, respectively, retrieve an already compiled WAR and deploy it on your Java Application Server. Once deployed, SIREN will automatically start to listen for incoming traffic and will handle it accordingly to its specification. Should any modification be made to the properties file that resides within the WAR (you can open it as you can any other archive like ZIP or RAR), you will need to redeploy SIREN. In case the modification has been made directly on the server, the application server will require to be restarted.

9 Conclusion

The integration of the different Automatic Item Generation components as well as the custom TAO instance and modules with the EAGLE platform has been completed in this deliverable. While the initial deliverable gave a rough view of the component integration, this deliverable specifies the result of the integration with more detail and also provides details on the integration of components of the final iteration.

10 License