Human Computation (2016) 3:1:119-141

ISSN: 2330-8001, DOI: 10.15346/hc.v3i1.7

Improving Citizen Science Games through Open Analytics Data

JESSE HIMMELSTEIN Centre for Research and Interdisciplinarity (CRI), Paris Descartes University

RAPHAEL GOUJET Centre for Research and Interdisciplinarity (CRI), Paris Descartes University

TAM KIEN DUONG Centre for Research and Interdisciplinarity (CRI), Paris Descartes University

JASON BLAND Centre for Research and Interdisciplinarity (CRI), Paris Descartes University and U1101, INSERM, 75014 Paris, France

ARIEL B. LINDNER Centre for Research and Interdisciplinarity (CRI), Paris Descartes University and U1101, INSERM, 75014 Paris, France

---

ABSTRACT

Video game developers make heavy use of "metrics" (or "analytics") in order to understand the players' actions. This serves to improve the games by identifying weaknesses in their design, such as sections of the games that are too difficult, or never explored. While such data may be of great interest to scientists and the public at large, it is rarely, if ever, publicly available. Specifically, for scientific games, in which players contribute to the advancement of science through their actions in a game, learn scientific concepts, or both, this represents a missed opportunity for the public to learn from the data that they and their fellow players are generating. In this paper, we introduce RedMetrics, an open source solution to support both collection and open publication of game metrics for scientific purposes. Inspired by the belief that anyone can contribute to science, all data gathered by the service is freely and immediately available online. RedMetrics can gather data from any platform (web, PC, console, etc.) and store it on an open repository. The data is available via a web API as well as a web application. To ease integration, we provide interfaces for the popular game engine Unity as well as for the web browser. We demonstrate how RedMetrics can empower scientific game development through a case study of *Hero.Coli*, a game that teaches synthetic biology. We study the progression of players through the game and use RedMetrics to identify bottlenecks in the game design that hinder learning.

1. INTRODUCTION

In a “scientific game”, players learn scientific concepts and/or contribute to scientific discovery through their actions, such as analyzing data or manipulating simulations to find better solutions. Players can solve protein structure problems in Foldit (Cooper et al., 2010), reconstruct 3D neuronal models in EyeWire (Kim et al., 2014), and align DNA sequences in Phylo (Kawrykow et al., 2012), to name just a few successful games of this genre.

Tracking the behavior of players as they interact with scientific games can be useful for game developers, researchers, and teachers. Game creators like to better understand what players are doing in their games. Are players stuck on a particular level? Are they bored with it? Are they missing key bits of information? Researchers studying a scientific game would like to know how much players are contributing to new knowledge, as well as how much they are learning. Finally, when teachers integrate games into their classroom, they would like to understand how well the class is progressing in the game, and perhaps identify which students are performing well and which are not.

1.1 Game Metrics

Quantitative information about a user’s interaction with software is commonly known as “metrics”, and the analysis of raw data to provide answers to concrete questions is called “analytics” (El-Nasr, Drachen, & Canossa, 2013). In practice, the two terms are often used interchangeably to refer to the practice of gathering and analyzing quantitative data to help designers understand how the design of the game affects player behavior. In the remainder of the paper, we will use the term “metrics” to designate the data that is used to drive game development.

Metrics can be especially powerful when used within the context of A/B testing, an experimental setup in which different versions of a product are shown to different users in a random fashion (Kohavi, Henne, & Sommerfield, 2007). A/B testing provides an objective measure of which version performed best, depending on the criteria chosen. A notable example of A/B testing was during the USA presidential campaign of Barack Obama, in which the campaign team had to choose between several videos that would be shown to visitors of the campaign web site (Christian, 2015).

It is important to note that metrics is not the only source of information about player behavior. Designers often directly observe players interacting with their game, a practice called “playtesting” (Schell, 2008). During a playtesting session, designers may ask players to “think aloud” in order to understand what reasoning motivates a given player’s action (Denzin, Simon, & Ericsson, 1985). When coupled with the quantitative data provided by metrics, the qualitative information obtained by observing players can help provide the “why” to the quantitative “what”.

Here are some examples of questions that metrics can help resolve for a game creator:

***Demographics**: what audience uses my game the most? What audience has the most difficulty with it?*

***Engagement**: what percentage of players complete a large number of levels? What percentage only try once and drop it?*

***Progress**: which parts of my game bore the users? Which parts are too difficult?*

***Rewards**: does offering badges and points help in encouraging users?*

***Appreciation**: which parts of my game do users like or dislike the most?*

1.2 Open Data

Since metrics are primarily used in the game development process, the information is usually kept private to the developers. However, there is a clear advantage to publishing this data, so that other scientists, whether professionals or volunteers, can contribute to data analysis, put forth their own relevant hypothesis and questions. In other words, why not publish the metrics as open data?

Open data has become a worldwide movement to make previously private or unreachable data easily accessible via the Internet. The motivation for publishing the data varies, from governments bowing to the ideal of transparency (Janssen, Charalabidis, & Zuiderwijk, 2012) or companies extracting analysis from the public (Bennett & Lanning, 2007). Within the field of professional science, there is an increasing demand to publish both articles and the associated data in free online repositories. In general, it is considered in the best interests of both scientists and the public in order to spread scientific knowledge and increase the reproducibility of experiments (Steele, 2013).

1.3 Our Contribution

As we will show in the related work section, none of the existing game metrics systems that we are aware of are suitable for immediately publishing the data in an open manner that allows for anyone to analyze. We have created RedMetrics to fill that gap.

RedMetrics is an open source game analytics service, in which all data gathered is freely available online. RedMetrics can gather data from any platform (web, PC, console, etc.) and store the data in an open repository. After this point, any person, including the original game designers and the public at large, has immediate access to the data for analysis. RedMetrics therefore erases the boundary between what data is available to scientist and citizen, providing a base for a public citizen-science project in which non-professional scientists can analyze the same data available to the professionals.

The purpose of this paper is to describe the structure and use of a new infrastructure to support both collection and publication of game metrics for scientific purposes, not to report scientific results. We believe that our contribution will be of interest to scientists and game developers alike.The remainder of this paper is structured as follows: we will first discuss related work, and then the design behind RedMetrics. We will then present how RedMetrics was used to study learning for a citizen science game called *Hero.Coli*. Finally, we will conclude with challenges for future work.

2. RELATED WORK

Since RedMetrics lies at the intersection of game metrics and open data, we will first discuss previous work in metrics and open data, before examining popular game metrics systems.

2.1 Metrics as a Game Design Tool

Before the Internet era, it was difficult for games to send back information to their creators. The widespread adoption of “always-on” Internet has enabled game creators to receive a constant stream of information about how their games are being used. The same technology also allows designers to quickly update their software and observe the reaction, which empowers techniques such as A/B testing discussed earlier.

With the explosion of Internet-enabled devices such as smartphones and tablets, metrics are increasingly embedded in a large number of games and applications. In the game and web industries, metrics are commonly tied to monetization efforts as well, which gives them a core role in informing companies how best to market and develop their products.

Historically, game companies have created custom systems in-house (El-Nasr et al., 2013; Medler, John, & Lane, 2011), but there are a growing number of generic systems available to game developers. In general, they follow a client-server approach, in which the clients are games that send the data for storage in private repositories for analysis, which could be done on-the-fly or on-demand. The client-server architecture allows the metrics systems to be offered as services, in which the same server can be used by multiple games.

Integration with game engines is an attractive feature of these metrics systems. Most games rely on a game engine to provide features commonly found in games, such as graphics, sound, user input, and networking. Game engines might also include an extension system so that third parties can add even more functionality to the engine. To ease integration, game metrics services often provide libraries for popular programming languages and game engines.

2.2 Open Data in Games and Citizen Science

There is limited academic literature on games that openly publish the data they generate outside of the game itself, although multiple academics have identified the opportunity. (Devlin et al., 2014) propose that the analysis of metrics from any game could potentially be of scientific interest, "potentially making all games into scientific discovery games." (Bamparopoulos, Konstantinidis, Bratsas, & Bamidis, 2016) argue that it is imperative to open data gathered by exercise games so that researchers may evaluate the effectiveness of such data as a health-monitoring tool. Releasing the algorithms that calculate caloric expenditure and heart rate would also help to establish the validity of the data (Staiano & Calvert, 2011).

As an example of the potential of this open model, a software glitch in the online game World of Warcraft led to an epidemic-like phenomenon that researchers have been able to study as a model for real-word epidemics (Lofgren & Fefferman, 2007). The authors note the "exciting new direction for epidemiological research", as well as the "missed opportunity" that the game had not been designed to gather all relevant data up to research standards. Subsequently, researchers have intentionally added disease mechanics into other online games to reproduce this effect (Gittens & Greaves, 2015).

Finally, open data has also been used as the basis for game content, such as creating Monopoly boards based on economic indicators published by the UK government (Friberger & Togelius, 2012).

Some citizen science projects publish their data openly. A good example is eBird, a citizen science project to report global bird observations to a centralized open database (Iliff et al., 2008). The data has been used to study conservation, evolution, and biogeography (Wood, Sullivan, Iliff, Fink, & Kelling, 2011). Additionally, some platforms for crowdsourcing such as EpiCollect+ (Aanensen, Huntley, Menegazzo, Powell, & Spratt, 2014) or PyBossa[1] also publish their data in an open fashion, implying that any citizen science projects built upon the platform will in turn be publishing open data.

2.3 Game Metrics Services for Open Science

Now that we have discussed related work in metrics as a game design tool as well as open data in games, we would like to examine how appropriate existing metrics services are for our purposes of gathering and publishing game metrics for scientific analysis.

1 http://pybossa.com/

Metrics services vary with how they provide information back to game creators. Many do not offer access to the raw data gathered (although they likely store it), but rather provide tools to crunch the data and provide graphs. Depending on the needs of the project, not having access to raw data could prove problematic for a citizen science project. For one thing, different explanations and theories could be proposed for the same set of data. Providing access to raw data therefore offers the possibility to different teams of professional scientists to elaborate different interpretations and therefore deal with the usually enormous amount of generated data. Furthermore, raw data empowers citizen scientists to elaborate new hypotheses, adding a new level of citizen participation to the scientific game. Gathered data can also be used for different studies. For instance, one could design a citizen science project primarily targeted at studying cognitive science but also provide the same data for a sociological study of citizen scientists' engagement (Tenopir et al., 2011). Finally, access to raw data abides by the principles of open data, transparency, fact-checking, and reproducibility, as we already highlighted. Indeed, besides having greatly increased reliability, scientific games that abide by such principles can also rightfully claim to be ethical, and consequently foster the contribution of ethical-task-seeking users, which can represent a non-negligible part of users (Raddick et al., 2010).

To the best of our knowledge, there is no academic resource describing or comparing game metrics services. Perhaps this is due to the commercial nature of many of the services, or the fact that the landscape of available services is constantly evolving to match popular game platforms, languages, and infrastructures. But in interest of placing our work within the state of the art, we have gathered a list of metrics services that we consider "popular", in that a game developer would likely find out about them with a single web search at the time of this writing. In the following table, we point out the business model, the platform that the clients integrate with, and the availability of raw data for scientific analysis.

Name	**Model**	**Client Integrations**	**Raw Data Availability**
Honey Tracks	Paid	Unity, Flash, PHP, Ruby, Java, JavaScript, IOS	2 weeks of raw data available
GameAnalytics	Free and commercial	Unity, iOS, Android, Xamarin	6 months of raw data available
deltaDNA	Free and commercial	Unity, iOS, Android	31 days by default, indefinitely extensible
SOOMLA	Free and commercial	Unity, Cocos2d-x	Not explicitly defined
Unity Analytics (beta)	Free and commercial	Unity	Yes, but limits unknown
Ninja Metrics	Paid	JavaScript, Java, PHP, iOS, Android, Unreal	No

Table 1 Existing game metrics services

As the previous table reveals there is currently no game analytics service that is open source and lets the game creator control what happens to the data that they gather. In addition, raw data is not always available.

Another possibility for game creators is to turn to web analytics, which is a wider market with more services available. Web analytics splits up a web site by page and analyzes the path that users take across the web site. For web-based games, it is possible to use a web analytics system by sending events to these services. Furthermore, the Piwik service is open source and can be self-hosted to provide access to the raw data (Miller, 2012). On the other hand, web analytics tools were built to handle web interactions and do not provide game-specific notions such as levels and positions. Although this approach is possible, the difficulty it poses implies that many citizen science game creators will stick with commercial game analytics services or abandon metrics altogether.

2.4 RedMetrics within the Related Work

In the previous sections, we have seen how game metrics have become an established practice for game development, and how open data has much to bring to scientific research, especially in the context of citizen science projects. We have also discussed how any game, even those strictly designed for entertainment, can be a potential source of scientifically relevant data.

Yet to our knowledge there are no existing game metrics systems that bridge this gap to provide open data gathering and publishing for scientific purposes. Our contribution is to provide a turnkey solution for scientific open data gathering that integrates with existing platforms for game development.

Next, we will explain how RedMetrics is built and how it gathers and organizes data from games.

3. REDMETRICS ARCHITECTURE

RedMetrics[2] is the name of our open source game analytics service that publishes all of its gathered information as an open data public repository. There is a great variety of platforms (mobile, web, desktop, and consoles) and programming languages used in modern game development. In order to allow any internet-connected game to use RedMetrics, we chose to build it as a web service. A web service provides an API that is accessible using web technologies, such as the HTTP protocol for communication and JSON or XML for encoding data (Booth et al., 2004). Unlike other Internet protocols, HTTP is rarely blocked by proxies.

---

2 http://redmetrics.io

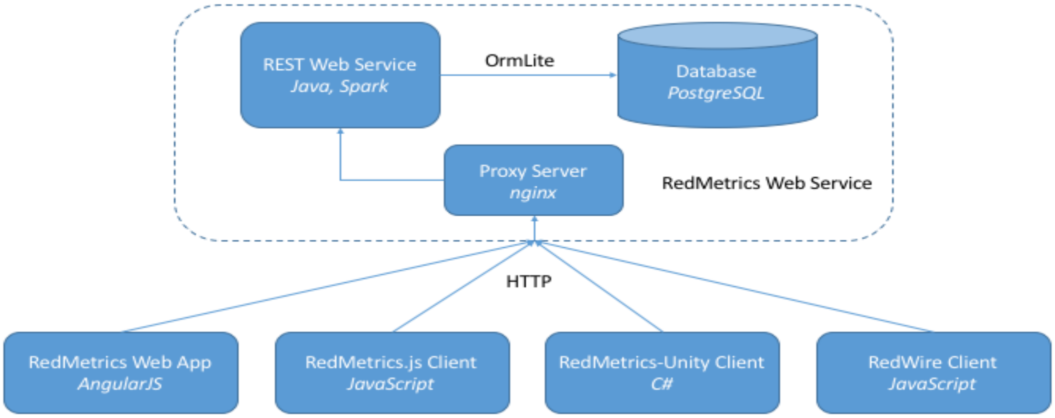

Figure 1. RedMetrics Architecture. Clients (bottom row), written in various languages, connect over HTTP to the Web Service (top section). The service is itself composed of a proxy server which reroutes to a REST web service written in Java. In turn, this web service stores and retrieves data in a PostgreSQL database.

We have built a web application as a convenience for users to view, filter, and download the metrics data. Furthermore, the web application provides administration features such as the addition of new games and game versions into the system.

Finally, there are the games themselves. To capture the data, developers need to insert "hooks" into their games that log key events and send them to the server. The data can be sent at whatever rhythm best suits the game (e.g. between levels of a game, or during an idle period). We have built 3 clients for RedMetrics: one for web games using JavaScript, one for the popular game engine Unity, and one for the experimental game engine RedWire (Himmelstein et al., 2014).

We first discuss how the web service is built before examining the clients.

3.1 Web Service Concepts

The web service is built around a few basic concepts. *Events* represent something happening at a particular point of time. Examples include starting a level, pausing the game, gaining points, or quitting the game. Events typically are specific to a particular game, although certain types of events are commonly found across games in the same genre.

Snapshots, on the other hand, contain of a representation of the game state at a given point of time. This could be the position of the mouse, or the number of points the player has gained so far, or the positions and health of enemies that the player is facing. Snapshots can be used to replay what a user saw and did during a game session. Snapshots can also be used to represent continuous recording of the player, such as eye- or emotion-tracking data.

RedWire allows the developer to send both events and snapshots, which complement each other to provide a complete picture of how players interact with their game.

In addition, RedWire stores demographic information on players, but is careful to avoid any kind of personally identifiable information. It does allow researchers to associate players with external IDs that can be referenced to external databases. In this way, metrics data can remain open and free for all to use, without endangering user privacy or holding back the researcher's ability to use the system for a scientific study.

Finally, RedWire supports game versioning, allowing developers to compare how player interaction changed in reaction to their modifications to the game. This feature also facilitates A/B testing.

3.2 Web Service Protocol

All communication between the server and client applications is done via JSON by default, although the client can request data in different formats (such as CSV). Following the RESTful architecture pattern (Fielding, 2000), GET is used only to request data, POST creates resources, PUT replaces or updates them, and DELETE removes them. To permit browsers to connect to an external web service, cross-domain headers are included in all web service responses.

RedMetrics provides 5 types of resources: games, game versions, players, events, and snapshots. Events and snapshots have similar attributes, including: user and server timestamps, type, section, coordinates, and custom data. All but the last attribute can be used for searching, and all but the first are optional. The *type* is a name such as "start", "quit", or "gainPoint". The *section* is a logical portion of a game, split into a hierarchy using dots, such as "level1.partB". The *coordinates* refer to 2D or 3D coordinates in a level, useful for games in which the player controls an avatar moving through a space.

The player resource only has attributes for an approximate birth date (month and year), a region and country, a gender, and an external ID. All attributes are optional.

When metrics are being used to track detailed information about a player's interaction, a great deal of data can be accumulated. Sending the data continuously can take up valuable CPU resources and slow down the gameplay. RedMetrics supports sending data in a bulk transfer. Under this scenario, it is important for developers to generate timestamps for their events and snapshots on the client side.

3.3 Web Application

The web service API provides an interface for teachers and researchers to download usage information about their application. But sending HTTP requests by hand is not a fantastic user experience. We therefore created a web application that provides a visual interface for filtering and downloading data from the web service in CSV format that can be directly plugging into common statistics software.

The web application allows researchers to search for the data that they are looking by establishing filters. A researcher can filter by game, game version, player, type, section, and time.

In addition, the interface provides a number of administration features, including adding game and game versions.

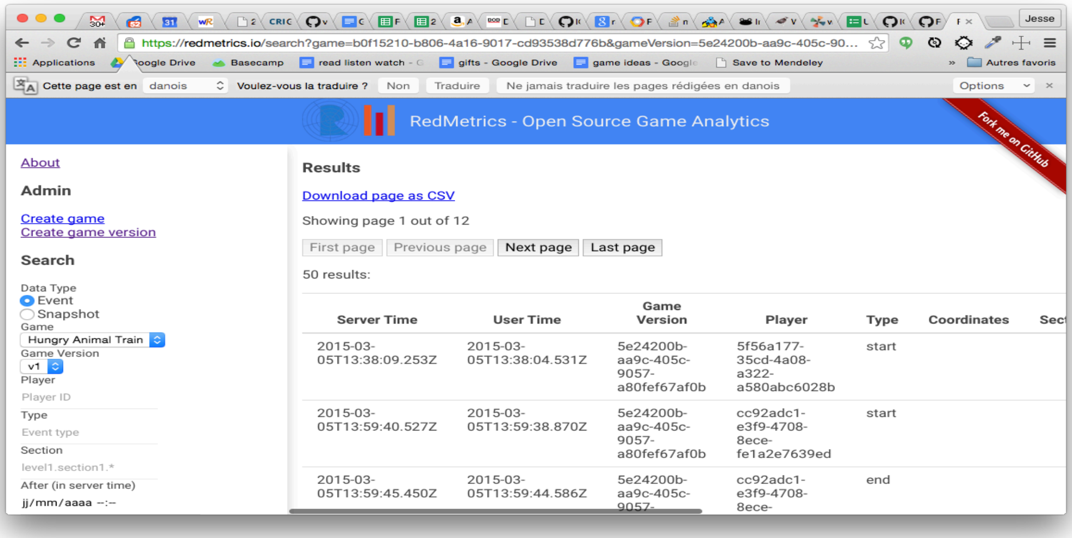

Figure 2. RedMetrics web application showing search results.

3.4 Game Clients

Strictly speaking, any developer can use the RedMetrics web service simply by sending HTTP requests. However, there are a number of common tasks, such as serializing JSON or batching requests, which can be eased by the use of an existing library. In addition, client libraries can handle integration with a specific game engine, doing some work automatically.

We have created three such integrations. The first is a generic JavaScript client for use in browser-based games, called *RedMetrics.js*[3]. It does not integrate with a specific engine, but can be used with any browser-based game. The second is *RedMetrics-Unity*[4], which integrates with the popular multi-platform game engine Unity. The third integrates with *RedWire*[5], a browser-based and open-source game engine for remixing games (Himmelstein et al., 2014). This integration is the most advanced, because it automatically takes snapshots of the game state throughout a play session.

3.5 Implementation

RedMetrics is an open source project whose code is under an MIT open source license. It is programmed in Java, on top of a PostgreSQL database. PostgreSQL has been particularly useful for its advanced features such as ltrees, which provide a way to quickly index hierarchical data such as the section within a game.

The clients are available as separate repositories, each using the MIT license. The web application is developed entirely in JavaScript, using AngularJS and Material Design components. The Unity client is written in C# and can be integrated as easily as any other Unity package. The RedWire client is integrated into the engine itself, and written in JavaScript.

3.6 Privacy Issues

When gathering data about users, it is important to adhere to a privacy policy that respects a user's rights. We have designed RedMetrics to not store identifying information about players. For example, the birthday is rounded to the nearest month, and the location to the nearest region. But as RedMetrics is capable of storing arbitrary data about users, it could be abused by game developers to store identifying information nonetheless.

For this reason, we have established a privacy policy that developers must agree to that they will not store identifying information about players. This approach is shared by other analytics tools such as Google Analytics.

3 https://github.com/CyberCRI/RedMetrics.js

4 https://github.com/CyberCRI/RedMetrics-Unity

5 https://redwire.io/

4. A TEST CASE FOR REDMETRICS: HERO.COLI

In order to test RedMetrics and exemplify its usefulness within the popular Unity game engine, we used a novel citizen science learning game of our own development, *Hero.Coli* as a textbook case of the kind of application RedMetrics has been developed for. We demonstrate here the role of RedMetrics in guiding game development, in studying learning and potentially performing citizen science.

Hero.Coli is a video game that aims to teach synthetic biology[6]. It is built with the Unity game engine to guarantee professional-level development pipeline: easier coding and deployment on the developer side, and best accessibility and performance on the player side. This game differs from classic learning games, which involve a phase where players are directly provided with text-based or video-based content, which players are then required to learn so that they can be tested using a multiple-choice quiz. Beating this challenge usually unlocks a reward in such games: either a vanity reward such as points or badges, such as in Influent, a learning game for foreign languages (Culbertson, Andersen, White, Zhang, & Jung, 2016), or a previously locked game element such as a level, a tool, a character, or a game mode. *Hero.Coli* uses stealth learning (Sharp, 2012) instead of this explicit learn-test-reward loop. In this way, the learning materials from *Hero.Coli* are directly integrated into the gameplay with no abstract step of knowledge evaluation, as this evaluation is integrated into the game flow itself.

Hero.Coli is a single-player 2D top-down adventure game where the player controls a bacterium to explore a fantasy aquatic world (**Figure 3**). The user directs this bacterium, to collect and combine functional DNA fragments in order to engineer and control its abilities. More precisely, the user can craft and equip genetic devices, based on real genetic circuits, that provide the bacteria with functionalities (phenotypes) that allows it to overcome the games' challenges.

The core target audience consists of biology students aged 17 - 25, and should complement their studies and provide them with a quick and cheap way to experiment. As this game is also intended to provide a learning platform for the general public, it is also designed to be accessible for children, teenagers and adults. In the future, this whole audience will also be invited to participate in citizen science on synthetic biology, the same way FoldIt and EteRNA did on molecular biology (Cooper et al., 2010; Lee et al., 2014), but by combining real BioBricks of the BioBricks Foundation[7] to create new genetic systems instead of determining the spatial structure of proteins and RNA.

---

6 http://herocoli.com

7 http://biobricks.org/

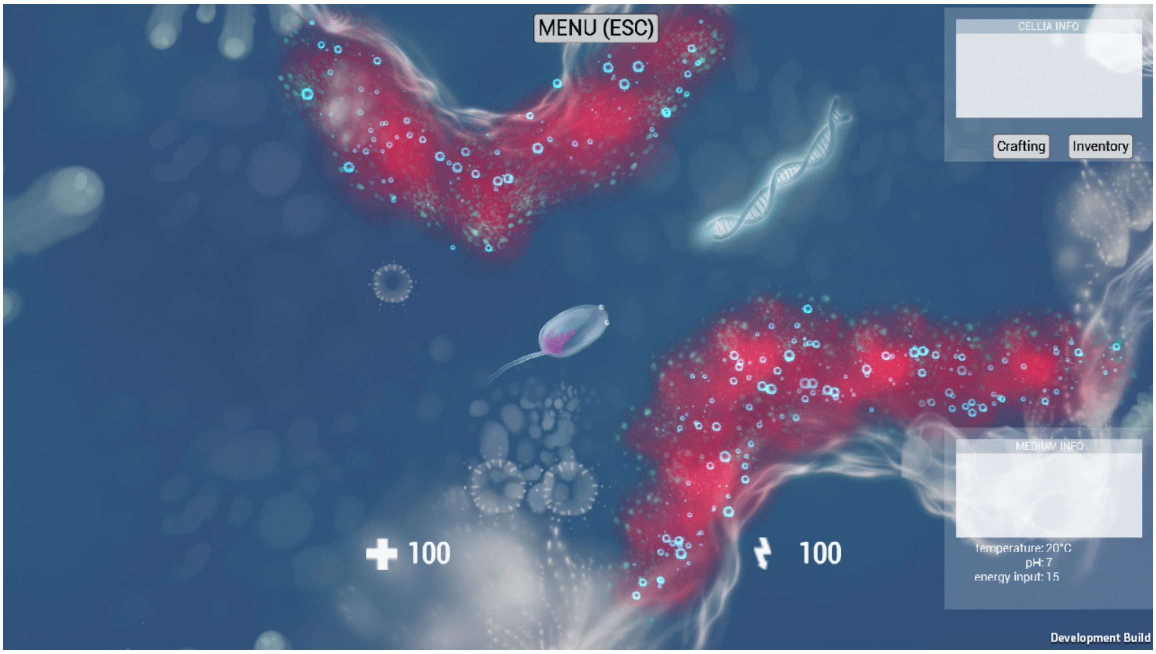

Figure 3. A screenshot of *Hero.Coli*, which demonstrates its top-down gameplay.

In order to improve the game experience and the learning efficiency, we relied on several complementary sources of information. Key factors such as direct quiz scores, occurrence of defined game events (*e.g.*, localization, genetic devices present in the bacterium *etc.*). This was achieved by organizing playtests and by tracking players' actions through the integration of RedMetrics into *Hero.Coli*.

4.1 Integration of RedMetrics

Players' performance was measured through statistical usage data, gathered using the RedMetrics service. Events are logged through RedMetrics any time a user advances or carries out an action in the game. This includes reaching a checkpoint *i.e.* a place in the game where the character re-spawns after dying ('reach'), or changes the game settings from the control panel ('configure'). Other logged events are "start", "pickup" (picking up DNA), "equip" (inserting DNA into the bacterium) and its opposite action "un-equip", "death", "craft" (when users equip or craft a "genetic device", *i.e.* when they correctly assemble BioBricks to create a functional genetic system), "switch" (switching game mode), "restart", or "complete" (beating the game).

To gather data about a specific event of the game, a line of code that invokes *RedMetrics-Unity* to sends data to the RedMetrics server has to be added to the relevant part of the game's code that manages the said event. This call uses the following parameters: a tracking event key that may be generic to any video game ('start') or specific to *Hero.Coli* ('craft'), and optional custom data which usually are parameters of the event that just occurred, for instance the BioBrick sequence that was crafted. A few other parameters are automatically added, which indicate the game session ID, the server and user time at which the event was logged, the coordinates of the character in the game and the section of the game map. This meta-information about the event helps establish why the event occurred when it did.

To highlight trends and cohorts of users, the gathered data was downloaded from RedMetrics and fed to an open-source Python script[8]. The script directly takes the CSV file downloaded from RedMetrics as input, and outputs a series of graphs and statistics on logged events. It is easily adaptable to any other RedMetrics-tracked project as the studied events (which is the only *Hero.Coli*-specific aspect) are structured as an editable parameter list in the script.

While coding this script, it immediately appeared that *Hero.Coli*'s lack of login system to identify unique users was a real hindrance to the project. Only game sessions can be considered instead of users. It means that long-term evolution of player behavior, including engagement, has to be inferred instead of being directly measured.

4.2 Game Duration and Completion

We measured the advancement of players along the game's levels, in order to get feedback on their difficulties. To this end, relevant *Hero.Coli* events were recorded when players reached various checkpoints placed regularly along the game path (Figure 4).

In first implementation (450 sessions recorded from May 20th to November 17th, 2015) we found that a considerable proportion (35%) of game sessions do not reach the first checkpoint, 25% of players stop playing before reaching the second checkpoint, 15% stop playing between the second and the third checkpoint (see ***Figure 5***). This is the "funnel effect", well-known by online game developers. The costlier it is for a user to perform a task in terms of complexity and time, the smaller the cohort of users that successfully perform the task. Complementary issues can be causative as technical difficulty, long loading time, low appeal, etc.

8 https://github.com/taniki/herocoli-analytics and https://github.com/CyberCRI/dataanalysis-herocoli-redmetrics

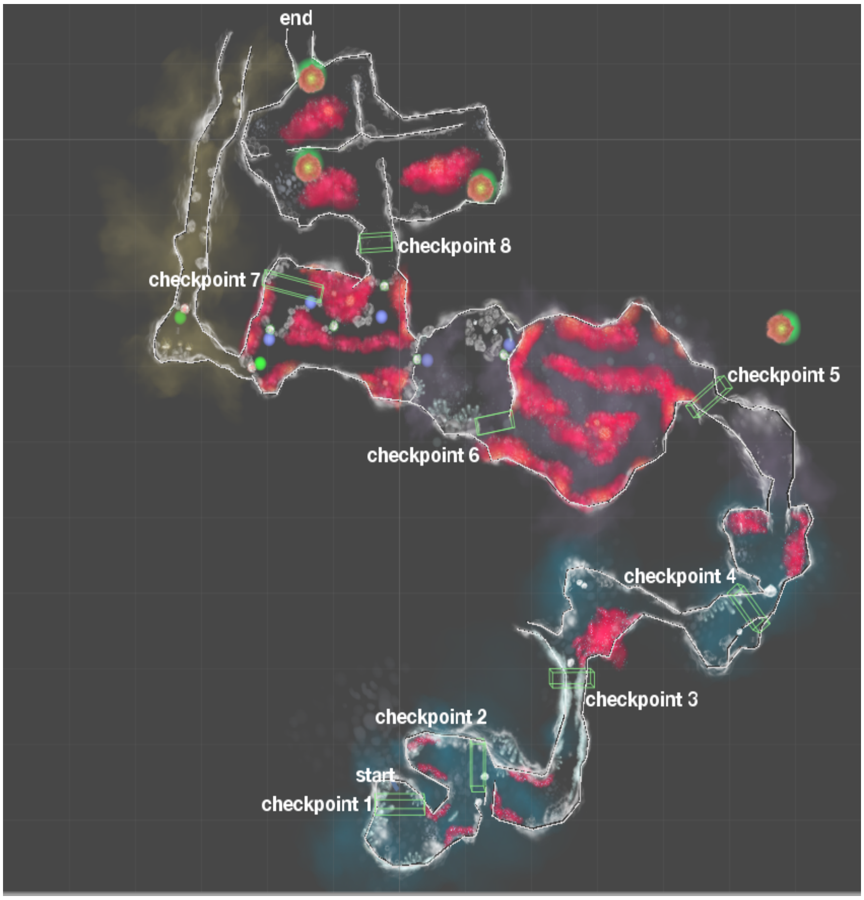

Figure 4. Checkpoints on the *Hero.Coli* map in adventure mode

This lead us to concentrate our efforts on making the game more accessible in terms of platform – getting a smooth-running game – and in terms of difficulty. Too many players quit the game early and could not achieve much in the game, due to a lack of understanding of the interface, mechanics, and scientific content as the playtest showed. Therefore, our next development effort was focused on simplifying level design and game mechanics, making the interface more intuitive, and proposing step-by-step tutorials and cut scenes demonstrating game mechanics.

Another interesting aspect of ***Figure 5*** is the rise on the last checkpoints: it shows that the game was appealing enough so that gamers would rather finish the game than quit. On the other hand, the part of the game where users quit the most is right at the beginning, until the third checkpoint.

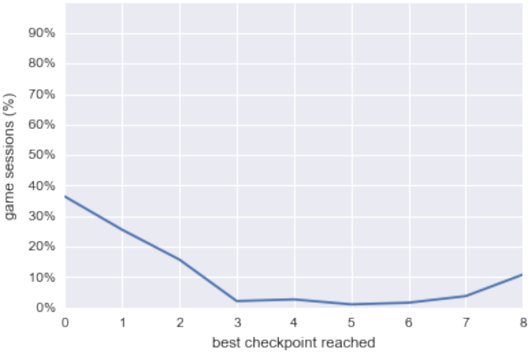

Figure 5. Furthest checkpoint reached by users

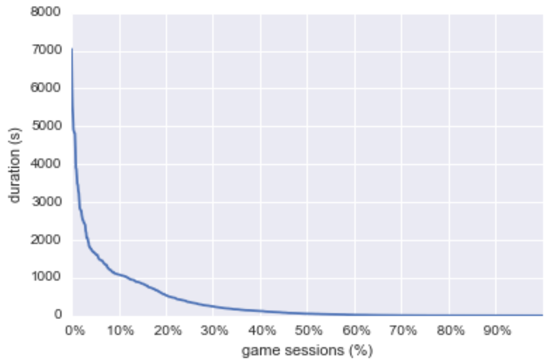

Figure 6. Time spent playing ***Hero.Coli*** by all users

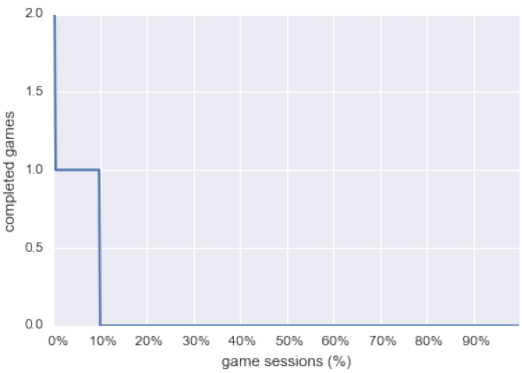

Figure 7. Number of game completions during game sessions

A second proxy for engagement is the duration of the game sessions (***Figure 6***). It shows again that a small proportion of game sessions lasted much longer than the time required to finish the game (around 10-30 minutes) and therefore demonstrates the appeal of the game and engagement of users even over the span of one game session. The final stage of the funnel can be observed when analyzing the proportion of game sessions that sent a "game completion" event (***Figure 7***). Only around 10% of game sessions ended up with a game completion, with a maximum of 2 completions of the same level achieved by a negligible number of game sessions.

4.3 Activity and Engagement

At this point, we knew that the game was engaging enough to keep 10% of its players to play at least 1000 s, and 10% to reach the last checkpoint. What needed to be known was which actions were easily accomplished by the player, and which were not.

Player engagement and participation can be measured by several statistical metrics. Two of these are the number of logged events in general and of player deaths in particular. ***Figure 8*** shows the total number of logged events for the 450 sessions recorded from May 20th to November 17th, 2015.

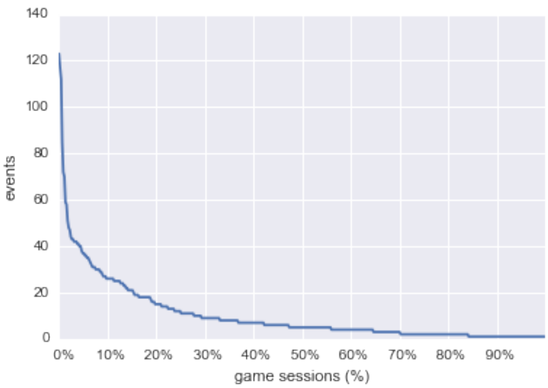

Figure 8. Number of events recorded during game sessions

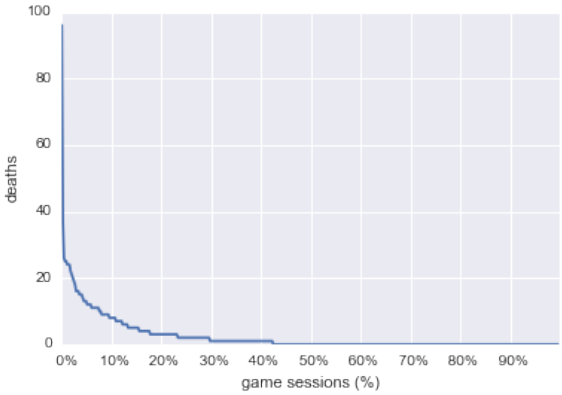

Figure 9. Number of deaths recorded during game sessions

This data shows that a small percentage (15%) of game sessions recorded more than 20 events, which would indicate a significant commitment to playing the game. Additionally, the number of deaths can also show game commitment (***Figure 9***). The 450 game sessions recorded 1135 deaths. Many users stopped playing the game after never dying or dying a single time, while only a few users died over 10 times. Those players who continued playing after dying can be considered to have a greater interest in the game. These two statistics complement each other, as they show similar usage trends. The maximum value (96 deaths) is due to 1 game session only that lasted 1 hour, the second maximum being 37 deaths.

The same trend appears on genetic device usage. In *Hero.Coli*, the players first pick up a genetic device and then insert it into their bacterium to have it work, an operation labelled as "equipping" in the game to parallel Role-Playing Games, and finally they can craft new genetic devices. Each subsequent operation is longer and more complex than the previous one. In the graphs below (**Figure 10**), the cohort of successful players starts at just over 50% for those who picked up a genetic device, then it stays stable at 50% of game sessions in which a genetic device was equipped, and finally, the proportion of successful game sessions falls to 20 %.

This means that the very basic operation of picking up a genetic device was very easily accomplished by users. The fact that the proportion of game sessions in which a device was equipped is close to the proportion of game sessions in which a device was picked up even though equipping is costlier than picking up can be explained by the sandbox game mode. In this mode, contrary to the adventure mode, the inventory of genetic devices is not empty when the game starts. The user can then equip devices right from the start of the game while exploring its features. This reasoning also implies that, conversely, some players were able to pick up devices but were unable to equip them.

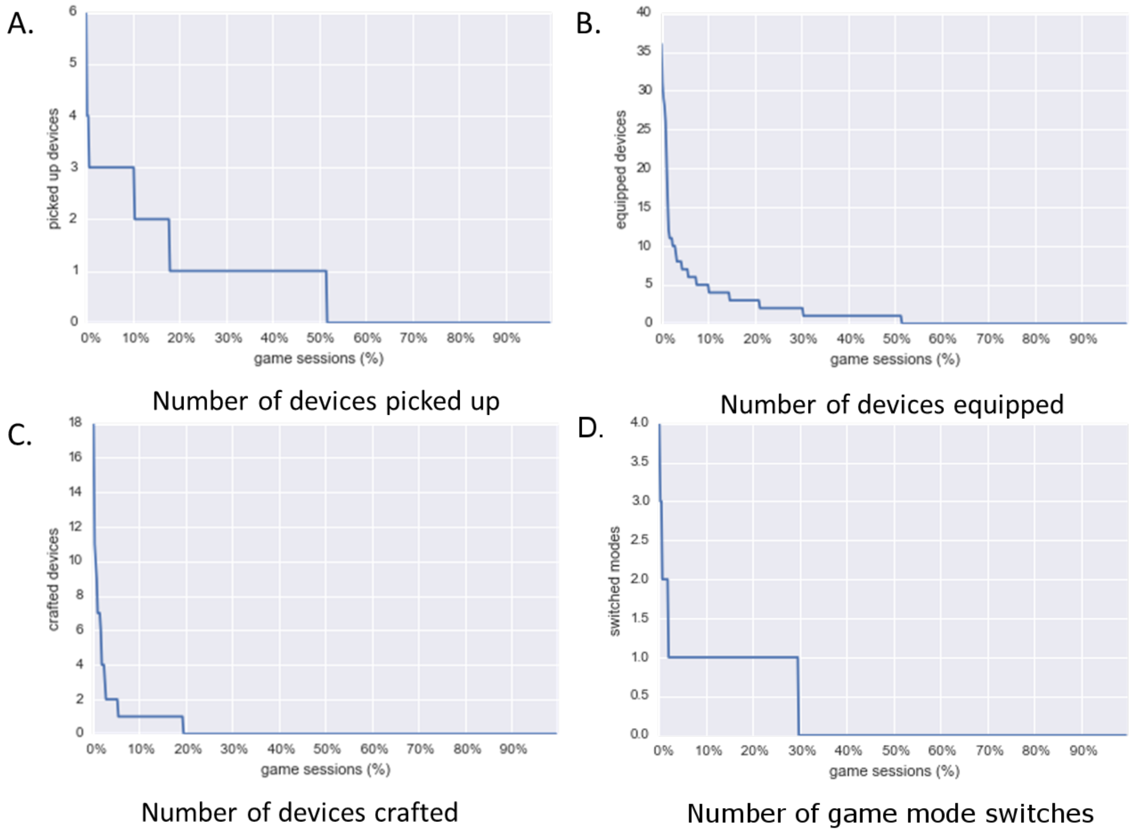

Figure 10. Device activity and game mode switches in *Hero.Coli*

We also found out that very few players demonstrated creative behavior in their gameplay. It appears that the majority of devices crafted were those required by the game to reach the exit of the maze in the adventure mode. Only a handful of them are devices created in the sandbox mode, among which only a small part are novel devices. This lack of experimentation could be linked to problems with the crafting interface itself as previously highlighted. But it could also be linked to the low number of phenotypes and mechanisms shown in the adventure mode, and to the limited content available in the game overall (only 25 available BioBricks, compared to the thousands referenced by the BioBricks Foundation). That is why in addition to the redesign of the crafting interface, we added even more bricks in both adventure and sandbox modes.

Figure 10 also shows that during 30% of game sessions, the game mode was switched from adventure to sandbox mode. This shows that this functionality was well understood and userfriendly. This also supports our hypothesis that some users never picked up devices because they switched modes early on.

Finally, by plotting game events against their location in the game map, level design issues can be spotted (**Figure 11**). The resulting hotspots of difficulty – zones with numerous black dots indicating several "deaths" events, or zones with irrelevant events – can then be redesigned.

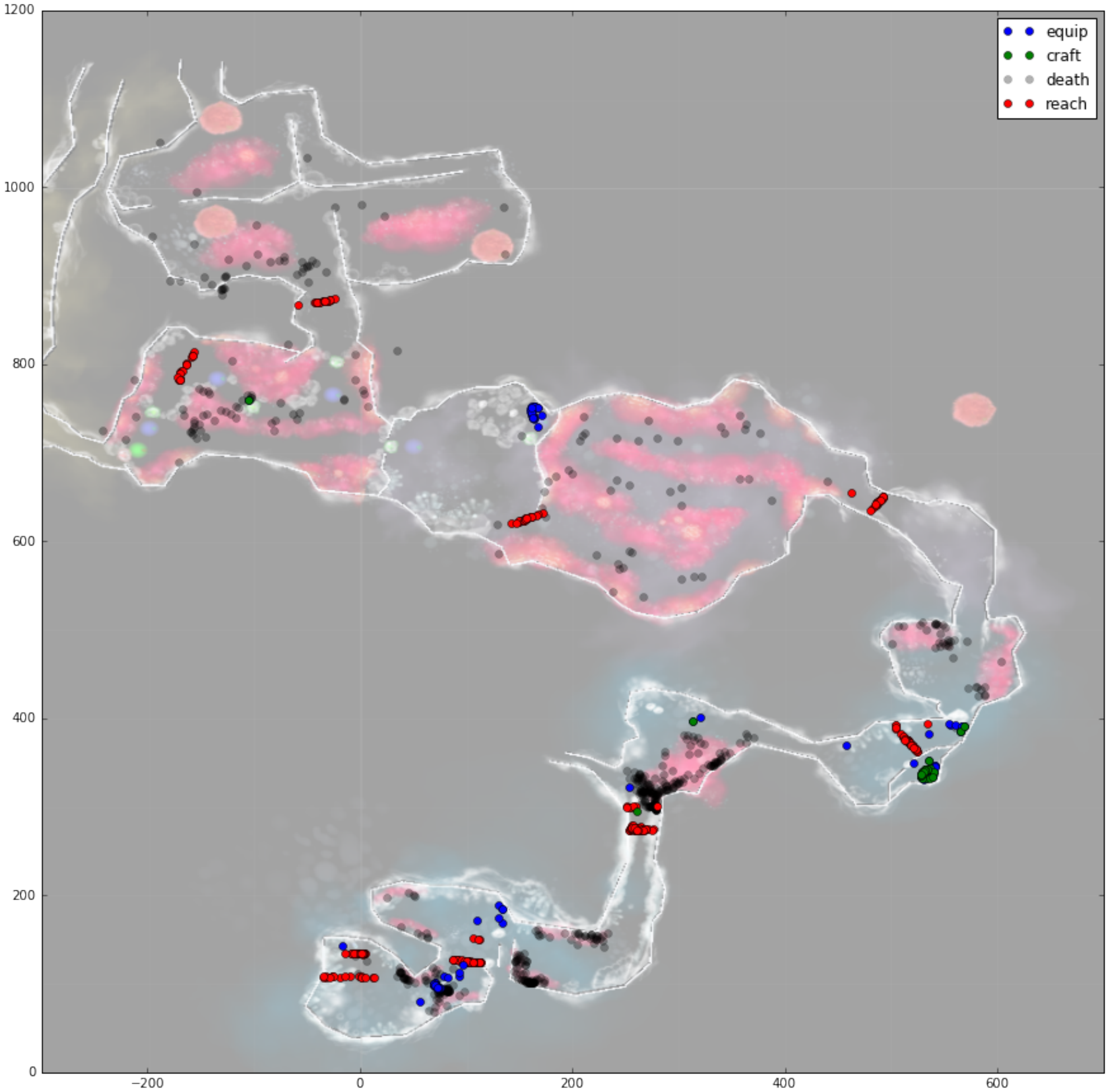

Figure 11. Location of events

4.4 Integration Results

What proved to be fruitful in RedMetrics was first the data format which enabled us to easily use Python scripts to analyze them. Logged events could quickly be integrated into the pipeline of Python functions that filter, aggregate, transform, and plot them. Moreover, the flexibility of the data format – the fact that new, custom data can be added by the game developer – enabled us to progressively increase the complexity of events being sent to the RedMetrics server, resulting in a finer and finer analysis of players' behavior, meeting our increasing requirements. This flexibility also enabled us to send information about very different aspects of the game, be it DNA sequences, user information, or game configuration.

Using RedMetrics during recent playtesting sessions also proved informative and made them even more fruitful. What players and what data told us sometimes differed. Data helped us discriminate red herrings we focused on from what really needed to be addressed. It also helped us quantify instead of relying on general impressions of the playtesting session. Online questionnaires finally helped us see the difference between gaming performances and learning outcomes. We are now planning new playtesting sessions in 2017 in which RedMetrics and online questionnaires will be used in conjunction to try and identify the sources of (un)successful learning processes.

Finally, in order to improve our understanding and analysis of the game, we needed to have data on mid- and long-term usage of the game. That's why we have also developed a player login system, which will help us distinguish unique players. Instead of studying game sessions, which are no longer than a couple hours of use, we can now see how players go back to the game and evolve in their use. Such a user identification system will deepen our understanding of the long-term learning curve, retention, and engagement. It will also help us act directly on the player by allowing vanity rewards, personal statistics, top scores and leaderboards, which can be strong incentives to continue playing.

What could not be evaluated in the scope of applying RedMetrics to *Hero.Coli* is the impact of open data and open source. There is still no community of contributors – developers or scientists willing to analyze our data. But this openness still offers the possibility of reproducibility and sustainability through community involvement. We have not stress-tested RedMetrics either, but in terms of performance, the metrics gathering did not slow down the player experience.

5. CONCLUSION AND FUTURE WORK

In this paper, we have expressed the need for an open data game analytics service for citizen science games that can be useful for the development of other games as well. We have described the design and implementation of RedMetrics, our open source system that already integrates with some of the existing game platforms and is open to all of them. Finally, we have detailed how RedMetrics was used to understand player behavior in the citizen science game *Hero.Coli*, and to inform design decisions. The data provided by RedMetrics can help improve game design as well as identify learning patterns within games.

Continuing their symbiotic relationship, *Hero.Coli* will continue spearheading the development of the Unity plugin for RedMetrics. In the near future, besides improved analysis of engagement and retention, RedMetrics will allow us to analyze creativity and learning within *Hero.Coli* and therefore help us understand the effect of our citizen science game.

In future work, we plan to improve RedMetrics upon three fronts: performance, integration, and ease of analysis. Sending snapshots of large game states has proved to be a significant demand on network resources. We can remedy this problem by only sending updates to the previous snapshot, using classic *diff* and *patch* algorithms (Hunt & McIlroy, 1976). Integrating with other popular game engines such as Unreal could boost adoption among game developers. Finally, by expanding the feature set of the web application to support common tasks such as cohort analysis and A/B testing, it would be easier for citizen scientists and developers alike to benefit from the open data provided.

6. ACKNOWLEDGEMENTS

This research was funded by the EU project Citizen Cyberlab (Grant No 317705).

REFERENCES

Aanensen, D. M., Huntley, D. M., Menegazzo, M., Powell, C. I., & Spratt, B. G. (2014). EpiCollect+: linking smartphones to web applications for complex data collection projects. *F1000Research*, *3*(0), 199. http://doi.org/10.12688/f1000research.4702.1

Bamparopoulos, G., Konstantinidis, E., Bratsas, C., & Bamidis, P. D. (2016). Towards exergaming commons: composing the exergame ontology for publishing open game data. *Journal of Biomedical Semantics*, *7*(1), 4. Retrieved from http://www.jbiomedsem.com/content/7/1/4

Bennett, J., & Lanning, S. (2007). The Netflix Prize. *KDD Cup and Workshop*, 3–6. http://doi.org/10.1145/1562764.1562769

Booth, D., Haas, H., McCabe, F., Newcomer, E., Champion, M., Ferris, C., & Orchard, D. (2004). Web Services Architecture. *Group*, *22*(February), 19–26. http://doi.org/10.1023/B:BTTJ.0000015492.03732.a6

Christian, B. (2015). The A/B Test: Inside the Technology That's Changing the Rules of Business. *Wired Magazine*.

Cooper, S., Khatib, F., Treuille, A., Barbero, J., Lee, J., Beenen, M., … Players, F. (2010). Predicting protein structures with a multiplayer online game. *Nature*, *466*(7307), 756–60. http://doi.org/10.1038/nature09304

Culbertson, G., Andersen, E., White, W., Zhang, D., & Jung, M. (2016). Crystallize: An Immersive, Collaborative Game for Second Language Learning. In *CSCW*.

Denzin, N. K., Simon, H. A., & Ericsson, K. A. (1985). *Protocol Analysis: Verbal Reports as Data. Contemporary Sociology* (Vol. 14).

Devlin, S., Cowling, P. I., Kudenko, D., Goumagias, N., Nucciareli, A., Cabras, I., … Li, F. (2014). Game intelligence. In *IEEE*

Conference on Computatonal Intelligence and Games, CIG. http://doi.org/10.1109/CIG.2014.6932917

El-Nasr, M. S., Drachen, A., & Canossa, A. (2013). *Game Analytics: Maximizing the Value of Player Data*. Retrieved from https://books.google.com/books?hl=fr&lr=&id=-guHadEPrFcC&pgis=1

Fielding, R. T. (2000). Architectural Styles and the Design of Network-based Software Architectures. *Building*, *54*, 162. http://doi.org/10.1.1.91.2433

Friberger, M. G., & Togelius, J. (2012). Generating game content from open data. *Proceedings of the International Conference on the Foundations of Digital Games - FDG '12*, 290. Retrieved from http://dl.acm.org/citation.cfm?id=2282338.2282404

Gittens, C., & Greaves, J. (2015). Transforming BrowserQuest into an epidemiological tool for modelling disease dissemination. In *Proceedings of CGAMES 2015 USA - 20th International Conference on Computer Games: AI, Animation, Mobile, Interactive Multimedia, Educational and Serious Games* (pp. 143–148). http://doi.org/10.1109/CGames.2015.7272972

Himmelstein, J., Couzic, M., Jennett, C. I., Cox, A. L., Goujet, R., Lindner, A., & Taddei, F. (2014). RedWire: A Novel Way to Create and Re-Mix Games. In *Proceedings of the first ACM SIGCHI annual symposium on Computer-human interaction in play - CHI PLAY '14*. http://doi.org/10.1145/2658537.2661315

Hunt, J. W., & McIlroy, M. D. (1976). *An Algorithm for Differential File Comparison*. *Bell Laboratories Computing Science Technical Report* (Vol. 41). Retrieved from http://www1.cs.dartmouth.edu/~doug/diff.ps

Iliff, M., Salas, L., Inzunza, E. R., Ballard, G., Lepage, D., & Kelling, S. (2008). The Avian Knowledge Network: a partnership to organize, analyze, and visualize bird observation data for education, conservation, research, and land management. In *Tundra to Tropics: Connecting Birds, Habitats and People. Proceedings of the 4th International Partners in Flight Conference* (pp. 13–16). Retrieved from http://www.partnersinflight.org/pubs/McAllenProc/articles/PIF09_Decision Support Tools/Iliff_PIF09.pdf

Janssen, M., Charalabidis, Y., & Zuiderwijk, A. (2012). Benefits, Adoption Barriers and Myths of Open Data and Open Government. *Information Systems Management*, *29*(4), 258–268. http://doi.org/10.1080/10580530.2012.716740

Kawrykow, A., Roumanis, G., Kam, A., Kwak, D., Leung, C., Wu, C., … Waldispühl, J. (2012). Phylo: a citizen science approach for improving multiple sequence alignment. *PloS One*, *7*(3), e31362. http://doi.org/10.1371/journal.pone.0031362

Kim, J. S., Greene, M. J., Zlateski, A., Lee, K., Richardson, M., Turaga, S. C., … Seung, H. S. (2014). Space-time wiring specificity supports direction selectivity in the retina. *Nature*, *509*(7500), 331–6. http://doi.org/10.1038/nature13240

Kohavi, R., Henne, R. M. R., & Sommerfield, D. (2007). Practical guide to controlled experiments on the web. *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '07*, *2007*, 959–967. http://doi.org/10.1145/1281192.1281295

Lee, J., Kladwang, W., Lee, M., Cantu, D., Azizyan, M., Kim, H., … Das, R. (2014). RNA design rules from a massive open laboratory. *Proceedings of the National Academy of Sciences of the United States of America*, *111*(6), 2122–7. http://doi.org/10.1073/pnas.1313039111

Lofgren, E. T., & Fefferman, N. H. (2007). The untapped potential of virtual game worlds to shed light on real world epidemics. *Lancet Infectious Diseases*, *7*(9), 625–629. http://doi.org/10.1016/S1473-3099(07)70212-8

Medler, B., John, M., & Lane, J. (2011). Data cracker: developing a visual game analytic tool for analyzing online gameplay. *Proceedings of the 2011 Annual Conference on Human Factors in Computing Systems - CHI '11*, 2365–2374. http://doi.org/10.1145/1978942.1979288

Miller, S. A. (2012). *Piwik Web Analytics Essentials*. Retrieved from https://books.google.com/books?hl=fr&lr=&id=xAI3NnTFDzUC&pgis=1

Raddick, M. J., Bracey, G., Gay, P. L., Lintott, C. J., Murray, P., Schawinski, K., … Vandenberg, J. (2010). Galaxy Zoo: Exploring the Motivations of Citizen Science Volunteers. *Astronomy Education Review*, *9*(1), 10103. Retrieved from http://scitation.aip.org/content/aas/journal/aer/9/1/10.3847/AER2009036

Schell, J. (2008). *The Art of Game Design: A Book of Lenses*. *Annals of Physics* (Vol. 54). http://doi.org/9780123694966

Sharp, L. (2012). Stealth Learning: Unexpected Learning Opportunities Through Games. *Journal of Instructional Research*, *1*.

Staiano, A. E., & Calvert, S. L. (2011). The promise of exergames as tools to measure physical health. *Entertainment Computing*, *2*(1), 17–21. http://doi.org/10.1016/j.entcom.2011.03.008

Steele, C. (2013). Open Access (The MIT Press Essential Knowledge Series). *Australian Library Journal*, *62*(3), 248–249. http://doi.org/http://dx.doi.org/10.1080/00049670.2013.811775

Tenopir, C., Allard, S., Douglass, K., Aydinoglu, A. U., Wu, L., Read, E., … Frame, M. (2011). Data sharing by scientists: practices and perceptions. *PloS One*, *6*(6), e21101. Retrieved from http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0021101

Wood, C., Sullivan, B., Iliff, M., Fink, D., & Kelling, S. (2011). eBird: Engaging birders in science and conservation. *PLoS Biology*, *9*(12). http://doi.org/10.1371/journal.pbio.1001220