Tools and Approaches for Simplifying Serious Games Development in Educational Settings

Antonio Calvo, Dan C. Rotaru, Manuel Freire, Baltasar Fernandez-Manjon
Dept. Software Engineering and Artificial Intelligence
Universidad Complutense de Madrid, Facultad de Informática
C/ Profesor JoseGarciaSantesmases, 9 28040 Madrid, Spain
{antcal01, drotaru, manuel.freire, balta}@fdi.ucm.es

***Abstract*—Serious Games can benefit from the commercial video games industry by taking advantage of current development tools. However, the economics and requirements of serious games and commercial games are very different. In this paper, we describe the factors that impact the total cost of ownership of serious games used in educational settings, review the specific requirements of games used as learning material, and analyze the different development tools available in the industry highlighting their advantages and disadvantages that must be taken into account when using them to develop a serious game.**

Keywords—serious games; total cost of ownership; applied games; learning analytics

I. Introduction

Digital games have increasingly gained relevance in society to the point of becoming one of the most popular forms of entertainment. The total revenue of the digital games sector, in the US market alone has grown from USD 2.6 billion in 1996 to over 15.4 billion in 2013 [1]. The digital games sector has suffered a moderate slowdown in the last years [1] due to the economic crisis, but is still able to generate massive revenues. For instance, in September 2013 Rockstar released *Grand Theft Auto V*, which generated more than USD 1 billion in retail sales during its first three days on sale, being the fastest entertainment product to reach this milestone, including digital games and feature films [2], [3]. However, those big budgets and numbers are orders of magnitude removed from those of serious games or educational games.

Digital games, when used to further goals such as health or education rather than simply entertain, are called serious games (SGs) [5]. Continuous education, at all ages, is becoming one of the great priorities for our society, and increasing student motivation, always an important problem in education, is now more relevant than ever. Since large and diverse portions of the population already play games regularly as part of their daily or weekly routine, serious games are a powerful tool to increase student engagement and motivation. Games provide players with challenging and immersive environments where they can fail without consequences. This safe, failure-tolerant environment allows players to experience situations from different points of view, or experiment freely and exercise their curiosity circumstances that are good precursors of learning. By providing constant feedback to the player, games can be used for procedural learning [4]. And by motivating and challenging the players to explore and overcome specific problems, serious games are gaining interest in teaching and training students. Serious games have been successfully applied to different domains such as health [6]–[9], marketing [10], research [11], and to multiple other disciplines, where they have proven to increase student motivation in the learning process [12]–[14] stimulating their focus and concentration [5], [15], [16]. Studies have shown SGs increasing academic performance [17]–[19] and providing an optimized learning process [20], [21], as part of a growing set of experiments that prove the effectiveness of correctly designed and implemented serious games [22]–[26].

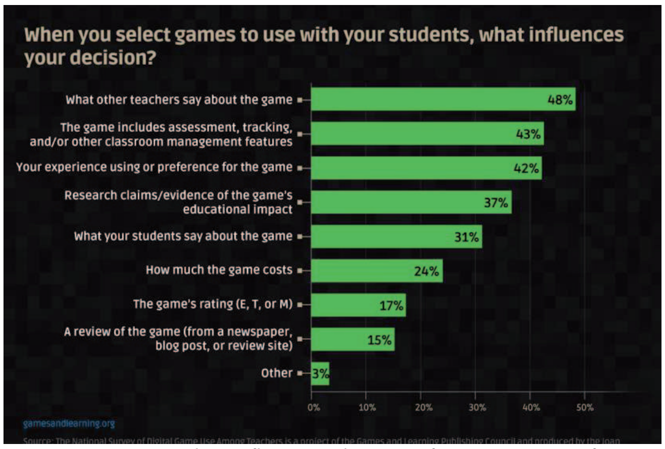


Fig. 1. Factors that influence adoption of serious games, from http://gamesandlearning.org

However, to generalize the use of games in formal education settings, educators need additional support. Beyond positive reviews by other teachers, games that include assessment, tracking and other classroom management features are strongly preferred (see Fig. 1). Evidence of educational impact is also important. Both factors can be addressed by using new game learning analytics techniques. However, these techniques need to be applied early and often, ideally starting during the design stages of the SG and continuing through the final implementation. Additionally, once a game has been considered educational content, it should remain available and ready to use regardless of technological changes or platform evolution. The decision to benefit from SGs or forego their usage completely hinges on the total cost of ownership (TCO) of the game including the cost of developing, fine-tuning, deploying and maintaining such a game.

This paper describes and analyzes the main factors that contribute to serious game TCO, and proposes strategies to keep this TCO in line with the SGs' requirements (including budget and maintainability). Section II describes the different lifecycle models for Serious Games. Section III analyzes different approaches to educational game development, mostly applicable to the game developers themselves. Section IV compares the different types of tools from a developer perspective. Finally, Section V presents our conclusions and outlines future work.

II. Serious Game Lifecycles

The growth of the games industry and market is echoed by a corresponding growth in game development. There are different methodologies and authoring tools that improve the game development process, which requires a careful integration of multidisciplinary skills and efforts [27]. Furthermore, for SGs that have an educational purpose, this process is even more demanding, as not only are budgets much smaller: it is also necessary make the educational aspects of the game work while maintaining high player engagement [28].

A. COTS games

Commercial Off-the-Shelf (COTS) games are, as their name indicates, commercially available digital (computer or console) games that are designed for entertainment rather than educational purposes. Some of these games can be given an alternate use in classrooms. However, making effective use of commercial games in the classroom requires careful though on how to extract this unintended pedagogical value. Indeed, as the following quote from [30] illustrates, certain games can be used to illustrate many more topics than teachersmay initially expect:

> "While some might assume that *Zoo Tycoon* might have application for biology, zoology, and ecology from the title alone, many would be surprised to learn that some of the other primary content areas for this game are economics, business, marketing, and mathematics".

The ability of the teachers to find and convey these educational aspects of the game is less relevant that their knowledge of the curriculum with which they are working to successfully achieve educational objectives [29]. And because the commercial games are not made to teach content, they will not be sufficient as the only teaching tool. Quoting from [30]:

> "As the designer, you will need to identify where there are gaps and inaccuracies in the game content, and where the strategies the game supports for solving the challenges do not align with your learning outcomes (e.g., trial and error vs. reasoned thinking) or may lead to misconceptions or an incomplete picture of the content and skills."

Before using these games, the teacher must research how to best integrate the game into the classroom. The fixed duration of classes is a significant constraint when planning and implementing game sessions in schools [29].

Multiple online sources advocate the use of COTS for classroom learning experiences, such as [29], [31]. *Civilization, Age of Empires II, CSI, Limbo, Universe Sandbox, The Sims 2, Rollercoaster Tycoon, Sim City 4, Imperium* and *Tropico* are only some examples of these kind of games used to teach history, forensics, criminal justice, physics, social sciences, civil engineering, business management etc.

Using COTS in the learning process is often more cost-effective than developing serious games from scratch [32]. However, finding a COTS game that can be used to teach a specific set of learning requirements may be impractical – such a game may simply not exist. Additionally, COTS games may require substantial modification to provide the type of game learning analytics and classroom support that teachers would take for granted in actual serious games. While it is possible to completely forego this support and rely on sharing game-play experiences through traditional classroom interactions, the lack of in-built support for evaluation requires more effort for teachers in COTS-used-as-SGs than in pure SGs.

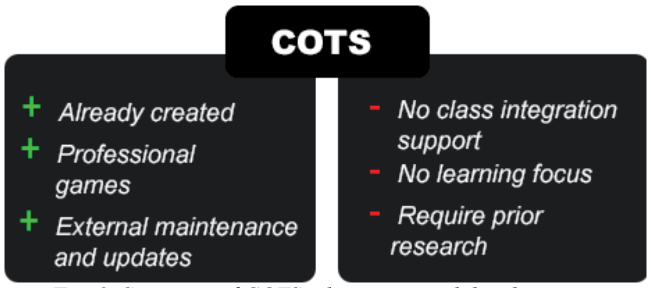


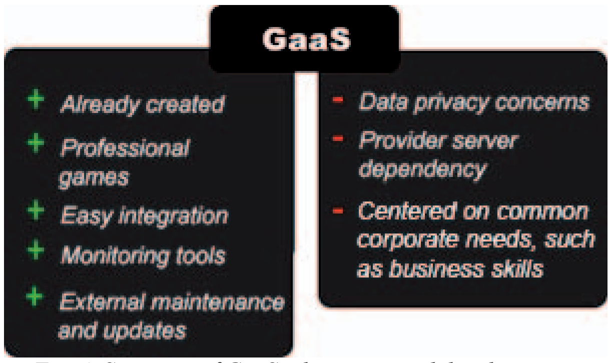
Fig. 2. Summary of COTS advantages and disadvantages

B. Games as a Service

Another second way to provision serious games in learning environments is to hire the services of specialized companies that provide both the game and the associated infrastructure "as a service". These businesses use learning games as services to train students and employees or to provide tools to aid in their instruction. An example is GameLearn [33], which provides games for improving employee skills. One of their games-as-a-service (GaaS) is *Merchants*, which focuses on improving negotiation and conflict-resolution skills, while another, *Pacific*, provides leadership and team-management training. A very different GaaS, *Classcraft* [34], aims to transform any classroom into a role-playing game that fosters stronger student collaboration and encourages better behaviour [34]. Classcraft allows the teachers to introduce personalized questions in the game and rewards the students with in-game bounties.

GaaS provide a set of benefits, such as an easy integration with the learning process, control and monitoring tools, and useful training, among others. GaaS are designed to be quickly integrated with the teacher's learning process providing at least a web-based platform that enables teachers and students to easily manage all the complexities on the game without installing software. GaaS, unlike COTS, also have an analytics layer that supports student assessment to give a clear view of how the game is being played. GaaS teach a series of soft skills such as leadership, team management, productivity, negotiation and team work, among others. Their cost it is considerably lower than the costs of developing a serious game from scratch.

However, there are some concerns that should be taken in consideration. The collected data in stored on the servers of the company supporting the game, which may result in data privacy concerns. The game's internationalization might suppose a problem if the desired language is not supported by the game. Additionally, GaaS companies are mostly interested in building a game once and serving it repeatedly to different corporate customers. Therefore, they are scarce or non-existent in knowledge areas that are in lower demand across the enterprise.


GaaS
+ Already created
+ Professional games
+ Easy integration
+ Monitoring tools
+ External maintenance and updates
- Data privacy concerns
- Provider server dependency
- Centered on common corporate needs, such as business skills
Fig. 4. Summary of GaaS advantages and disadvantages

C. Pre-existing Serious Games

These are games that, unlike COTS, have been developed specifically for learning specific topics. Unlike GaaS, the client only has to pay for the game once (if at all – many are freely available), and unlike COTS, the game is specifically designed with learning goals in mind, making it much more amenable to classroom use.

There are many examples of specialized games. For instance, learning a new language has become a demanded skill in our society. Duolingo [35] is the perhaps the most influential language-learning game, and has been proven to significantly increase the overall average language skills of its players[36],[37]; while offering study courses for languages such as English, Spanish, French, German, Italian, Portuguese, Irish or Russian. *Citizen Science* [38]allows players to attach a real-world context to the research that is done to understand fresh water science. *CodeSpells* teaches programming. *Relive* is a first person 3D game developed by the Italian Resuscitation Council, using the Unity engine, which instructs on CPR protocol methods. Relive focuses in preparing its players to intervene in cases of cardiac arrest, offering two different game modes: a story mode where the player must achieve certain objectives in order to move forward; and a tournament mode, based on a simulated emergency scene.

As in COTS, even though there are many specialized SGs, sometimes it can be hard to find an existing specialized game that suits a specific pedagogical goal. However, if the teacher does find one, they are certainly the easiest way to introduce game-based learning into a classroom.

Unlike GaaS, some specialized games might not provide learning analytics functionalities. In order to implement the learning assessment support, the total cost will increase. On the other hand, if the game provides learning analytics functionalities, the collected data privacy policy must be considered. There might be data privacy concerns if the collected data is stored on third-party servers. If the game is executed locally and does not store player data in external servers, data privacy concerns are greatly reduced.

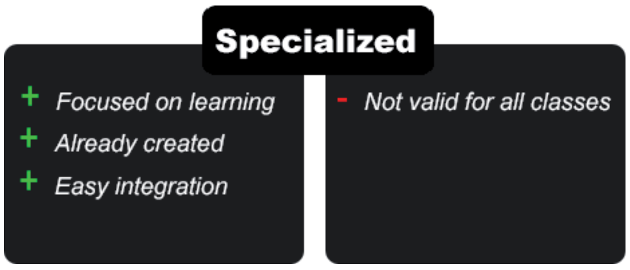


Fig. 3. Pre-existing SG advantages and disadvantages

D. Developing serious games ad-hoc

There are authoring tools that simplify such a complex endeavour, even to the point of not requiring any programming skills. For instance, *Adventure Game Studio* or *RPG maker* both allow the development of an adventure or role-playing game without writing a single line of code. These tools contrast with general-purpose environments, such as *Unity*, *ShiVa* or *Game Maker Studio*, which allow the creation of much more complex games of different genres, and are suitable for amateur and indie developers, and even for professional studios of much greater size and budget.

While authoring tools assist with the actual game code, developing a game requires the use of a dizzying array of additional programs to manage and create the graphics and sounds that will make it come to life. Image editors, 3D editors –such as *Maya, Blender* or *3DS Max*–, sound authoring tools, and so on are highly specialized, and commercial game budgets are often dominated by *game asset* development, as such artistic elements are often termed. Even in low-budget SGs, it is often necessary to hire a graphic designer for some assets, buy others, and/or search for freely available alternatives.

Developing a game also requires the choice of the platforms where it is going to be played and what data is more important or relevant for the course or for students. The widespread use of the Unity game engine, providing an easy-to-use graphic user interface editor, has increased the amount of people interested in game development, including professional development teams, small indie developers and people with limited programming skills and development knowledge.

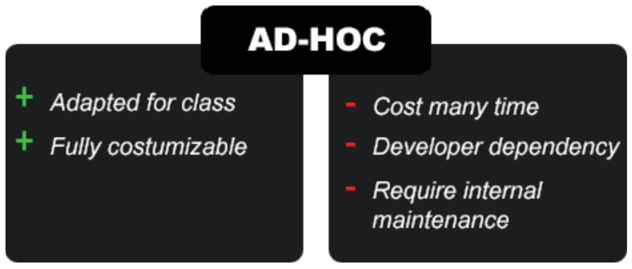


Fig. 5. Ad-hoc SG advantages and disadvantages.

The option of developing a game from scratch is, without doubt, the most expensive in terms of time and resources: it requires the use of multidisciplinary skills –design, implementation, content creation, etc– in order to scaffold, code and flesh out the game. Furthermore, the costs are increased if an additional layer of learning analytics is required. On the other hand, developing a serious game ad-hoc provides the highest amount of control over the final result. This is relevant when the learning requirements are very specific or demanding.

III. Game Development And Approaches

There are a many of different tools available for digital game development, depending on factors such as game genre, purpose, platform, team size or available budget. These tools have in common the little support provided for any activity related to the design of the game. This limitation is especially relevant at the very beginning of the process, when the team is focused on capturing and evolving the concept of the digital game into a stable design ready for implementation.

This section provides a concise overview of the state of the art in game development tools, structured into four subsections.

A. AAA

In the authoring tools industry, *AAA* –pronounced "triple A"– is a classification term used for authoring tools intended for teams with the highest development budgets and levels of promotion or the highest ratings by reviewers. An authoring tool considered to be AAA is therefore expected to be high quality, and focused on the development of high quality games – a.k.a. AAA games. To achieve this goal, these tools provide optimized features needed in a game (artificial intelligence, physics engine, animations, level design, etc.) as well as support for the roles that take part in the development (programmers, graphic artists, testers and designers). Their user interface is intended for experts, focused on implementation and advanced features provided by the tool to create the game, and leaving aside the features that would allow for rapid prototyping of game ideas.

Among other features, these environments provide an engine that abstracts platform-dependent features from the game code. AAA authoring tools are designed by and for programmers experienced in game development, and manage different features such as assets, input/output, network connections, the rendering pipeline, etc.

Regarding content creation, level designers can create new content for the game without modifying any code from the engine itself. These editors also provide authoring tools for animations and assets targeted at artists with little or no programming knowledge. *CryEngine* and *Unreal* Engine are two outstanding examples. Developed by *Crytek* and *Epic Games* respectively, they were initially used to create two successful commercial entertainment games (*Unreal Tournament* and *FarCry*), and have been actively improved ever since. Both have been used in dozens of other commercial games across different platforms (consoles, PC and mobile devices) and feature multiple licensing tiers, which can represent a substantial portion of the game development budget. For example, the highest tier for Unreal Engine 4 requires 5% of the total game profits.

B. All in one

The AAA tools previously exposed are very powerful but at the same time their cost is too high for most developers. The team that created CryEngine numbers nearly 300 game-development professionals from over the world. There are other semi-professional tools created for teams with lower budgets.

One of the most popular is Unity, an all-in-one (AiO) tool equipped with all the necessary modules to configure all the aspects of a game, including the ability to export for different platforms (consoles, PC, Mac, Web and mobile devices). Currently, it is also being used for mobile development by professional teams. Nintendo's Wii U video game console uses Unity as the default software development kit (SDK) including a free copy with each Wii U developer license.

AiO tools lack native support for content creation, which is done with other tools such as Maya and *3D Studio*.

C. Frameworks

The main focus of these tools is to provide lower-level access to the game engine. In this section we can find engines, frameworks and libraries focused on digital games, which can be commercial or free, and are designed so that the programmer makes use of its desired development environment (Eclipse, NetBeans, IntelliJ, Visual Studio, etc.). Examples are *libGDX*, *Ogre3D* or *Allegro*.

In general these tools require advanced game programming knowledge and can also target different platforms.

D. Specialized tools

These tools have less functionality and potential but they simplify the game creation process in order to bring them to a more casual public. They are mostly used in education, aimed at teachers and students to allow them to make simple games with a high educational value. One of the first examples is GameMaker, developed by Mark Overmars. Scratch [39] is another popular tool, where kids learn to program while they create digital games [40]. eAdventure [41] is a tool that started out with the goal of allowing teachers to create educational games; it focuses on the development of two different types of 2D *point-and-click* games: *first person*, where the scene is seen through the eyes of the main player, and *third person*, where the player is represented as an avatar in the scene.

IV. Comparison Between The Different Kind Of Tools

The use of the different types of tools has effects on the monetary costs, time, and the end result. It is important to have a deeper understanding about these tools and their applicability. Below we analyze some of the properties to consider.

A. Platforms

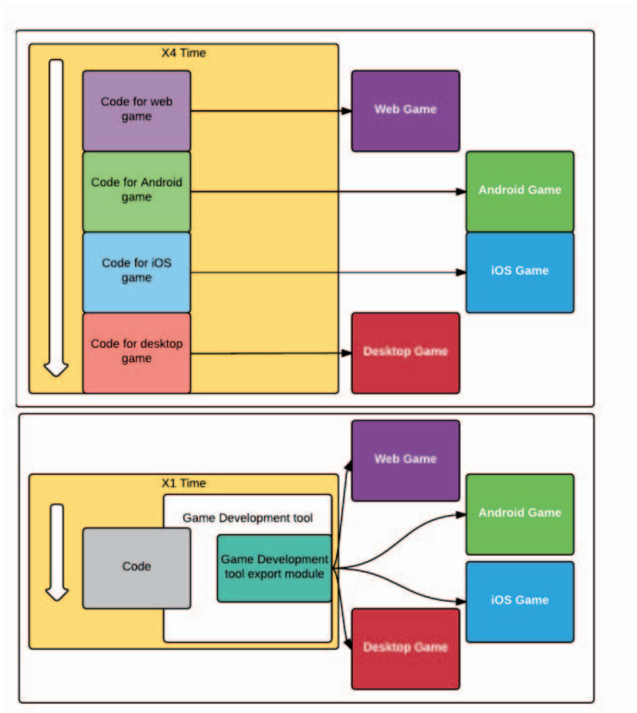


Fig. 6. Multi-platform game development time costs comparison between the traditional method (top) and making use of a tool/framework (bottom).

One of the first things that should be considered in a game development is the set of target platforms that it should be runnable in. In serious games, the most common environment is the desktop, where the three main Operating Systems are Windows, Mac and Linux. With the expansion of mobile devices,it is increasingly frequent to find games developed for mobile platforms, such as Android, iOS or Windows Phone. Some tools do not have the capacity to export the game for all these platforms, and lack of support for the chosen platforms often determines tool choice. Fig. 6 illustrates the time-savings realized by using multi-platform game development frameworks or tools.

AAA tools are specialized in high budget professional games and deploy to the highest amount of platforms. Their engines can create games with the latest and greatest graphics quality, physics and technologies. On the other side, frameworks and specialized tools have more platform deployment restrictions. Generally they provide support for one or two platforms (i.e. Windows and Android). As an exception, the libGDX framework supports development for Windows, Mac, Linux, Android, iOS and HTML5.

Unity has the industry-leading multi-platform support. Unity can export to a very wide range of devices across different platforms. IOS, Android, Windows Phone, BlackBerry and Tizen are the main mobile devices platforms, and are all supported by Unity. For the desktop environment Unity can export to Windows, Windows Store Apps, Mac, Linux and Steam OS (Linux). Unity also targets the web through its Unity Web Player plugin or by directly exporting to WebGL. The past and current generation of game consoles have sold an important amount of units, as we can see in Fig. 7.

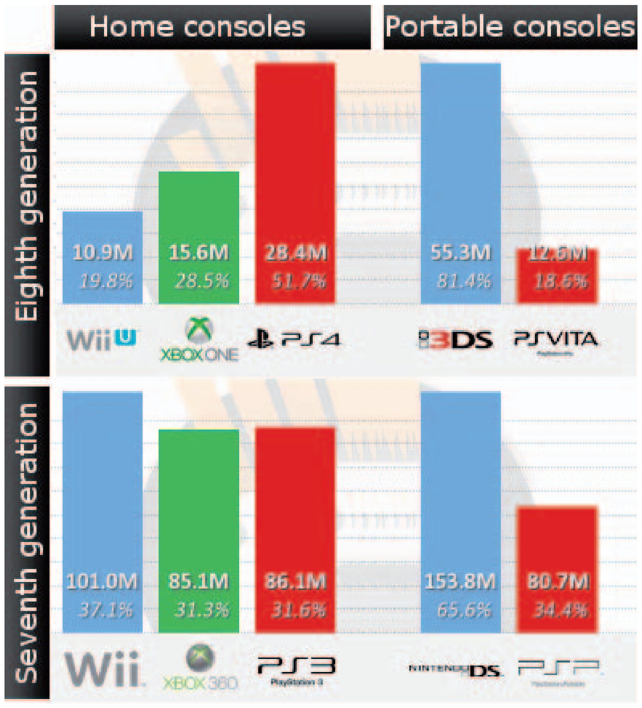


Fig. 7. Consoles sales as of 14th November 2015, from http://www.vgchartz.com/

Unity allows the developers to target PlayStation 4, PlayStation 3, Xbox One, XBox 360, PlayStation Mobile, PlayStation Vita and Wii U free of charge. Publishing to a console requires an additional approval process that varies from platform holder to platform holder. Unity also provides support for Virtual Reality and Augmented Reality platforms such as Oculus Rift, Gear VR and PlayStation VR. Microsoft HoloLens support has been announced –as of April 29, 2015– and is still under development as of November 2015. Other platforms supported by Unity are Android TV, Samsung Smart TV and Windows Universal Platform.

The difficulty to publish games on different platforms must be considered by the developers when choosing a desired engine or framework. For instance, while libGDX does a good job of easing deployment to different platforms, multi-platform deployment typically entails multiple rounds of debugging and additional testing. The libGDX code base is written in Java and provides great cross-platform integration between desktop –Windows, Mac, Linux– and Android devices as long as the developer keeps the source code compatible with Java version 6 during compilation. For HTML5 deployment, libGDX uses Google Web Toolkit to compile the source to highly optimized JavaScript code that runs across all browsers.

Unity offers a faster solution by providing a user interface for the deployment process. Unity's solution requires less platform-specific knowledge from the developers and can speed up the development process. There is typically some minimal effort required, such as integrating with each platform's store for in-app purchases.

Developing games with technologies that might get discontinued and deprecated is arisk that can be prevented by carefully choosing the most suitable development framework or engine. For instance, *Science Pirates: The Curse of Captain Brownbeard* is a case of an educational game that is now difficult to play because of an obsolete target platform. The proposed solution is to invest in maintainable game development frameworks or engines that can deploy to multiple platforms from the same codebase, that are properly documented, have the support of active communities and provide features that can satisfy the game's requirements.

B. *Game mechanics and interactions*

Another thing to keep in mind is the kind of game and the mechanics that must be developed. Games may vary in features, mechanics, and user interaction. Some tools only support the development of a specific type of games, especially the specialized tools –eAdventure focuses on *point'n'click* game development– that generally only allow the development of one or two genres. RPG Maker is a specialized tool targeting RPG games. AAA and all-in-one tools allow the creation of a much broader spectrum of game genres, e.g. platforms, first or third person shooters, adventures, simulations, strategy games, etc.

Specialized tools provide easier ways to create simple games, faster and with lower costs because these games require fewer scripts and elements, and their graphic user interface editor is highly optimized for the creation of a specific game genre. A caveat is the difficulty to innovate or add new features to these kinds of games. As an example, while eAdventure, allows the creation of *point'n'click* games in few hours assuming pre-existing graphical assets, with all mechanics and characteristics that characterize the adventure graphics games, adding additional game mechanics, such as 3D shooting mini-games, would be difficult or impossible.

C. *Features*

In order to choose the best game development framework or engine, the nature of the graphics –2D or 3D– must be taken into consideration. Some tools can only make 2D games. It is necessary to keep in mind that not all the tools have the same affordances for development.

Unity supports both approaches, but it was designed mainly for 3D game development while libGDX, which has been originally designed for 2D games, also supports 3D. Unity has officially supported 2D development since the release of its 2D module, introduced in Unity 4.3 (November, 2013). The 2D dedicated engine in Unity was introduced because of the specific challenges in 2D game development, for instance the use of sprites. Unity's 2D module introduced tools to work with sprites and a physics specialized for 2D mechanics, tiles editor, sprite animations, masks, improved packages manager and a better performance for the 2D engine.

Game development frameworks are different than game engines because they provide different features. Game development frameworks are a collection of lower level libraries, tools and other frameworks used to create games. Game engines encapsulate powerful logic designed to create games and provide more high-level features. Game engines may have a graphic user interface editor while a framework is mainly code-based. Unity has a visual editor that can increase development speed. Developers have to consider whether the tool is actively supported or there is a risk of it becoming discontinued. The frameworks and engines have to continuously update their features with the new emerging technologies, improving the game's development workflow and guaranteeing that deployment to newly-appeared platforms will remain an option.

Also, engines such as Unity and Unreal have been designed considering the profiles of game developers and also designers, providing easy-to-use *drag and drop* editor user interfaces. They also provide *asset stores*, repositories of different kinds of assets –graphics, sounds, game objects, etc.– that can either be downloaded for free or for a fee.

Yet another important feature is ease of testing. Engines and tools that export to mobile devices allow the developers to test the game in the platform while the game is developed. Having to export the game to a different platform every time a new feature must be tested or to find and fix a bug can be tedious. It is important that the tool allows the game to be easily played from within the same platform that it has been developed in. Engines like Unity or Construct2 have this property, but not all frameworks do. For example, while libGDX does allow such testing, AndEngine or E3roid do not.

D. *License and Cost*

The license of a game development framework or engine may be a critical factor when choosing which to use. In order to be competitive, Unreal Engine's license has been made free, with the condition of paying a 5% royalty on purchases of games and applications released by the developers after the first USD 3,000 of revenue per product per quarter.

LibGDX is licensed under Apache 2.0 and maintained by its user community. The code is open source, available at GitHub. Developers can use it free of charge, without strings attached in commercial and non-commercial projects.

Unity has a free version –the Personal Edition– available to developers with revenue under USD 100,000 in the previous fiscal year. Unity's Professional Edition license has a minimum cost of USD 75/month and packs additional features not available in the Personal Edition: customizable splash screen, Unity Analytics Pro, prioritized bug handling, Game Performance Profiling, Unlimited Revenue and Funding, professional editor skin, etc. Unity offers support to large enterprises with its tailored Enterprise Solutions, a package with additional features such as source code licensing, on-demand support and volume discounts available on Unity licenses. Unity Education aims to support learners, educators and educational institutions through special offers on licenses, content, services and resources –e.g. Professional Skills Standard and Curricular Framework– a grand program for secondary institutions, discounted tickets for students and faculty to Unity's Unite developer conferences, etc.

E. *Documentation*

The framework or engine documentation reduces the developer's learning curve and is an important factor to improve reusability and maintainability. The documentation is composed of several kinds of documents and contents such as wikis, code documentation, manuals and tutorials, example code and demos, books and additional third-party and community support.

Wikis help developers understand the tools' abstract –and most important– concepts. Manuals and tutorials teach "best practices", code examples and demos. An active community helps the developers with specific problems, issues and misunderstandings which can be used to further clarify the documentation and direct tool improvements.

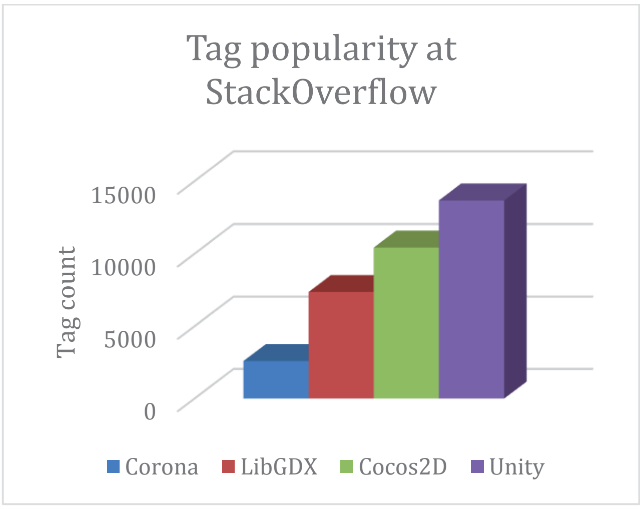


Fig. 8. Tags popularity about some game development tools in StackOverflow community http://stackoverflow.com/

Community size is closely related with the tools' efficiency, effectiveness, and growth potential. It is important to consider not only the documentation, but also the community behind the tool. The community's main goal is to help developers –indie or professional teams– resolve their unexpected problems and doubts. In addition, one of the key components of software engineering is reusability, and in this case it is important to consider the contributions of the other game developers that use the same tool. Unity has an official documentation composed of source-code documentation, feature manuals and tutorials exemplifying the creation of different types of games. Furthermore, Unity has a community with easy find tutorials and developers eager to help with solutions.

StackOverflow is a well-known developers' question-and-answer [42] with a large number of easily-searchable questions on game programming [43]. As of November 2015 there were +2500 tags about Corona framework, +7300 about libGDX, +10400 about Cocos2D, and +13600 about Unity, as illustrated in Fig. 8.

F. *Performance*

Performance is another key factor considered by the developers when choosing a game development framework or engine. By using game development frameworks, developers can achieve better performance than a fully-featured game engine, since they generally do not provide so many canned game features. Some game development frameworks, such as libGDX, allow developers to access low-level programming APIs to squeeze the most performance from the underlying graphics technology – currently, either OpenGL or DirectX.

A good example of framework is libGDX, which focuses on avoiding garbage collection for Dalvik/JavaScript by careful API design and the use of custom collections. It also provides a single code base for all the platforms which greatly increases code maintainability (see Fig. 6). Additionally, it recommends using the Facade programming pattern to isolate platform-specific code, and provides utility methods to check the platform in which the code is being run.

The Unity AiO game engine leverages its GUI editor to allow developers to *drag'n'drop* game entities and components into the game scene, with the possibility of adding behaviours as C# or JavaScript code snippets. Unity also provides environmental variables to easily check the current platform, allowing the execution of platform-specific code. The GUI editor allows fast changes in the objects' positions or appearances, and uses the same code snippets for all the targeted platforms.

G. *Educational assessment*

Most frameworks and engines do not consider educational assessment. However, the serious games created with these tools can and should use additional services or tools to enable educational assessments and learning analytics.

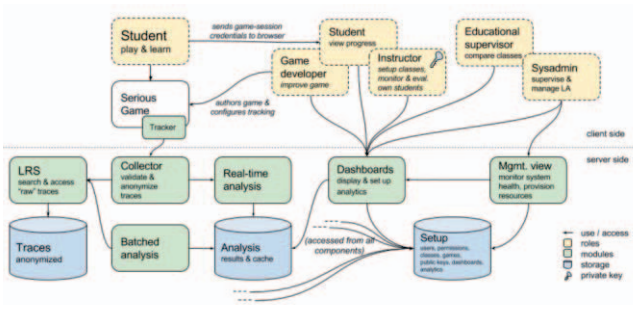

Student
play & learn
sends game-session credentials to browser
Student
view progress
Educational supervisor
compare classes
Game developer
improve game
Instructor
setup classes, monitor & eval. own students
Sysadmin
supervise & manage LA
Serious Game
Tracker
authors game & configures tracking
client side
server side
LRS
search & access "raw" traces
Collector
validate & anonymize traces
Real-time analysis
Dashboards
display & set up analytics
Mgmt. view
monitor system health, provision resources
Traces
anonymized
Batched analysis
Analysis
results & cache
(accessed from all components)
Setup
users, permissions, classes, games, public keys, dashboards, analytics
use / access
roles
modules
storage
private key

Fig. 9. Learning Analytics architecture at RAGE project. http://rageproject.eu/

Most game development frameworks and engines do not provide direct support for educational assessment. For example, libGDX has no direct support for any educational assessment mechanism. The developer must integrate any specific educational feature, which costs both time and resources. Developers using frameworks that lack the support for educational assessment mechanics should not develop the required features' infrastructure from scratch. As we can see in Fig. 9, developing a learning analytics infrastructure from scratch is an effort-consuming activity due to its complexity.

Instead it is recommended to integrate a service that provides a similar functionality out of the box. There are different analytics services available, each providing different features, and differentiated by target platform, cost, specific reports, data storage policies, and so on. Google Analytics is the de-facto industry standard for web and cloud-based analytics and dashboards. It provides a Rest API that can be used through all platforms but lacks official Software Development Kits on many platforms (e.g. Sony's PlayStation 4 or Microsoft's Xbox One). Google Analytics offers data visualization tools and the ability to define custom dashboards. Game Analytics is an alternative to Google Analytics that adds an extra layer of functionality and reports designed for games. Flurry Analytics offers similar features to Google Analytics, while eliminating certain thresholds. Although these analytics services were not created specifically for learning assessment, they can be adapted for this purpose.

Unity offers, in its Professional Edition license, the Unity Analytics Pro feature. Unity Analytics Pro gathers data from the game and transfers that data to cloud-based storage. The gathered data is processed, analyzed and sent to the Unity Analytics Dashboard. Like Google Analytics, it is very business-oriented, but can be adapted to provide a layer of learning analytics.

eAdventure was designed to create games with educational purposes in mind. It allows the developer to integrate specific educational features, such as adaptation profiles or evaluation profiles, within the game. Each profile is composed by a set of rules where a condition must be satisfied in order to trigger a series of effects. Developers can define different ways to evaluate or adapt the game in order to target the game to different kinds of contexts and public. Evaluation profiles can generate XML reports for automated processing or HTML reports for a game instructor or player. Adaptation profiles are executed when a set of external properties are satisfied, at the start of each game scene. An adaptation profile execution may change the content of the scene in terms of available objects, puzzle difficulty, etc.

Realising an Applied Gaming Eco-system (RAGE) is a project developing an open source infrastructure to deploy learning analytics and other educational assessment features. RAGE aims to develop, transform and enrich advanced technologies from the leisure games industry into self-contained gaming assets that support game studios at developing serious games easier, faster and more cost-effectively. The project assets will be available along with a large volume of high-quality knowledge resources through a self-sustainable ecosystem, which is a social space that connects research institutions, gaming industries, intermediaries, education providers, policy makers and end-users [REF]. One of these assets is the Learning Analytics asset, which allows teachers to monitor students and receive alerts and warnings depending on the student's game state, progress or score. The learning analytics asset provides a tracker client available in different programming languages, including JavaScript, Java and C#, which can be integrated with other frameworks and game engines such as libGDX and Unity.

V. CONCLUSIONS

The rise in economic relevance of the game industry has increased the number and quality of tools to create games. There is a large game development community and a broad spectrum of authoring tools, some aimed at the creation of specific games, such as eAdventure, for *point'n'click* games, and others designed to create a much broader variety of game genres, such as Unity.

While there are tools to create a wide range of different games genres, serious games have special requirements and there is still no proven and widely-accepted approach to create an effective SG. The developer must choose the best options depending on the game requirements, and considering factors such as the target platforms, desired interactions and mechanics, game genre, learning objectives, time constraints, and resources and supported technologies. Therefore, there are many aspects to consider and some of the technical characteristics are frequently neglected.

In this paper we have introduced and analysed the main aspects that a developer has to consider before choosing a specific tool to develop a game, or before choosing to develop a game at all. A game development framework might be preferred in cases where the developers already know the required programming language and the game requirements are aligned with the framework's features. Frameworks are chosen over game engines by developers that want to learn or exercise the core principles of game development and by small work teams. Developers with high time constraints may choose a specialized authoring tool in order to optimize the time spent developing the game. *All in one* (AiO) authoring tools might be the best option for most developers since they balance the amount of provided features, learning curve and targeted game mechanics. Unity is probably the most influential AiO tool, excelling in many of its provided features, and is able to compete with some *AAA* tools in some graphics features while, at the same time, being able to support multi-platform deployment and compete with game-development frameworks in terms of control over the game. However, Unity still requires programming.

Our analysis reflects that most of the available tools for creating games are conceived for the general game market and lack specific assessment features to analyse player behaviour or learning aspects. Only some specialized tools have these features (e.g. eAdventure) and those tools usually do not provide other relevant characteristics (e.g. multiplatform support).

Furthermore, the demand of SGs adapted to specific scenarios is growing, as we see teachers trying to integrate serious games within their classes and educational curricula. However, these teachers rarely have the programming skills that they would need to modify and adapt these games for different scenarios.

As final conclusion, one of the largest gaps in learning games is the lack of a software ecosystem that facilitates the creation of video games for educational purposes. There is also a lack of tools that can be easily integrated with the games in order to assess the learning outcomes through analytics while providing the main features listed in Section IV, such as exporting to several platforms, having a large quality community, efficiency and effectivity, etc. Through the different cases reviewed in this paper we have also identified the necessity of a software ecosystem that would facilitate the adaptation of serious games to specific scenarios and needs, such as allowing teachers to adapt existing games for their classes (using COTS) without programming knowledge. Some of those aspects are being currently addressed in the H2020 European project RAGE, which develops, transforms and enriches advanced technologies from the leisure games industry into self-contained gaming assets (modules) that support game studios at developing applied games easier, faster and more cost-effectively. A key and relevant asset is the entire infrastructure required to simplify the application of learning analytics in serious games.

Acknowledgments

The e-UCM research group has been partially funded by Regional Government of Madrid (eMadridS2013/ICE-2715), by the Complutense University of Madrid (GR3/14-921340), by the Ministry of Education (TIN2013-46149-C2-1-R), by the RIURE Network (CYTED 513RT0471) and by the European Commission (RAGE H2020-ICT-2014-1-644187 and BEACONING H2020-ICT-2015-687676).

References

[1] ESA, "Games: Improving the economy," pp. 7–9, 2015.

[2] "IGN - GTA 5 SALES HIT $1 BILLION IN THREE DAYS." [Online]. Available: http://www.ign.com/articles/2013/09/20/gta-5-sales-hit-1-billion-in-three-days.

[3] "Forbes - 'Grand Theft Auto V' Crosses $1B In Sales, Biggest Entertainment Launch In History." [Online]. Available: http://www.forbes.com/sites/erikkain/2013/09/20/grand-theft-auto-v-crosses-1b-in-sales-biggest-entertainment-launch-in-history/.

[4] S. Jarvis and S. De Freitas, "Evaluation of an Immersive Learning Programme to Support Triage Training," in *Games and Virtual Words for Serious Applications*, 2009, pp. 117–122.

[5] S. De Freitas, "Learning in Immersive worlds A review of game-based learning Prepared for the JISC e-Learning Programme," *JISC eLearning Innov.*, vol. 3.3, no. October 14, p. 73, 2006.

[6] E. A. Akl, V. F. Kairouz, and K. M. Sackett, "Educational games for health professionals," *... Database Syst Rev*, 2013.

[7] E. Brox, L. Fernandez-Luque, and T. Tøllefsen, "Healthy Gaming – Video Game Design to promote Health," *Appl. Clin. Inform.*, vol. 2, no. 2, pp. 128–142, 2011.

[8] S. Arnab, I. Dunwell, and K. Debattista, *Serious Games for Healthcare: Applications and Implications*, vol. 2. 2013.

[9] J. C. R. Jr, P. J. Lynch, L. Cuddihy, D. A. Gentile, J. Klonsky, and R. Merrell, "The Impact of Video Games on Training Surgeons in the 21st Century," *Arch. Surg.*, vol. 142, no. 2, pp. 181–186, 2007.

[10] T. A. Pempek and S. L. Calvert, "Tipping the balance: use of advergames to promote consumption of nutritious foods and beverages by low-income African American children.," *Arch. Pediatr. Adolesc. Med.*, vol. 163, no. 7, pp. 633–7, 2009.

[11] S. Cooper, F. Khatib, A. Treuille, J. Barbero, J. Lee, M. Beenen, A. Leaver-Fay, D. Baker, Z. Popovic, and F. Players, "Predicting protein structures with a multiplayer online game," *Nature*, vol. 466, no. 7307, pp. 756–760, 2010.

[12] M. D. Dickey, "Engaging by design: How engagement strategies in popular computer and video games can inform instructional design," *Educ. Technol. Res. Dev.*, vol. 53, no. 2, pp. 67–83, 2005.

[13] J. Kirriemuir and A. McFarlane, "Literature Review in Games and learning - Futurelab," *futurelab*, 2004. [Online]. Available: http://www.futurelab.org.uk/resources/publications_reports_articles/literature_reviews/Literature_Review378/.

[14] D. R. Michael and S. L. Chen, *Serious Games: Games That Educate, Train, and Inform*, vol. October 31. 2005.

[15] C. Dede, "Immersive Interfaces for Engagement and Learning," *Science (80-.).*, vol. 323, no. 5910, pp. 66–69, 2009.

[16] J. P. Gee, "What video games have to teach us about learning and literacy," *Comput. Entertain.*, vol. 1, no. 1, p. 20, 2003.

[17] T. M. Connolly, E. A. Boyle, E. Macarthur, T. Hainey, and J. M. Boyle, "Computers & Education A systematic literature review of empirical evidence on computer games and serious games," *Comput. Educ.*, vol. 59, no. 2, pp. 661–686, 2012.

[18] G. J. Hwang and P. H. Wu, "Advancements and trends in digital game-based learning research: A review of publications in selected journals from 2001 to 2010," *Br. J. Educ. Technol.*, vol. 43, no. 1, pp. 6–10, 2012.

[19] C. Perrotta, G. Featherstone, H. Aston, and E. Houghton, *Game-based learning: Latest evidence and future directions*. 2013.

[20] J. Chen, "Flow in games (and everything else)," *Commun. ACM*,

vol. 50, no. 4, p. 31, 2007.

[21] M. Mayo, “Video games: a route to large-scale STEM education?,” *Science (80-.).*, no. January, pp. 79–82, 2009.

[22] L. A. Annetta, J. Minogue, S. Y. Holmes, and M.-T. Cheng, “Investigating the impact of video games on high school students’ engagement and learning about genetics,” *Comput. Educ.*, vol. 53, no. 1, pp. 74–85, 2009.

[23] S. Barzilai and I. Blau, “Scaffolding game-based learning: Impact on learning achievements, perceived learning, and game experiences,” *Comput. Educ.*, vol. 70, pp. 65–79, 2014.

[24] M. T. Cheng, T. Su, W. Y. Huang, and J. H. Chen, “An educational game for learning human immunology: What do students learn and how do they perceive?,” *Br. J. Educ. Technol.*, vol. 45, no. 5, pp. 820–833, 2013.

[25] M. Papastergiou, “Exploring the potential of computer and video games for health and physical education: A literature review,” *Comput. Educ.*, vol. 53, no. 3, pp. 603–622, 2009.

[26] H. Tüzün, M. Yilmaz-Soylu, T. Karakuş, Y. Inal, and G. Kizilkaya, “The effects of computer games on primary school students’ achievement and motivation in geography learning,” *Comput. Educ.*, vol. 52, no. 1, pp. 68–77, 2009.

[27] J. Blow, “Game Development: Harder than you think,” *Queue*, vol. 1, no. 10, p. 28, 2004.

[28] P. Moreno-Ger, D. Burgos, I. Martínez-Ortiz, J. L. Sierra, and B. Fernández-Manjón, “Educational game design for online education,” *Comput. Human Behav.*, vol. 24, no. 6, pp. 2530–2540, 2008.

[29] R. Sandford, M. Ulicsak, K. Facer, and T. Rudd, “Teaching with Games,” *Comput. Educ. Educ. Gr.*, vol. 112, p. 12, 2006.

[30] R. Van Eck, “COTS in the classroom: A teacher’s guide to integrating commercial off-the-shelf (COTS) games,” *Handb. Res. Eff. Electron. gaming Educ.*, pp. 179–199, 2008.

[31] “ELTSandbox - videogames usage in classrooms.” [Online]. Available: http://eltsandbox.weebly.com/blog/category/lessons with games.

[32] R. Van Eck, “Digital Game-Based Learning: It’s Not Just the Digital Natives Who Are Restless,” *Educ. Rev.*, vol. 41, no. 2, pp. 16–30, 2006.

[33] “Gamelearn - Soft Skills Training Through Video Games.” [Online]. Available: https://game-learn.com/.

[34] “Classcraft - Make Your Classes Unforgettable.” [Online]. Available: http://www.classcraft.com/?utm_expid=68436248-15.WG4DGSkHTDqoAs056aTRRQ.0.

[35] “Duolingo - Learn a language for free. Forever.” [Online]. Available: https://en.duolingo.com/.

[36] V. Jašková, “Department of English Language and Literature Duolingo As a New Language-Learning Website and Its Contribution To E-Learning Education,” 2014.

[37] R. Vesselinov and J. Grego, “Duolingo Effectiveness Study,” *City Univ. New York, USA*, no. December 2012, 2012.

[38] “Citizen Science - Play it online!” [Online]. Available: http://citizenscience.gameslearningsociety.org/node/23.

[39] “Scratch - Imagine, Program, Share.” [Online]. Available: https://scratch.mit.edu/.

[40] M. Resnick, B. Silverman, Y. Kafai, J. Maloney, A. Monroy-Hernández, N. Rusk, E. Eastmond, K. Brennan, A. Millner, E. Rosenbaum, and J. Silver, “Scratch,” *Commun. ACM*, vol. 52, no. 11, p. 60, 2009.

[41] “e-Adventure e-Learning Games.” [Online]. Available: http://e-adventure.e-ucm.es/.

[42] Muenchen, Robert A., “The Popularity of Data Analysis Software,” no. April, 2014.

[43] C. Stanley and M. Byrne, “Predicting Tags for StackOverflow Posts,” *Chil.Rice.Edu*, pp. 414–419, 2011.