Towards an Open Standard for Gameplay Metrics

Sven Charleer

KU Leuven
3001 Leuven, Belgium
sven.charleer@kuleuven.be

Katrien Verbert

KU Leuven
3001 Leuven, Belgium
katrien.verbert@kuleuven.be

Francisco Gutiérrez

KU Leuven
3001 Leuven, Belgium
francisco.gutierrez@kuleuven.be

Kathrin Gerling

KU Leuven
3001 Leuven, Belgium
kathrin.gerling@kuleuven.be

Abstract

Gameplay Metrics can help provide insights into player behaviour and performance, supporting game designers and players wishing to better understand their own game, yet such data is often not easily available to the wider public. We take our inspiration from the field of Learning Analytics, which attempts to create awareness and insights for learners by leveraging the gathered activity data and feeding this back to the students in an open way. This paper suggests an open standard for gathering and storing Gameplay Metrics data based on proven standards in Learning Analytics, facilitating easier development of analytical tools across different video games and genres to the benefit of researchers, developers, and players.

CCS Concepts

•Information systems → Data structures; •Applied computing → Computer games; •Human-centered computing → Human computer interaction (HCI); Visualization;

Author Keywords

xAPI; xVGM; Gameplay Metrics; Video Games; Standardisation; Open Data

Introduction

Gameplay Metrics [6] have been of great interest to better understand player behaviour [22], develop prediction

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the Owner/Author.

CHI PLAY '18 Extended Abstracts, October 28–31, 2018, Melbourne, VIC, Australia
Copyright © 2018 Copyright is held by the owner/author(s).
ACM ISBN 978-1-4503-5968-9/18/10.
<https://doi.org/10.1145/3270316.3271529>

Figure 1: A dashboard for League of Legends to accompany live eSports matches, providing real-time visualisations of the game state and interpretations of the data, to help spectators create a better understanding and new insights of the match. [3]

models [13], and create better AI agents [24]. Players can benefit from these data as well, through so-called “player dossiers” [15], websites that aggregate player data and allows players to explore their personal and each others’ data. Such player focused solutions can reach beyond just “casual information visualisation” [17]. Charleer et al. [3] visualise League of Legends and Counter-Strike: Global Offensive eSports match data in order to support “expert spectators”; in the context of this work it was revealed that many games are often viewed to improve one’s own game, outlining the importance of easy accessibility of Gameplay Metrics to support players.

However, extracting the data required for such applications is not straightforward. A common way for data scientists and hobbyists to attain the data is through replay files [18, 3], log files [14], and Application Programming Interfaces (APIs) [3]. API data are often limited to high level aggregated statistics regarding player and team performance. Replay files are often in binary format and the accessibility of these data depends on the availability of extraction tools, provided by either the game developers or enthusiasts. Changes in these proprietary data formats are common through the video game life-cycle making maintenance of these tools not straightforward as it often relies on the goodwill and motivation of hobbyists.

We take inspiration from the field of Educational Data Mining [26] and Learning Analytics (LA) [21], which use learner traces to get further insights into learner behaviour, but also empowers learners through Learning Dashboards [4]. This field has explored multiple standardisation initiatives in order to create simple and flexible formats that defines user interaction data across multiple technologies [19]. Experience API (xAPI) is one of the standards that emerged successfully and is widely used [2], creating a platform for

dashboard development, data portability, and analysis on both individual and large scales.

Alonso-Fernandez et al. [1] already advocate the use of xAPI for Serious Games to tackle the ad-hoc analysis present in Game Learning Analytics. However, we do believe that leveraging such a standard is beneficial to all kinds of games, its designers, players, and spectators. It can help facilitate easier development of LA-like applications such as post-processing analytics and real-time visualisation systems beneficial to improving actual play. With the popularity of eSports, “Gameplay Metrics dashboards” could help professional players with better awareness, reflection and insights of their activities which in turn could impact their performance [23]. An open standard could expand Gameplay Metrics beyond single game data, allowing comparison and aggregation across entire genres. This paper explores the first steps to create a flexible, open standard. Based on our expertise in the field of LA, we wish to explore the feasibility of using xAPI for Video Game Metrics (xVGM), taking into account the challenges that come with the wide range of video game genres.

A preliminary exploration

As a first step to explore the feasibility of an open standard, we explored the Gameplay Metrics of three games of different genres: the Multiplayer Online Battle Arena (MOBA) League of Legends (LoL) [9], the Real-time Strategy game (RTS) StarCraft (2) (SC2) [7], and First Person Shooter (FPS) Counter-Strike: Global Offensive (CS:GO) [11].

League of Legends

LoL provides an Application Programming Interface (API) which allows access to game state information [10]. However, most of this data is available in aggregated format, after the match. Another source of data is Riot Games’ web-

```

{
  "1002480198": {
    "t": 50467,
    "playerStats": {
      "...": "...",
      "4": {
        "x": 10635,
        "h": 539,
        "y": 3268,
        "movementSpeed": 243,
        "healthRegen": 7
      },
      "...": "...",
      "9": {
        "p": 281,
        "td": 58,
        "tdc": 58,
        "pd": 58,
        "x": 11166,
        "y": 3278,
        "pdc": 58
      }
    }
  },
  ...
},
},

```

Figure 2: Snapshot of the websocket data stream provided by Riot Games [10]. This update contains game state changes for both player 4 and 9, such as location (x,y), health (h), and damage dealt (td).

socket, which provides live statistics (game state) data on a one second interval [16]. This data stream is used by the official Riot website during eSports live-streams to visualise **contextual** information on teams and players. Each update contains the delta in the game state (difference in player health compared to the previous update, damage dealt, player positions,...). However, player and game events are not present in the data. Figure 2 shows an example update in the data stream.

An effort can be made to deduce events from these contextual changes, however, they are only estimates: a player X dealing damage in the same update as a player Y receiving damage *might* indicate damage dealt from player X to Y. However, enemy units, monsters, and towers could have also been the reason for the damage.

While the **context** can provide us with a lot of information, the lack of events prevents us from understanding the interaction between different units. This information is useful for pure statistical information, but does not provide insights into behaviour and cause and effect.

StarCraft 2

SC2 replay files contain all data required to re-simulate the entire match. Blizzard provides a tool to decode these files into Python data structures [8] containing e.g. initial match data, game events, and player message events. SC2's export files focus on game **events**: activities performed by the game, units, and players. Contextual data is limited to the initial state of the game and is not repeated through the log files. This logical approach is fine for replays: the game engine only requires the initial state and subsequent changes (e.g. player actions) to produce a replay. But it makes proper analysis difficult: the logic and effect of activities must be simulated to provide insights at specific time-stamps during the game, unlike LoL which provides

the game state changes at every time-stamp. Extra effort is required to recreate the missing **contextual data**: one approach [5, 12] is to load replay files back into StarCraft and use external tools [20] to extract further game state data.

Counter-Strike:Global Offensive

With no lack of third party extractions tools for CS:GO, replay files can easily be converted into usable and readable formats. Similar to SC2, as data is exported from replay files, contextual data is sparse. The data does however cover a wide spectrum: from player deaths to the specific location of a single bullet's impact [25].

Again, the lack of **contextual data** makes it difficult to interpret every event: while important events such as player deaths contains elaborate information, an event such as bullet impact only contains the shooter and spatial coordinates. Information regarding the object of impact is hard (locations of other players), or even impossible (position of crates, walls) to extract.

An Open & Flexible Standard

As mentioned in the introduction, xAPI is a standardisation for learning objects containing events performed by learners. It also contains a service description in order to manage, store and expose the data [19]. For the purpose of this paper, we solely focus on the data format.

In order to tackle the limitations mentioned in the previous section, we must provide both **event** and **contextual** data. This allows the data to be interpretable at every timestamp, without the need for knowledge regarding the intricate details of inner game engine workings. The data can be understood as-is without requiring to be processed through a game engine.

xAPI is an *event-based* format. Learners perform activities

Figure 3: Structure of the xAPI event object.

in (online) environments which can for example be used for live visualisations or stored for post-analysis. Similarly, players, but also the game, perform events. xAPI is *adaptable* and *flexible*. Every event is defined by *verbs* e.g. a course is “attended”, a paper is “submitted”. The broad range of games and game genres results in a multitude of different events that happen and must be registered [6]. CS:GO players can “jump”, “shoot”, “plant bombs”, while LoL players “perform abilities” and “target” locations. xAPI also provides room for contextual data. Through its standardised definition it also allows data to be shared across platforms and tools.

xVGM

We follow the simplified specification of the xAPI data format according to Santos et al. [19] Figure 3 shows the overall structure of the xAPI object. We explain the different fields and how they can be applied to Gameplay Metrics to create xVGM objects:

- **Actor:** An identifier of the user who performs the action. This field can contain any player identifier, enemy unit identifiers, level or the game in general.
- **Verb:** An identifier of the action that the user performs. Examples can be the player or enemy unit that *performs an ability*, *shoots a weapon*, *jumps*, or *picks up* an item. The game can *spawn* objects and *trigger* events.
- **StartTime:** A timestamp at which the event occurs.
- **Object:** The identifier of the object/outcome resulted from the action e.g. the item that was *spawned*, an item that was *crafted*.
- **Target:** Target contains the identifier of the target of the action e.g. the player/unit hit by the weapon or

ability, the target healed, or the receiver of a transaction such as gold or items.

- **Context:** Context in LA is often limited to the course name or other identifier of places in which the learning occurs (e.g. a webpage on an online course). However, in video games this context can be very elaborate. One could argue the entire game state can be beneficial for analytics. On the other hand, game state details related to the actor could be plenty. Therefore, context can contain details such as player health, status (alive/dead), inventory. Other game states could be contained in events related to its actors e.g. the health of enemy units, win and lose conditions of the game “actor”. We will further elaborate on this in the Discussion section.

A LoL Example

We captured match data using the LoL websocket [16]. This websocket initialises the data stream with the starting game state containing information on team state and individual player states. Every second an update is sent with the delta in state difference between the new state and the previous, such as new player location, player health, damage done, etc. Figure 2 presents (a subset of) the data of two players sent within one update: player 4’s health (h) and position (x,y) have changed. Player 9’s total damage to champions (“tdc”) has also changed. Therefore we could assume the damage by player 9 is dealt to player 4.

To accurately verify whether this is the case, we would need to know the previous state of health of player 4 and see if the damage dealt by player 9 matches the change in health points. This information could be found in a previous update, the initial state of the game, or somewhere in between. Even when this event is found, the difference in

```

[[
  {
    "actor": {
      "playerId": 4
    },
    "verb": "damageTaken",
    "target": "TL Doublelift",
    "object": {
      "value": -58
    },
    "starttime": 50467,
    "context": {
      "mana": 231,
      "playerName": "FOX Altec",
      "level": 1,
      "health": 539,
      "...": "...",
      "team": 100,
      "y": 3268,
      "x": 10635
    }
  },
  {
    "actor": {
      "playerId": 9
    },
    "verb": "damageDealt",
    "target": "FOX Altec",
    "object": {
      "value": 58
    },
    "starttime": 50467,
    "context": {
      "mana": 281,
      "playerName": "TL Doublelift",
      "level": 1,
      "health": 617,
      "...": "...",
      "team": 200,
      "y": 3278,
      "x": 11166
    }
  }
]]

```

Figure 4: A first attempt at converting LoL websocket game state updates to the xVGM format.

health might not be equal to the amount of damage dealt (tdc=58). Attributes such as armour and magic resistance mitigate the damage, making an accurate comparison difficult without prior knowledge of the LoL damage calculations (e.g. 58 damage can result in smaller amounts of health points lost).

In order to see what LoL data could look like in xVGM format, we created a script that keeps track of every player's game state across updates. If a change is detected, a new xVGM event is created and contextual data (game state) is updated. Whenever for example health of a player lowers and damage is dealt to a champion in the same update, we generate an xVGM *damageDealt* and *damageTaken* event. Contextual data such as current player position and health are added to the event. The actor (player dealing damage) and target (player taking damage) are also added. Note that, with the limited amount of contextual data, when multiple players are damaged, it is not possible to differentiate between the multiple attackers. The resulting events are thus an approximation.

A sample of the output is depicted in Figure 4. The result is descriptive, providing more context and, in case this was generated from a source including event data (thus not requiring "interpretation" of the data), it also creates an immediate and clear picture of the current activities and interactions between players (and others). The script, input, and output files can be found at <https://github.com/svencharleer/xVGM/>.

Discussion

Current processes to extract Gameplay Metrics are time consuming and game dependent. Creating an open standard is one step towards better access for data scientists and video game developers. This standard can also re-

sult in a portable data format, making it easier to share and compare data from different sources. This data is useful beyond mere scientific or development analysis purposes. Similar to LA, players can access personal data in order to handle them as they please. This data can be used to start developing analytics tools to support players to become better aware of their activities, reflect on their own and peer skills, and generate new insights [23].

Granularity of the required data depends on the target use cases. A one-size-fits-all solution is hard to achieve. Robertson et al. [18] acknowledged the problem of the different levels of granularity required and proposed adaptive extraction methods with different levels of detailed output. With the xVGM format, different levels of granularity can easily be extracted: through the filtering of verbs, events that are irrelevant to the application can be omitted. Unnecessary actor data can be removed to create a more manageable export.

However, the amount of events is not always the problem. Game states like that of StarCraft can be very large [18]. Pushing entire game states into every event's Context field will not be manageable. While deltas between states solve part of the problem, processing specific events requires again more effort.

Drachen et al. [6] define Gameplay Metrics as measures of player behavior, including in-game, interface, and system (game engine events). The levels of detail required to be included in the metrics (event types, time intervals, and context size) could be defined according to different needs. Developing individual player dashboards might only require the relevant player's interactions with the system while context can be limited to player states. This can provide sufficient information for insights in player performance (e.g. kills, hit rate, win/loss rate) and individual behaviour. But

when deeper insights are required, the entire spectrum of Gameplay Metrics can be taken into account. By adding for example enemy unit, environment, and AI events, each relevant actor's state can be stored in their respective context.

In previous work [3], we created spectator dashboards for CS:GO and LoL. However, due to the different data formats (event-based versus context based), a lot of time and effort was spent into understanding and converting the data. The resulting dashboards differed in what could be visualised: CS:GO's events allowed for the development of e.g. straightforward map visualisations, while LoL's game state data required estimated interpretations in order to visualise data beyond mere statistics. An open standard would allow for the distribution and reuse of modules across games. The presence of both context and event data would remove current limitation we run into.

Conclusion & Future Work

In this paper, we have provided our preliminary thoughts on the need for an open standard for Game Metrics. We have only scratched the surface by exploring three popular games that provide support for external parties to analyse and use video game data. Based on our Learning Analytics expertise, we feel xAPI for Video Game Metrics is a good starting point in creating such standard. Our preliminary findings show that xVGM is flexible enough to support the genres explored, but the current state of available data does not make the conversion straightforward.

To evaluate the feasibility of xVGM, we will further explore video game data sources of other games and genres in an attempt to get a better understanding of the available Gameplay Metrics. This process will allow for us to test whether the current format definition suffices, or where we should alter or expand the current specification. Due to

the abundance of data generated by the game events and states, we must take into account performance and storage issues, which will depend on the context in which the data is used (e.g. live visualisations versus post-processing). We will also explore the adaptability of the granularity of the export.

Our work on xVGM will result in a number of open-source parsers to generate this open standard format for multiple games. We invite both developers and researchers to collaborate on this process at <https://github.com/svencharleer/xVGM>. We will build upon our expertise in Learning Dashboards to take these data and create actionable dashboards. We believe these steps will allow us to create data visualisations that work across games of specific genres, giving insights in e.g. general MOBA, RTS, and FPS behaviour. Even data across multiple genres could be aggregated into valuable dashboards for both players, spectators, fans, and game developers.

We hope this work will result in an open standard that can be adopted in both the scientific community as well as the games industry. We believe a standardisation of Gameplay Metrics data in both serious and non-serious games can be beneficial for the scientific community, game developers, and players. The readability, the details on both event- and context-based information, and the possible independence of game logic (i.e. xVGM data can contain enough information to require no extensive knowledge of inner-game workings or external information) should simplify data handling for developers and researchers. Once this data is easily accessible, it can make way for applications that range from algorithms that predict player actions and match outcomes to "Quantified Self" approaches that improve player self-awareness, help generate insights, and impact player skill.

REFERENCES

1. C. Alonso-Fernandez, A. Calvo, M. Freire, I. Martinez-Ortiz, and B. Fernandez-Manjon. 2017. Systematizing game learning analytics for serious games. In *2017 IEEE Global Engineering Education Conference (EDUCON)*. 1111–1118. DOI: <http://dx.doi.org/10.1109/EDUCON.2017.7942988>
2. Aneesha Bakharia, Kirsty Kitto, Abelardo Pardo, Dragan Gašević, and Shane Dawson. 2016. Recipe for success: lessons learnt from using xAPI within the connected learning analytics toolkit. In *Proceedings of the sixth international conference on learning analytics & knowledge*. ACM, 378–382.
3. Sven Charleer, Kathrin Gerling, Francisco Guterrez, Hans Cauwenbergh, Bram Luycx, and Katrien Verbert. Real-Time Dashboards to Support eSports Spectating. In *CHIPLAY 2018*. in press, 1–13.
4. Sven Charleer, Joris Klerkx, Erik Duval, Tinne De Laet, and Katrien Verbert. 2016. Creating Effective Learning Analytics Dashboards: Lessons Learnt. In *Adaptive and Adaptable Learning: 11th European Conference on Technology Enhanced Learning, EC-TEL 2016, Lyon, France, September 13-16, 2016, Proceedings*, Katrien Verbert, Mike Sharples, and Tomaž Klobučar (Eds.). Springer International Publishing, Cham, 42–56. DOI: http://dx.doi.org/10.1007/978-3-319-45153-4_4
5. David Churchill and Michael Buro. 2011. Build Order Optimization in StarCraft.. In *AIIDE*. 14–19.
6. Anders Drachen, Magy Seif El-Nasr, and Alessandro Canossa. 2013. *Game Analytics – The Basics*. Springer London, London, 13–40. DOI: http://dx.doi.org/10.1007/978-1-4471-4769-5_2
7. Blizzard Entertainment. 2010. *StarCraft II: Wings of Liberty*. Game [PC]. (27 July 2010). Blizzard Entertainment, Irvine, California, USA. Last played April 2018.
8. Blizzard Entertainment. 2018. S2 Protocol. <https://github.com/Blizzard/s2protocol>. (June 2018). Accessed: 2018-07-13.
9. Riot Games. 2009. *League of Legends*. Game [PC]. (27 October 2009). Riot Games, Los Angeles, California, USA. Last played April 2018.
10. Riot Games. 2018. Riot Games API. <https://developer.riotgames.com/>. (June 2018). Accessed: 2018-07-13.
11. Hidden Path Entertainment and Valve Corporation. 2012. *Counter-Strike: Global Offensive*. Game [PC]. (21 August 2012). Valve Corporation, Bellevue, Washington, USA. Last played April 2018.
12. Jesse Hostetler, Ethan W Dereszynski, Thomas G Dietterich, and Alan Fern. 2012. Inferring strategies from limited reconnaissance in real-time strategy games. *arXiv preprint arXiv:1210.4880* (2012).
13. J. L. Hsieh and C. T. Sun. 2008. Building a player strategy model by analyzing replays of real-time strategy games. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*. 3106–3111. DOI: <http://dx.doi.org/10.1109/IJCNN.2008.4634237>

14. Pascal Lessel, Alexander Vielhauer, and Antonio Krüger. 2017. Expanding Video Game Live-Streams with Enhanced Communication Channels: A case study. *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)* (2017), 1571–1576. DOI: <http://dx.doi.org/10.1145/3025453.3025708>
15. Ben Medler and Brian Magerko. 2011. Analytics of play: Using information visualization and gameplay practices for visualizing video game data. *Parsons Journal for Information Mapping* 3, 1 (2011), 1–12.
16. Nelson Minar. 2015. Riot Games API. <https://nelsonsllog.wordpress.com/2015/08/15/101-live-match-stats/>. (August 2015). Accessed: 2018-07-13.
17. Zachary Pousman, John Stasko, and Michael Mateas. 2007. Casual information visualization: Depictions of data in everyday life. *IEEE transactions on visualization and computer graphics* 13, 6 (2007), 1145–1152.
18. Glen Robertson and Ian Watson. 2014. An Improved Dataset and Extraction Process for Starcraft AI. *Proceedings of the Twenty-Seventh International Florida Artificial Intelligence Research Society Conference* (2014), 255–260. <http://www.aaai.org/ocs/index.php/FLAIRS/FLAIRS14/paper/view/7867>
19. Jose Luis Santos, Katrien Verbert, Joris Klerkx, Stefaan Ternier, Sven Charleer, and Erik Duval. 2015. Tracking Data in Open Learning Environments. *Journal of Universal Computer Science* 21, 7 (2015), 976–996.
20. sf editor1. 2018. Brood War Application Programming Interface. <https://sourceforge.net/projects/bwapi.mirror/>. (June 2018). Accessed: 2018-07-13.
21. George Siemens and Phil Long. 2011. Penetrating the fog: Analytics in learning and education. *Educause Review* 46, 5 (2011), 30–32.
22. Rafet Sifa, Anders Drachen, and Christian Bauckhage. 2018. Profiling in Games: Understanding Behavior from Telemetry. *Social Interactions in Virtual Worlds: An Interdisciplinary Perspective* (2018).
23. Katrien Verbert, Erik Duval, Joris Klerkx, Sten Govaerts, and José Luis Santos. 2013. Learning Analytics Dashboard Applications. *American Behavioral Scientist* 57, 10 (2013), 1500–1509. DOI: <http://dx.doi.org/10.1177/0002764213479363>
24. Ben G. Weber, Michael Mateas, and Arnav Jhala. 2012. Learning from Demonstration for Goal-driven Autonomy. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence (AAAI'12)*. AAAI Press, 1176–1182. <http://dl.acm.org/citation.cfm?id=2900728.2900895>
25. AlliedModders Wiki. 2015. Counter-Strike: Global Offensive Events. https://wiki.alliedmods.net/Counter-Strike:_Global_Offensive_Events. (November 2015). Accessed: 2018-07-13.
26. Annika Wolff, Zdenek Zdrahal, Andriy Nikolov, and Michal Pantucek. 2013. Improving Retention: Predicting At-risk Students by Analysing Clicking Behaviour in a Virtual Learning Environment. In *Proceedings of the Third International Conference on Learning Analytics and Knowledge (LAK '13)*. ACM, New York, NY, USA, 145–149. DOI: <http://dx.doi.org/10.1145/2460296.2460324>