Democratizing Game Learning Analytics for Serious Games

Víctor M. Pérez-Colado(✉), Iván J. Pérez-Colado, Iván Martínez-Ortiz, Manuel Freire-Morán, and Baltasar Fernández-Manjón

Department of Software Engineering and Artificial Intelligence, Complutense University of Madrid, C/ Profesor José García Santesmases, 9, 28040 Madrid, Spain
victormp@ucm.es

Abstract. Interest in the field of serious games (SGs) has grown during the last few years due to its multiple advantages. For example, SGs provide immersive learning environments, where risky or complex scenarios can be tested in safety while keeping players engaged. Moreover, the highly interactive nature of serious games opens new opportunities for applying learning analytics to the interaction data gathered from the gameplays. These interaction data can be used, for example, to measure the impact of serious games on their players. At e-UCM, we have developed open code tools to support serious game learning analytics (GLA), especially an xAPI tracker that collects the player interactions and sends them to a cloud analytic store, SIMVA. Although this tracker uses the xAPI specification as a basis, it includes extensions tailored to our tools. However, not all game developers have the knowledge to operate our analytics infrastructure or are willing to use our tools. We present the design of a GLA system based on existing software modules, focused on collecting and storing analytics generated by SGs in xAPI format. The main elements of this lean architecture are the Learning Record Store (LRS) and the xAPI tracker. With this work, we aim to facilitate and lower the barrier of applying learning analytics in serious games.

Keywords: Game learning analytics · Serious games · xAPI · Tracker

1 Introduction

The growth in interest in serious games (SGs) is due to their multiple advantages, such as greater attention retention capacity, the ability to teach skills through fun mechanics, or to simulate real environments safely, ensuring more authentic learning. Additionally, SGs that include assessment capabilities can produce evidence-based assessments.

However, few SGs include assessment and, even when present, it is rare to have scientifically validated the assessment with enough learners and in real environments. In most cases, SGs are developed as black box systems [1, 2]. That is, the SG only provides a result (e.g., score), and although they may have a complex internal logic that responds to the interactions and educational needs of the learner, this is not observable from the outside. This implies that there is no evidence of the learning process produced with the game, limiting its educational usefulness, and hindering its scientific validation [1, 3].

F. de Rosa et al. (Eds.): GALA 2021, LNCS 13134, pp. 164–173, 2021.
https://doi.org/10.1007/978-3-030-92182-8_16

At best, the game provides scores, progress, or evaluations as output data; but there is a lack of specific data about the player's process to obtain these results.

To improve the usefulness and reliability of SGs, work has been done mainly on formal game validation [4] and a white-box evaluation or reporting model [5]. Through validation it is possible to ensure that the game produces the expected outcome and, therefore, its application in a real setting should be effective. The most popular method used for validation of SGs is through questionnaires applied before and after using the game in which the desired change is reflected (usually this change is also compared with a control group) [4]. On the other hand, through a white box evaluation, the game sends out the meaningful interactions of the player with the game. This exhaustive report adds meaning or extra information to the activity and can be monitored in real time or analyzed a posteriori to assess the player and even to predict the player's outcome. A particular use case of application of this approach is stealth assessment [6]. We consider that, by combining both methods, formal validation and learning analytics (LA) could be a method capable of providing both assessment and scientific validations.

The open issue is that, despite recognizing the usefulness and potential of LA in the SG community, its application is still limited [7]. Usually, in those cases where analytics are implemented, it is done ad-hoc and from scratch. But implementing LA in games requires specific skills (software development or data analytics) that are not very common in small or medium-sized development studios, which are the ones that usually develop SGs. The use of LA requires not only the deployment of a complex software infrastructure often only within reach of experts, but also meeting both the general and specific educational needs of each game and the regulatory needs for the treatment of information (e.g., EU GDPR privacy law). It is, therefore, necessary to develop methodologies and software modules that simplify this complex task.

To systematize and democratize the application of game learning analytics (GLA), the responsibilities of SG developers should be clearly delimited, with a primary focus on selecting relevant in-game events to be analyzed by data analytics experts. To achieve this goal, two aspects need to be considered from the very beginning: the data exchange format (to achieve common semantics on which to perform the analysis), and the Application Program Interface (API) to be used to communicate formatted data, so SGs can be easily deployed in different environments. Both aspects can be simplified with software modules to both assist with formatting and communicating this information outside the game (i.e., a tracker) and receiving and storing it somewhere (usually in the cloud) securely, while providing at least basic query and analysis functionalities.

Our approach is centered on the use of an e-learning standard designed to capture user interaction as part of an educational scenario such as xAPI (eXperience API), that in the specific case of SGs, provides a format to represent user interactions within the game and that will allow us to record the user traces within the game. The xAPI specification (version 2.0 will become an IEEE standard during 2021) is a specification that provides a statement-based data model that is generic and extensible. The structure of a basic statement consists of the following elements: an actor (*who* performs the action), a verb (*what* action is performed) and an object (the action's *target*). The purpose of this basic structure is to enable communities of practice to define precisely, through custom application profiles, specific vocabularies for each of the elements of statements.

For example, there is an application profile for SGs (xAPI-SG), created by e-UCM in collaboration with ADL as part of the European H2020 RAGE project [8].

As part of previous projects, we developed an open-source tool (tracker) that implements xAPI-SG and simplifies the communication of user traces between the game and an external store, without requiring the SG developer to be an xAPI expert. Until now, this tracker communicated SGs with our analytics tools such as SIMVA and T-Mon, which simplify the management of SG experiments and deployments and automate some aspects of data analysis on xAPI traces [9]. Although these tools are free and open-source, technical knowledge is required for their use and deployment. On the other hand, the xAPI specification also defines a standard API that focuses mainly on sending and querying xAPI *statements*. This standardized API allows an educational tool, such as a SG, to communicate with a generic xAPI Learning Record Store (LRS).

The present work describes how we have combined and adapted our technology (primarily the tracker) to interact with any standard LRS. This should democratize GLA by making it more accessible to game developers, allowing SG developers, educational organizations, and teachers to use the LRSs that best suits their specific needs. For example, some will opt to use an LRS that is already integrated with their existing institutional Learning Management System (LMS), while others may chose a commercial or open-source LRS offered as SaaS. With these enhancements, we can maximize the benefits of using our tracker during SG development (by being able to interact with our SG validation tools) while allowing choice regarding the LRS to use once deployed.

The following sections describe the current model for collecting GLA using the e-UCM group tools; how SG LA can be collected using reusable generic software components; the modifications to our tools to further the democratization of analytics for SGs; and finally, present conclusions and future work.

2 A Tailor-Made Scalable Learning Analytics Ecosystem

Systematization of LA in SGs requires software to store analytics, simplify the collection of analytics within the SG, and communicate it with the analytics store. This section describes our experience developing tools for these problems.

The e-UCM group has developed different tools related to the collection and analysis of data generated by SGs. The most recent is the SIMple VAlidator (SIMVA) platform [9]. SIMVA's main objective is the creation and orchestration of SG validation experiments using GLA. Usually, SGs are validated by using questionnaires before and after the activity in which participants play the SG. SIMVA includes an integrated questionnaire management tool to simplify this process. In addition, during the game activity, SIMVA takes care of collecting the LA generated by the SG [7, 10].

Using SIMVA it is possible to design different user experiments (called studies) for different games with full control over the flow of the experiment, offering, for example, the possibility of comparing two versions of the same game (A/B testing). As part of the management of the experiments, SIMVA allows us to generate a list of experiment participants identified exclusively by anonymized identifier tokens, which allow the actions of each participant in the different activities of a study to be linked together, while complying with regulations related to privacy and data protection. Likewise, the

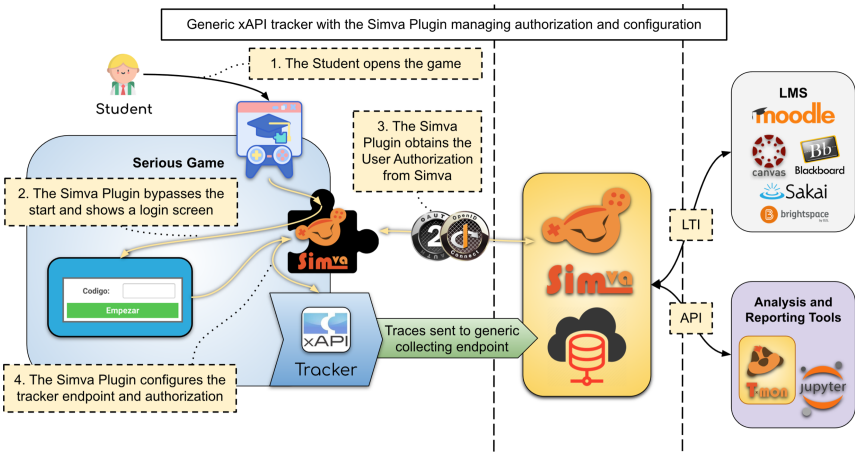

Fig. 1. Interaction flow of a SG with SIMVA, and of SIMVA with other tools such as T-Mon.

researcher or teacher involved in the experiment can monitor the progress of participants in each activity. Finally, SIMVA offers an API that can be used by data analysis experts to download and analyze the data collected during the experiment. This API is compatible with AWS S3, which is one of the APIs commonly supported by big data and data analytics tools such as Jupyter Notebooks.

From a game development perspective, SIMVA provides a plugin compatible with the Unity platform that simplifies authentication and trace submission for study participants/players. The usual process of interaction of a SG with SIMVA is as follows (Fig. 1): 1) players launch the game; 2) the game displays a simplified login interface, where players enter their 5-letter participant code; 3) the game connects to SIMVA by sending the study code and participant code, verifying that the user is assigned to the study and that the activity to be performed is to play the game (for instance, studies may require their participants to complete a survey before playing; in this case, an informative message with a link to the survey would be displayed, and the game would not start); 4) the game obtains the necessary configuration data to use the xAPI tracker and gameplay would start, during which analytics will be sent to SIMVA.

Once the experiment is finished, it is time for data science experts to analyze the data generated during the experiment. To simplify this task, we have created the T-Mon tool built on top of Jupyter Notebooks. T-Mon is a data analysis tool specialized on xAPI data compatible with the xAPI-SG profile, and provides a set of generic analyses and predefined visualizations adapted to this xAPI-SG profile.

Game learning analytics is a complex and error-prone process, requiring skills not always present in a game studio. Collecting game analytics data alone requires linking the game to a data store, identifying the user who is playing the game, and considering many other aspects of security and reliability. The following sections discuss the current capabilities of our analytics tracker, and the design changes needed to extend it so that user traces can be sent to xAPI-standard data stores such as the Learning Record Store (LRS).

3 The e-UCM xAPI Tracker

Adding support for LA to a SG can be a challenge for game developers who do not have experience in LA. This section describes the main features of the tracker software component to identify the functionality required to make it work with a generic LRS.

To simplify the analytics collection process, we have developed a series of reusable software components, called xAPI trackers, compatible with different platforms and programming languages (Unity, .NET, JavaScript, and Java), which were developed and used as part of the EU H2020 projects RAGE and BEACONING. These trackers are middleware components that simplify the sending of traces from games (solving authentication and communication problems), avoiding an extensive knowledge of the xAPI-SG application profile. The rest of this section focuses on the Unity tracker description, which has the most advanced features and the widest application in SGs.

The xAPI tracker provides a high-level interface for interacting with a compatible analytics store (e.g., LRS), specifically targeted for SG development. It currently implements the xAPI-SG application profile and includes another xAPI application profile for creating geopositioned games [11]. In addition, the tracker has other features that differentiate it from other existing xAPI support modules (see Fig. 2):

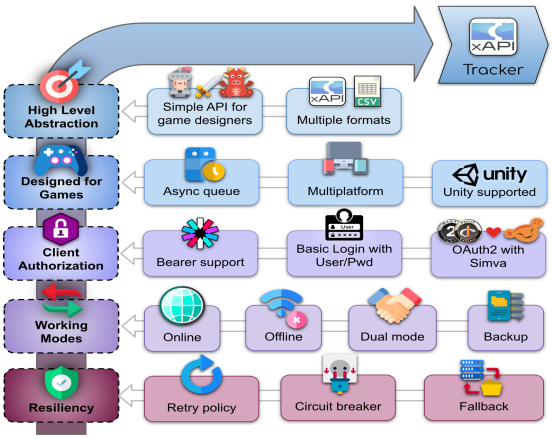

Fig. 2. e-UCM xAPI Tracker functionalities. Its five key aspects are shown on the left.

The first and main feature of the tracker is to provide a high-level and simplified interface so that developers and designers do not have to be experts in the xAPI format to record the game session or send interaction information. In fact, it allows traces to be created in multiple formats. For example, in addition to xAPI format, it is also possible to generate the traces using the more compact CSV format. For example, this is the format used to store a local backup of the traces inside the gaming device (if the backup mode of the tracker is enabled).

The tracker is implemented using Unity's primitives (such as network access), without which it would be more complex to create cross-platform SGs. In addition, the

tracker has an interface to build partial traces to maintain the logical order of events even considering the internal logic of the game and its possible interactions.

The tracker supports authentication and privacy aspects. The current xAPI specification mandates only basic web authentication and OAuth1, neither of which is without problems. Basic authentication only authenticates tools, but not users; and, if game privileges are desired, different credentials for the same game must be created, complicating its use. On the other hand, OAuth1 has been discouraged for some time due to protocol security flaws. Thus, our tracker has been updated to support the new OAuth2 and OpenID connect standards, which are considered secure and allow not only to identify the tool but also the user who is using it. This improves security and minimizes the possibility that a user can access or send data on behalf of another user.

The tracker currently provides three modes of operation: online, offline, and with backups. These modes reflect our experience and the complexities of the actual use and application of SGs in real educational environments, where technical limitations and limited network reliability in schools or institutions where games are deployed games are a frequent occurrence. In online mode the tracker sends the data to an analytics store such as SIMVA, in offline mode the tracker stores the data on the gaming device, and in backup mode both strategies are combined to achieve greater reliability.

The tracker must be robust to be able to operate in non-reliable environments where technical problems may emerge. To this end, the tracker incorporates different strategies, such as exponential back off retry, bulkhead isolation, or circuit-breaker. For example, if the tracker detects that the analytics store is not responding properly for a certain period (which is configurable), it automatically switches to offline mode, and after a certain period it will retry the pending operations.

Despite the advanced features described above, before this work our tracker only implemented the xAPI-SG application profile, and due to issues with authentication and authorization was not designed to interact directly with a standard LRS.

4 DA Standard-Based Scalable Generic Game Learning Analytics Infrastructure

The creation of a platform to support LA for SGs requires different components depending on intended goals and required functionalities. Basic analytics support which includes only the collection of traces from a single game for later analysis is very different from a complete system that supports multiple games and also supports real-time user interaction analysis (e.g., H2020 BEACONING or RAGE); or a complete system to support experimentation with SGs such as SIMVA, where playing a game is only one of multiple activities that can be built into a study.

Game analytics processes are complex and fragile, with many sources of potential errors, such as communication, latency or availability – and it is not always possible to develop a fully integrated approach. The present work describes a strategy for implementing a GLA system based on existing software modules, focusing on the collection and storage of analytics generated by SGs in xAPI format. The main elements of this lean architecture are the Learning Record Store (LRS) and the xAPI tracker.

By adopting the full xAPI specification, which includes both a data format and programming interface for standards-compliant LRSs, we can reduce development costs and allow analytics collected in different deployment environments to share a conforming LRS, or the use of existing xAPI API-compatible services to interact with those LRSs. At the time of this writing, several popular LRSs are available:

- ADL LRS is an open source LRS by ADL, supporting OAuth1. However, this LRS is a reference implementation, designed only to carry out proof-of-concept activities with small numbers of users.
- Learning Locker is an open source LRS with support for basic data analysis and visualization and a business rules layer for easier integration with other systems. It also has an enterprise-oriented SaaS version.
- Rustici LRS is the main reference LRS, as Rustici works closely with ADL; and is offered as part of SCORM Cloud, an LMS for xAPI activities that can act as a mediator between traditional LMSs that support SCORM 1.2 or LTI and CMI-5 packages. This LRS supports similar data processing and business rules to Learning Locker, but offloads reports, analysis and visualizations to SCORM Cloud.
- Yet Analytics LRS is a commercial LRS that is offered in three versions: as a Cloud deployment, as a self-contained LRS compatible with a SQL interface installable on-premises, and as a hardware appliance that can even be deployed in experiments or field activities. To provide analytics and visualizations, Yet Analytics offers an xAPI Sandbox, a free platform based on its Yet Pro v2 LRS that provides on-demand LRSs and supports multiple dashboards.
- Watershed LRS is a commercial LRS that provides reports, performs data conversion to different formats and allows editing trace data or metadata. Watershed's LRS is currently free, although features such as dashboards, analysis, or visualizations require paid licenses.
- Apereo OpenLRW is an open source project that provides an xAPI-compliant LRS, as well as other analytics specifications such as IMS Caliper or IMS OneRoster. To analyze data, it can be connected with Apereo OpenDashboard-API, a framework for creating dashboards and visualizations.

An LRS is only useful if it receives traces that it can store for later analysis. We have identified three open-source trackers that can be used for development of SGs in Unity:

- TinCan.NET by Rustici Software is developed on.NET Framework 3.5, and can therefore be used from within the Unity platform. However, due to the peculiarities of Unity's.NET support, development of cross-platform games presents several issues. In addition, its use of synchronous communication clashes with the Unity game development model. Rustici also provides a JavaScript version: TinCan.JS.
- UnityGBLxAPI (formerly GBLxAPI) by Dig-iT! Games is a wrapper for TinCan.NET, with several improvements: i) better setup and compatibility, since it is Unity-specific; ii) an asynchronous queue approach to send the statements, which avoids game freezes while sending data; and iii) cross-platform compatibility of the generated games by using the specific network primitives offered by Unity. As a limitation, this tracker loses the ability to interpret LRS certain responses, preventing

developers from relating traces to each other via xAPI StatementRefs; also, it may be difficult to update if the underlying TinCan.NET library changes.
- Unity-xAPIWrapper by ADL is a lightweight tracker, with less dependencies and easier to use than the above alternatives. It uses Unity primitives for network communication, with similar advantages to those of UnityGBLxAPI. Its main limitations are that it is very simple, and requires more work from developer to achieve a flexible, secure, and resilient tracker; and a low rate of updates.

To use the xAPI API to communicate with an LRS from a game, certain configuration parameters must be known: base URL of the LRS, and usernames and passwords for the LRS. The xAPI specification does not detail how activities obtain these parameters. Fortunately, this problem has already been addressed in the CMI-5 specification [12]. CMI-5 is considered to be a natural evolution of SCORM, but using xAPI to communicate activity interactions with an LRS. This specification addresses several aspects: packaging, launching (including the exchange of credentials and configuration parameters needed to communicate with an LRS), and an xAPI application profile (xAPI-CMI5) that provides insight into the status and progress of an activity. Despite its benefits, CMI-5 adoption is currently very limited, especially by LMSs.

5 Adaptation of the Tracker for Use in the Generic Architecture

The use of a standard LRS can have advantages, especially for the actual exploitation of SGs in real educational settings, for example, by integrating them with a pre-existing e-learning infrastructure. We believe that our tracker has several unique and desirable characteristics, but we are also aware that to lower the entry barrier of applying LA we should be compatible with xAPI-compliant LRSs. In this regard, we have identified the following modifications to achieve this compatibility with our tracker: i) improve adherence to the xAPI specification, ii) provided support for the CMI-5 protocol, and iii) implement different modes of operation that allow running both SIMVA and a generic LRS configured either manually or via CMI-5 release.

Two additional modifications allow the standard to be followed more closely. On the one hand, integration with SIMVA extends the standard with additional services such as user authentication or retrieval of game configuration. We replaced them with traces following the CMI-5 profile. On the other hand, the generated traces were intended to be stored in a specific storage associated with the specific activity created in SIMVA to represent the game session. However, standard LRSs do not partition data like this. Using the *context.contextActivities* property of a statement, it is possible to specify the context in which it has been generated, for example, specifying in which SG the trace was generated, the educational activity that represents the session (or sessions) where the SG is used, or even the course where the educational activity is being included. Additionally, by using the *context.registration* property we can identify statements that belong to each attempt of a player.

Moreover, implementing the CMI-5 launch protocol allows the tracker to receive both the location of the LRS where information should be sent, as well as the authentication of the user who is developing the activity. Should CMI-5 not be available, the tracker can read this configuration from locations chosen at development time.

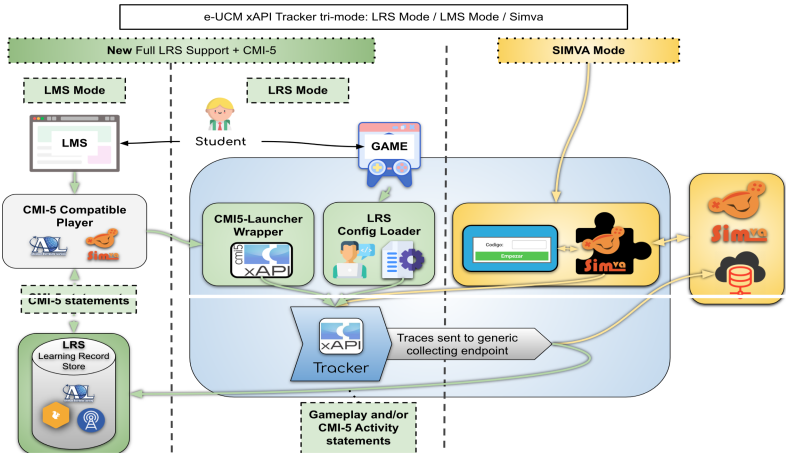

Fig. 3. Tracker working modes, internal components and interaction with external systems

Finally, in addition to the operating modes described in Sect. 3 (online, offline, backup), three working modes have been established: the SIMVA mode (the current mode), the standard LRS mode and the LMS-CMI-5 mode (Fig. 3). The SIMVA mode represents the tracker behavior prior to proposed modifications, and uses specific capabilities provided by SIMVA to simplify validation of SGs. In the standard LRS mode, the tracker will be configured to use the xAPI API exclusively, obtaining the configuration parameters for the LRS from the pre-established locations configured during development. Finally, in LMS-CMI-5 mode, LMS support will be used to launch the SG as a CMI-5 activity, retrieving configuration parameters directly from the LMS.

6 Conclusions and Future Work

Learning analytics is a decisive step forward in adopting serious games as reliable tools in the learning process. Analytics provides evidence beyond simple results, and allows insights into how players achieved those results. However, GLA implementations are still complex and fragile, and require significant technical skills to deploy.

In previous work, we proposed a SG analytics architecture that relies on the use of xAPI as a data standard and on a set of software modules to capture SG information (tracker) and communicate it to a data warehouse in the cloud for secure storage and analysis (SIMVA, T-Mon). However, our approach is difficult to integrate with existing infrastructure. To decrease the cost of entry and increase reliability and adoption, this work proposes a similar architecture using pre-existing software components in the xAPI ecosystem (e.g., trackers and LRS). Thanks to the use of standards, both xAPI and LRS, it is possible to swap some of the software components without causing vendor lock-in. Furthermore, it is possible to reuse components such as LRS, which are increasingly already deployed in existing e-learning infrastructure.

Since the e-UCM tracker was, until now, strongly coupled with e-UCM tools such as SIMVA and did not offer full support for third-party LRSs, we are implementing

two new standards-compatible modes for the tracker: an LRS mode to connect to any standards-based LRS, and an LMS mode to allow our newly CMI-5 compliant tracker to convert SGs into packages that can be easily deployed as activities in LMSs.

Acknowledgements. This work has been partially funded by Regional Government of Madrid (eMadrid S2018/TCS4307, co-funded by the European Structural Funds FSE and FEDER) and by the Ministry of Education (TIN2017-89238-R, PID2020-119620RB-I00).

References

1. Chaudy, Y., Connolly, T.: Specification and evaluation of an assessment engine for educational games: empowering educators with an assessment editor and a learning analytics dashboard. Entertain. Comput. **27**(September), 209–224 (2018). https://doi.org/10.1016/j.entcom.2018.07.003
2. Freire, M., Serrano-Laguna, Á., Manero-Iglesias, B., Martínez-Ortiz, I.: Game learning analytics: learning analytics for serious games. Learn. Des. Technol. https://doi.org/10.1007/978-3-319-17727-4
3. Marchiori, E.J., Torrente, J., Del Blanco, Á., Moreno-Ger, P., Sancho, P., Fernández-Manjón, B.: A narrative metaphor to facilitate educational game authoring. Comput. Educ. **58**(1), 590–599 (2012). https://doi.org/10.1016/j.compedu.2011.09.017
4. Boyle, E.A., et al.: An update to the systematic literature review of empirical evidence of the impacts and outcomes of computer games and serious games. Comput. Educ. **94**, 178–192 (2016). https://doi.org/10.1016/j.compedu.2015.11.003
5. Hauge, J.B., et al.: Implications of learning analytics for serious game design. In: Proceedings of the IEEE 14th International Conference on Advanced Learning Technologies, ICALT 2014, pp. 230–232 (2014). https://doi.org/10.1109/ICALT.2014.73
6. Shute, V.J.: Stealth assessment in computer-based games to support learning. Comput. Games Instr. **55**(2), 503–524 (2011)
7. Alonso-Fernández, C., Calvo-Morata, A., Freire, M., Martínez-Ortiz, I., Fernández-Manjón, B.: Applications of data science to game learning analytics data: a systematic literature review. Comput. Educ.**141**(June), 103612 (2019). https://doi.org/10.1016/j.compedu.2019.103612
8. Serrano-Laguna, Á., Martínez-Ortiz, I., Haag, J., Regan, D., Johnson, A., Fernández-Manjón, B.: Applying standards to systematize learning analytics in serious games. Comput. Stand. Interfaces **50**, 116–123 (2017). https://doi.org/10.1016/j.csi.2016.09.014
9. Perez-Colado, I.J., Calvo-Morata, A., Alonso-Fernández, C., Freire, M., Martínez-Ortiz, I., Fernández-Manjón, B.: Simva: simplifying the scientific validation of serious games. Proceedings of the IEEE 14th International Conference on Advanced Learning Technologies, ICALT 2014, pp. 113–115 (2019). https://doi.org/10.1109/ICALT.2019.00033
10. Alonso-Fernández, C., Martínez-Ortiz, I., Caballero, R., Freire, M., Fernández-Manjón, B.: Predicting students' knowledge after playing a serious game based on learning analytics data: a case study. J. Comput. Assist. Learn. **36**(3), 350–358 (2020). https://doi.org/10.1111/jcal.12405
11. Perez-Colado, V.M., Rotaru, D.C., Freire, M., Martinez-Ortiz, I., Fernandez-Manjon, B.: Learning analytics for location-based serious games. In: 2018 IEEE Global Engineering Education Conference (EDUCON), vol. 2018, pp. 1192–1200, April 2018. https://doi.org/10.1109/EDUCON.2018.8363365
12. Bakhouyi, A., Dehbi, R., Lti, M.T., Hajoui, O.: Evolution of standardization and interoperability on E-learning systems: an overview. In: 201716th International Conference on Information Technology Based Higher Education and Training, ITHET 2017 (2017). https://doi.org/10.1109/ITHET.2017.8067789