Realising and Applied Gaming Eco-system

Research and Innovation Action
Grant agreement no.: 644187

D2.5 - First bundle of dashboard components
RAGE – WP2 – D2.5

<table>
<thead>
<tr>
<th>Project Number</th>
<th>H2020-ICT-2014-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Due Date</td>
<td>31 August 2016</td>
</tr>
<tr>
<td>Actual Date</td>
<td>31 August 2016</td>
</tr>
<tr>
<td>Document Author/s</td>
<td>e-UCM</td>
</tr>
<tr>
<td>Version</td>
<td>0.4</td>
</tr>
<tr>
<td>Dissemination level</td>
<td>PU</td>
</tr>
<tr>
<td>Status</td>
<td>Final</td>
</tr>
<tr>
<td>Document approved by</td>
<td>Wim</td>
</tr>
</tbody>
</table>
### Document Version Control

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Change Made (and if appropriate reason for change)</th>
<th>Initials of Commentator(s) or Author(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>16 August 2016</td>
<td>First version</td>
<td>MFM</td>
</tr>
<tr>
<td>0.2</td>
<td>26 August 2016</td>
<td>Internal WP2 review and minor improvements</td>
<td>IMO</td>
</tr>
<tr>
<td>0.3</td>
<td>30 August 2016</td>
<td>Complete WP2 draft</td>
<td>BFM</td>
</tr>
<tr>
<td>0.4</td>
<td>31 August 2016</td>
<td>Final WP2</td>
<td>BFM</td>
</tr>
</tbody>
</table>

### Document Change Commentator or Author

<table>
<thead>
<tr>
<th>Author</th>
<th>Name of Author</th>
<th>Institution</th>
</tr>
</thead>
<tbody>
<tr>
<td>MFM</td>
<td>Manuel Freire</td>
<td>UCM</td>
</tr>
<tr>
<td>IMO</td>
<td>Iván Martínez-Ortiz</td>
<td>UCM</td>
</tr>
<tr>
<td>BFM</td>
<td>Baltasar Fernández-Manjón</td>
<td>UCM</td>
</tr>
</tbody>
</table>

### Document Quality Control

<table>
<thead>
<tr>
<th>Version QA</th>
<th>Date</th>
<th>Comments (and if appropriate reason for change)</th>
<th>Initials of QA Person</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## TABLE OF CONTENTS

**EXECUTIVE SUMMARY** ......................................................................................................................... 5

1 **INCLUDED ASSETS** ................................................................................................................................. 6

2.1a Client Tracker (UCM) ......................................................................................................................... 6
    Status ......................................................................................................................................................... 7
    Links ......................................................................................................................................................... 7

2.1b Server-Side Interaction Storage and Analytics (UCM) ..................................................................... 8
    Current status ........................................................................................................................................... 9
    Links ......................................................................................................................................................... 9

2.4a Authentication & Authorization (UCM) ............................................................................................... 10
    Current status ......................................................................................................................................... 10
    Links ....................................................................................................................................................... 10

2.4b Server-side Dashboard and Analysis (UCM) ....................................................................................... 12
    Current status ......................................................................................................................................... 17
    Links ....................................................................................................................................................... 17

2 **CONCLUSIONS** .................................................................................................................................... 18
**TABLE OF FIGURES**

Figure 1. Tracker top level architecture overview old (left) and new (right) model. .......................... 6
Figure 2. RAGE Analytics configuration user interface. ........................................................................ 8
Figure 3. A2 asset top level architecture overview............................................................................. 10
Figure 4. Average score per player visualization. ............................................................................... 13
Figure 5. Session activity over time visualization. ................................................................................. 13
Figure 6. Activity count per player visualization. .................................................................................. 14
Figure 7. Different alternatives responses count visualization.............................................................. 14
Figure 8. xAPI verbs activity visualization. .......................................................................................... 15
Figure 9. Number of learners that have passed/failed custom visualization......................................... 15
Figure 10. Single learner custom visualization..................................................................................... 16
Figure 11. Players ranking (higher number of errors) custom visualization........................................ 16
EXECUTIVE SUMMARY

This document describes the core components used to create customizable analytics and dashboards. This document details how to configure, take advantage of the analytics visualizations and the underneath architecture of the platform. The architecture uses a powerful and highly configurable dashboard platform that simplify the creation of specific visualizations and the personalized display of the results obtained by the analysis.

This deliverable describes the main RAGE assets and the changes performed to upload custom analysis and configure visualizations. This architecture provides a number of visualizations but it is also scalable and can be extended and adapted to perform ad-hoc visualizations to cope with different game specific or project specific requirements. All these dashboard component are working with data collected using the xAPI data format that the RAGE project has developed in collaboration with ADL Co-Lab.

Included assets contain a description of their current status, and links to their full designs and downloadable versions. It should be noted, however, that various additional dashboards and further improvements are being developed continuously.

All the dashboard components take into account and include security features to assure the data analytics security guidelines defined in RAGE (including aspects of anonymization when possible) and to comply with the law about personal data storage and protection in accordance with EC data regulations.
1 INCLUDED ASSETS

2.1a Client Tracker (UCM)

This component sends analytics information to a server; or, if the server is currently unavailable, stores them locally until it becomes available again (this is a semi-connected mode). It is a central part of the RAGE analytics ecosystem.

In this deliverable the tracker implementation as described in deliverable D2.3 has been updated to suit a new xAPI model for serious games. The original model contains four ways to determine what happens in the game: Screen, for specifying the game stage, Zone, for setting where in that stage, Variable, for elements like score or progress, and Click, for showing where the player has clicked.

However, the initial model didn’t represent all the elements in a game, so it was not enough for expressing low-level concepts, like the option selected in a question, or a quest completed successfully or failed. The new specification provides more control to the teachers who want to use serious games as teaching tool, as they can know exactly where is each learner and receive alerts if the learner is going wrong way.

This new model includes all changes presented in the figure, and is specified in the Github repository (https://github.com/e-ucm/xapi-seriousgames).

This new API is presented in the figure below (see Fig. 1).

![Tracker top level architecture overview old (left) and new (right) model.](image)

This new API is presented in the figure below (see Fig. 1).

Three versions of this tracker have been implemented or updated for this deliverable, including a C# version, a Java/Libgdx version, and Unity3D version (also written in C#, but using Unity3D libraries). The C# version of the Tracker is fully compliant with the RAGE asset architecture explained in D 1.4.
D2.5 - First bundle of dashboard components

**Status**

Fully functional and distributed with two tests projects. About the test projects, on one hand, TrackerAssetTestApp is designed to manually test the TrackerAsset. On other hand, TrackerAssetUnitTests is designed to automatically test TrackerAsset functions, including the ones specified in the new Tracker API.

**Links**

- **Sources**
  - [https://github.com/e-ucm/unity-tracker](https://github.com/e-ucm/unity-tracker) (Unity version)
  - [https://github.com/e-ucm/ClientSideTrackerAsset](https://github.com/e-ucm/ClientSideTrackerAsset) (C# version)
  - [https://github.com/e-ucm/libgdx-tracker](https://github.com/e-ucm/libgdx-tracker) (Java/Libgdx version)
- **Design**
  - [https://docs.google.com/document/d/1_XtAMvl9mNhR5Qxc9RH6OWhRBjyL2lq6biAwTZegAQ/edit?usp=sharing](https://docs.google.com/document/d/1_XtAMvl9mNhR5Qxc9RH6OWhRBjyL2lq6biAwTZegAQ/edit?usp=sharing)
- **User documentation**
  - [https://github.com/e-ucm/unity-tracker/blob/master/README.md](https://github.com/e-ucm/unity-tracker/blob/master/README.md)
- **Example integration with game**
  - [https://github.com/e-ucm/QuizDemo](https://github.com/e-ucm/QuizDemo)
2.1b Server-Side Interaction Storage and Analytics (UCM)

This asset offers a ready-to-deploy server-side implementation of a data collection and storage service. It provides an API to manage games and classes, and, also, the data collected by the tracker (2.1a).

The asset described in this deliverable adds new features to the previous version, described in D2.1. For instance, it now includes the option to upload custom analysis through the API or through the front end (2.4b) (see Fig. 2). The analysis has to be created using Apache Flux (http://storm.apache.org/releases/2.0.0-SNAPSHOT/flux.html), it is compatible with Apache Storm (http://storm.apache.org/), Storm Trident (http://storm.apache.org/releases/1.0.1/Trident-tutorial.html) and other scripting languages (python, java…) described by its documentation.

![Game Example Serious Game](image)

Figure 2. RAGE Analytics configuration user interface.

The custom analysis package is a .zip package composed of the following files. A zip file with a ‘realtime.jar’ (analysis) and ‘flux.yml’ (configuration) files, for instance:

1. realtime.jar - A jar file with the analysis topology in a correct Storm & Flux format.
2. flux.yml - A configuration file for the analysis.

The ‘realtime.jar’ contains the classes (and dependencies) referenced within the ‘flux.yml’ configuration file. It may also contain the complete topology used in the analysis and launched by Flux. More information about the Storm-Flux specification can be found at: https://github.com/apache/storm/blob/master/external/flux/README.md
The data stored by the analysis can be shown via Dashboard Asset (2.4b)

**Current status**

The asset is fully functional, although full support for data anonymization / deanonymization is still under development. Also, different analysis options for different kind of games that use the new xAPI model ([https://github.com/e-ucm/xapi-seriousgames](https://github.com/e-ucm/xapi-seriousgames)) are being developed.

**Links**

- Sources [https://github.com/e-ucm/rage-analytics-backend](https://github.com/e-ucm/rage-analytics-backend)
- Design [https://docs.google.com/document/d/1imtLyl59yOv9CrPRZdSu8UaUmgS_3ZYxhQYphcmnUPE/edit?usp=sharing](https://docs.google.com/document/d/1imtLyl59yOv9CrPRZdSu8UaUmgS_3ZYxhQYphcmnUPE/edit?usp=sharing)
- User documentation (shared with 2.1b + 2.4a + 2.4b) [https://github.com/e-ucm/rage-analytics/wiki](https://github.com/e-ucm/rage-analytics/wiki)
- Launcher (launches 2.1b + 2.4a + 2.4b + 2.4.c.s) [https://github.com/e-ucm/rage-analytics](https://github.com/e-ucm/rage-analytics)
2.4a Authentication & Authorization (UCM)

This asset provides a central location where client-side assets can authenticate and locate server-side assets, including analytics. Servers can also register to locate other servers and be locatable by clients, restricting access using configurable roles (see Fig. 3). This section describes the updated version of the asset described in D2.1 summarizing the added functionalities and changes.

In this deliverable we have added a new functionality to protect the data and restrict the access to the custom dashboards by the users. For instance, when it is decided that a teacher must have access to a new data source (i.e. index Elasticsearch), a new permission is created for the Authentication & Authorization asset side (2.4a). This functionality is used to protect and restrict access to the data sources used by the visualizations.

The goal is to avoid the access to the data from a client without authorization. Only teachers that supervise a learning session should have access to non-anonymized data of learners in the learning session. For instance, when Kibana queries data from Elasticsearch, this request is being performed through this asset (A2). Before proxying the request, this asset (A2) checks that the request user has the authorization to access a given data source. These access control permissions are stored in MongoDB, which is not accessible from outside the system.

All requests to the analysis results data source are checked in Authentication & Authorization asset side (2.4a) to verify that the users that made the requests are allowed to access the results data source.

Current status

This asset is fully functional and includes test cases.

Links
D2.5 - First bundle of dashboard components

- Sources
  https://github.com/e-ucm/a2
- Release
  https://github.com/e-ucm/a2/releases/latest
- User documentation
  https://github.com/e-ucm/a2/wiki
- Complete RAGE Analytics documentation
  https://github.com/e-ucm/rage-analytics/wiki
- Design
  https://docs.google.com/document/d/1Ve6qGs30uGUYrIMsB-UrIMaaFw2VdbSKROml8_TT78/edit?usp=sharing
- Launcher (for 2.4a + minimal dependencies; see also launchers for 2.1b, 2.4b and 2.4c.s)
  https://github.com/e-ucm/rage-auth2
- REST API documentation
  http://e-ucm.github.io/a2/
2.4b Server-side Dashboard and Analysis (UCM)

This asset provides the user interface to access the analytics functionalities of the system. The stakeholders can log in using different roles to access different functionalities. The current available roles are: learner, teacher, and developer.

Kibana is a powerful highly configurable open-source dashboard platform used to display the analysis results to the stakeholders. Some of the technologies and tools analyzed for visualizations were Kibana, Angular-charts, D3, ChartJS and OpenDashboard. After studying and testing these different technologies we decided to use Kibana due to its provided features:

- It is open source with a license compatible with RAGE requirements.
- It doesn't require programming skills to perform queries in most cases, facilitating the creation of new dashboards.
- Supports different types of views (bar charts, line charts, scatter plots, histograms, pie charts and heat maps).
- The visualizations are interactive, allowing different types of filters, including "cross-filtering".
- The visualizations are updated in real time. The refresh time and the time window of the displayed data can be configured.
- It integrates seamlessly with Elasticsearch, a distributed database that provides scalable search. Elasticsearch is widely used by large companies like LinkedIn, Netflix, eBay, Wikipedia, Github, Microsoft, Mozilla, etc.
- It can be extended using plugins.
- Has a large community behind offering constant support.

The developer role must set up a game before a teacher can use it in its class. In the configuration process the teacher may provide a customized analysis process by uploading an analysis package that contains the required information used by the analysis. The analysis package will define how the analysis process will be performed and where the results will be stored. Any case some commons default analysis are provided.

The developer can configure the visualizations. The visualizations need to know from what source they will receive the result data and how it will be displayed. The visualizations configuration is stored using a JSON format passed to Kibana, the visualizations platform used by the architecture.
Default visualizations are available and they display general game information extracted from the result data. Some initial general examples that are game independent can be seen below:

This visualization allows the teacher to track which learners are performing well and which learners are having problems progressing in the game (see Fig. 4). This visualization displays the average score on the $y$-axis and each learner in the session $x$-axis (those user ID can be configured to be more significant and easy to manage for the teacher).

The session’s activity over time provides general information about when the game is being played and how many traces are being sent to the game at any given moment (see Fig. 5).
Figure 6 visualization shows the activity of each learner in a given session. This visualization shows which learner is being very active and which learners are inactive or have abandoned the learning session.

Figure 7 visualization displays information about the different alternatives and how many times a specific alternative has been selected.
D2.5 - First bundle of dashboard components

Figure 8. xAPI verbs activity visualization.

Figure 8 visualization displays the xAPI verbs activity and provides general information about the xAPI verb usage in a session.

The visualizations data can be filtered to display information in different time periods and to sub-groups of learners, to the point of tracking a single learner if necessary. The learner’s data is only accessible by the teacher that is supervising the learning session.

Customizing the analysis to a specific game, more game-related information can be considered when displaying the visualizations. For instance, considering a game where the learners have to answer simple questions about geography, it could be obtained the following visualizations:

Figure 9. Number of learners that have passed/failed custom visualization.
Figure 9 dashboard provides a classification of the learners in terms of academic goals achieved. Assuming each of the four types of questions corresponds to a topic to be evaluated, this dashboard shows the results of the learners in each topic. The classification is made fixing a minimum number of questions of each type that the learners need to answer to be evaluated. In this specific example, learners are classified as passed, if at least half of questions were answered correctly, failed otherwise, or no data if they have not reached the minimum number of answers in that topic.

Figure 10 dashboard provides personalized information for a single learner, showing the results of the game session. The top half of the dashboard includes the total count of questions answered and the count of different questions answered; the bottom half contains a pie chart with the percentage of correct and wrong answers during the session and the timestamp of the answers.

Figure 11 dashboard shows the top 10 users with highest number of errors made, with their user identifiers and count of errors.

Each stakeholder only has access to data that they are authorized to see, this feature has been added by the Authentication & Authorization asset side (2.4a).
Current status

Default analyses and visualizations based on the xAPI model developed with ADL Co-Lab are now available and they have been tested with a demo game. Pluggable custom analyses and visualizations can be added according to the requirements of the clients or games.

Links

- Sources
  https://github.com/e-ucm/rage-analytics-frontend
- Release
  https://github.com/e-ucm/rage-analytics-frontend/releases/latest
- Design
  https://docs.google.com/document/d/13rrTuglezXwNTfXriQeg1Jb0H4Jil7I4inpXCVMv8/edit?usp=sharing
- User documentation (shared with 2.1b + 2.4a + 2.4b)
  https://github.com/e-ucm/rage-analytics/wiki
- Launcher (launches 2.1b + 2.4a + 2.4b + 2.4.c.s)
  https://github.com/e-ucm/rage-analytics
2 CONCLUSIONS

D2.5 presents significant challenges in its main goal of producing a customizable dashboard infrastructure that can be configured for each game specifically. Also it includes a set of default visualizations that provide general game information automatically and the possibility to configure more detailed visualizations, specific to each game.

This deliverable describes the processes required to use this new functionality and the design of the architecture. All these dashboard components are already working with data collected using the xAPI data format that the RAGE project has developed in collaboration with ADL Co-Lab.

This is the first deliverable of a complex new functionality and will continue to be improved, addressing usability issues based on the continuous feedback from internal tests and the inclusion into RAGE pilots.

As previously described, all the dashboard components take into account and include security features to assure the data analytics security guidelines defined in RAGE (including aspects of anonymization when possible) and to comply with the law about personal data storage and protection in accordance with EC data regulations.