GDEV60001 GAMES DEVELOPMENT PROJECT

Integrating Real-Time Metrics and Telemetry in Live-Multiplayer Games

Written by:

Jamie Dovaston

Supervisor:

Davin Ward & Benjamin Williams

Second Supervisor:

David White

Level 6 Games Design and Programming

Final Year Project

jamdov

University of Staffordshire

Contents

1. Abstract

Throughout this paper, research has been conducted into the integration of real-time metrics and telemetry within games development. Data systems structure is a key step to success within data management, particularly when dealing with large-scale datasets commonly described as big data (Moorthy *et al.*, 2015). Within this paper, data architecture is evaluated, looking into the primary purposes behind utilising both Relational Database Management Systems (RDBMS) and Non-Relational Database Management Systems (NoSQL). Furthermore, research has been conducted into modern data technologies including GoLang (The Go Programming Language, 2009) and Redis (Redis, 2009), highlighting their roles in building scalable, concurrent, and real-time capable backend services (Seguin, 2012). While organisations adopt a wide range of proposed data models and pipelines, this paper specifically targets developing an understanding of the core foundations required for implementation.

In almost all of AAA multiplayer game development, data analysis is used to make informed design decisions. Notable examples include Tom Clancy's: Rainbow Six Siege (Ubisoft Montreal, 2015) and Max Payne 3 (Rockstar Studios, 2012b), both of which utilise player data collection to monitor community behaviour and iteratively improve the player experience (Ubisoft Montreal, 2024). This report therefore aims to act as a practical handbook for smaller developers and studios, outlining approachable methods for implementing telemetry systems and enabling a greater level of competitiveness within an increasingly data-driven industry.

Additionally, an open-source data collection project has been developed and evaluated to demonstrate how data systems can be applied in practice. Utilising the modern practices explored within this research, a framework has been created to collect spatial telemetry and behavioural data relevant to both game ideation and for user recreation. User testing was conducted using this framework, where participants were asked to evaluate the ethics of telemetry in games and justify their decision when choosing whether to enable or disable data collection. 10 participants answered – results were evenly split (50/50), with participants indicating that their decision was strongly influenced by trust, transparency, and their overall perception of the company or product.

2. Introduction

With the globalisation of games and the competitiveness of the games space, data analytics are a vital step to maintaining competitiveness in an increasingly difficult industry to navigate. For better data-driven decision-making and results, companies need to include data analytics and decision support systems (DSS) (Choudhary *et al.*, 2024) to improve the survivability of their organisation, and this does not stop within just the business world.

Big data has rapidly developed into a hotspot that attracts great attention within the tech industry (Jin *et al.*, 2015). The issue of effectively and efficiently supporting analytics over big data is a relevant data management context for big data research. Modern analytics engines like *Google Analytics (Google, 2005)* and *Amazon S3 (Amazon AWS, 2006)* have been widely adopted by companies to process billions of transactions at one time, like *Facebook* (Meta Platforms Inc., 2004) and *X* (X Holdings Corp., 2006) *(Cuzzocrea, 2014)*. Throughout this paper, reviews of several techniques which have been (and could be) adopted by the games industry to collect real-time metrics in live multiplayer scenarios will be covered.

The introduction of telemetry has helped to eliminate traditional management challenges associated with a lack of accurate data on a product's use (Shevchenko, 2025). In the context of the World Wide Web, large amounts of data can be associated with. Problems may begin

to arise during the process of information retrieval; however, operating under a single source can make data integration simpler (Ahamed and Ramkumar, 2016). As part of research, evaluating understanding of primary single-source data through backed business examples and successes and applying it to the game's development process is the key to integrating these systems into the games industry.

In this paper, understanding the architecture required to integrate data collection and telemetry systems into modern multiplayer video games is the main goal. This includes describing how these systems can assist in making data-backed development decisions through user testing in both controlled and uncontrolled environments. Included is a review of different development solutions for these systems, including strengths and weaknesses contextualised by the project.

3. Aims and Objectives

- Review big data terminology and big data integration methodologies, including SQL, NoSQL, and modern analytics engines (*Golang*, *Redis*, etc.) for integrating data analytics systems.
- Review common understanding of multiplayer games, including (but not limited to) gameplay analytics tracking - using web telemetry methodologies (d3.js, Chart.js, etc.) to make data-backed design decisions
- Understand the importance of real-time metrics and telemetry on multiplayer game development with an example, conducting user testing to evaluate ethics and test data verification.

4. Literature Review

4.1 Storing Big Data, Methodologies, & Terminology

When researching into the integration of real-time metrics and telemetry for live multiplayer games, normal data storage methodologies are a primary element to understand best practices in large storage bases. The term big data is typically used to describe large-scale data models; however, its definition is largely vague and amorphous (Moorthy *et al.*, 2015). Wang et al. (2018) explain that the term big data comes from a heavily computer science-based background, the term being used for the first time in 1997 to explain the challenges posed to visualise data on computer systems. Cosic et al. (2012) define big data analytics capabilities as the ability to utilise resources as a method to complete business data analytics tasks. This data management terminology can be summarised in the following four ways (Moorthy *et al.*, 2015): Business Assessment, Data Assessment, Solution Development, and Insight.

Business Assessment

Businesses use big data in business assessment/analysis of themselves to understand business needs and objectives. Chen et al. (2012) described the term big data analytics (BDA) as a heavily related field for business intelligence & analytics. Côrte-Real et al. (2017) explains that in the strategic management of organisations, BDA can boost a company's agility and dynamic capability when in search of a competitive advantage.

Data Assessment

Data Assessment is used when auditing real-world information, both internally and externally. This can help in determining what information is relevant to address a challenge. The characteristics of big data quality can be shortened to 4Vs: Volume, Velocity, Variety, and

Value (Parashar, 2013). Volume refers to the large amounts of data. Velocity means that the data being formed is at unprecedented speeds while requiring to be dealt with timely. Variety is the indication of multiple data types to deal with, which requires being divided into structured and non-structured data. Value density becomes inversely proportional to data size – the greater the scale, the less relatively valuable the data is (Cai and Zhu, 2015).

Solution Development

Big data analysis enhances solution development through uncovering information for the solution to a specific case. Cybulski et al. (2015) explain that Interactive Visual Analytics (IVA) can help to visualise approaches through analytic reasoning. These IVAs can help to make data more structured and assist in understanding arising problems within products. Big data is not only about accessing new data sources, transactions, and collecting information, but proper analysis of data and the relationships they form between themselves. Sen et al. (2015) explain that this is where big data will become an effective solution for complex business-based issues.

Insight

The insight of big data can create visualisations that demonstrate the value of big data in analytical models. The ideology of visual communication links to the modern-day switch to consumerism and market towards mainstream platforms. Under the change of the market economy, the transformation to visual communication performance is also particularly important (Weiming Zhu, 2021). As the amount of data organisations generate grows every day, the ability to visualise data is a crucial element of scientific research. It can help to discover patterns, comprehend information, and form opinions based on them (Matthew N. O. Sadiku *et al.*, 2016). Modern data analysis has complexity with the inclusion of online data and sway. The trend of creating multiple data sources and building data warehouses has created untold potential of data through business analytics, with visualisation trends likely only to grow with time (Moorthy *et al.*, 2015).

When choosing the correct data management system for a project, there are two main solutions which you can choose from – Relational (RDBMS) & Non-Relational (NoSQL) Database Management Systems. NoSQL has been proposed to handle data volumes with no structure more efficiently, while classical RDBMS relational models consist of relations to record the data, revealing many performance issues (Abbas Fadhel and Ali Jameel, 2022). In the business sector, many systems use SQL and NoSQL for their daily tasks, especially in web analysis and the support of large websites with high availability and scalability (Lawrence, 2014).

4.1.1 Relational Database Management Systems (RDBMS)

The question of what data model for storing and managing large quantities of structured data seemed to be settled when the relational model was presented by Baxendale and Codd (1970). This model gained wide acceptance among commercial database vendors and the database research community (Hidders, 2001).

Relational database management systems (RDBMS) are a sort of database where information is stored in tables. Relational databases are the evolution of four decades of work within the data hierarchy structure (Carpenter and Cassandra, 2016). A key component of relational databases is the Structured Query Language (SQL), which is powerful for several reasons. Firstly, it allows the user to represent relationships within data, using multiple forms of Data Manipulation Language (DML). This includes the ability to insert, select, delete, update, truncate, and merge data. Most databases utilised by organisations are usually relational

databases. Relational databases can deal with large amounts of information and complex inquiries (Ali *et al.*, 2019).

With these databases being used for decades, it has led to well-defined best practices, standardisation, and extensive community support. Major RDBMS providers (MySQL (Oracle Corporation, 1995); PostgreSQL (The PostgreSQL Global Development Group, 1996); Microsoft SQL Server (Microsoft Corporation, 1989); and Oracle (Oracle Corporation, 1979)) offer robust solutions for enterprise applications (Antonio *et al.*, 2025). In 1994, relational database vendors scrambled not only to compete but also to provide products and services that better satisfied the demand for the next generation of client/server, distributed computational systems and their users. These ever-skyrocketing demands from developers and users for faster and better access to data lead database vendors to continue to provide tools and services that satisfy the needs of modern tech (Rosenblatt, 1994). This evolution of data providers continues even today, as the age of artificial intelligence begins to take over.

When developing relational database management systems, Modelling improves understandability, reusability, and maintenance of these systems. UML has been widely adopted as a standardised modelling method for business, database, application, and system architecture. By having one single language, everyone involved can communicate their thoughts, ideas, and requirements (Yin and Ray, 2005).

RDBMS have long been recognised as the dominant method for organising and querying structured data in information systems. Since Baxendale and Codd formally introduced the relational model in 1970, which underpins modern RDBMS technology, these systems have provided a principled and flexible means of structuring data into tables with clearly defined relationships and integrity constraints, enabling complex querying through SQL (Structured Query Language) and robust transactional support. Despite the emergence of alternative data storage paradigms such as NoSQL, the relational model's maturity and widespread adoption mean that RDBMS continue to play a central role in enterprise-scale applications where strong consistency and complex relationships are required. The enduring relevance of the relational approach is evident in how organisations still leverage RDBMS for a broad spectrum of operational workloads while selectively integrating other models when different performance or scalability characteristics are needed (Nance *et al.*, 2013).

Relational Databases in the Video Game Industry

Relational databases remain a foundational technology for many videogame backends, particularly when managing highly structured and interconnected user data. Core game systems such as player accounts, authentication, inventories, leaderboards, and transaction records are often best served by relational models due to their strong consistency, transactional integrity (ACID), and mature tooling for complex queries (Tencent Cloud, 2025).

For example, massively multiplayer online games (MMOs) like *World of Warcraft* (Blizzard Entertainment, 2004) and *Second Life* (Linden Lab, 2003) have historically relied on RDBMS such as MySQL and Microsoft SQL Server to handle vast amounts of account and game state data reliably across distributed systems, demonstrating how relational models help maintain persistent player state and social relationships at scale (Kasenides and Paspallis, 2019).

Cloud providers also highlight how distributed SQL databases — which extend traditional RDBMS capabilities across global deployments — can support large player populations without sacrificing transactional accuracy, making them suitable for global authentication, inventory systems, and competitive features (Kantamani *et al.*, 2025). Despite the rise of NoSQL for analytics and real-time telemetry, relational systems continue to serve as the

backbone for structured game data, offering predictability and robustness that developers rely on for critical game logic and user data integrity.

4.1.2 Non-relational Database Management Systems (NoSQL)

Not only SQL (NoSQL) is the term used to describe databases that lack querying (SQL) or relational data models. These databases are primarily used to handle Big Data (Martinez-Mosquera *et al.*, 2019). Choi et al. (2014) explains that they are useful databases for increasing application productivity and are used primarily when dealing with large amounts of data, further providing more stable and faster performance at the expense of data consistency, which is an important reason for moving towards non-relational databases (Venkatraman *et al.*, 2016). SQL-based querying has become a significant challenge when handling huge volumes of data (Ali *et al.*, 2019).

NoSQL databases provide mechanisms for storing and retrieving model data in ways other than just table relationships, as used in RDBMS. The non-relational databases are primarily categorised by four types: Key-Value Data Model, Document Data Model, Column Oriented Data Models, Graph-Oriented Data Model (Samanta *et al.*, 2018).

Key-Value Data Model

Key-value storage systems are generally referred to as schema-less, which suggests that it is not necessary to develop a model before development starts. This gives many advantages for prototyping or in exploratory development; however, as data expands and the application grows, the need for organisation in some way arises (Rossel and Manna, 2017). Rossel and Manna (2017) explains that to model and implement a key-value data store correctly, it is necessary to consider the access patterns established within the requirements. It is assumed that before developing this data model, you have considered the feasibility of this storage management system and have analysed it.

Although the NoSQL data model was designed to operate without a schema, it is suggested to have some form of data structure in the database (Abdelhedi *et al.*, 2016). Modelling helps to design the data processing flow (Da Silva *et al.*, 2014) – a key concept needed to maintain any form of structured data, especially in the analysis of information.

Document Data Model

The document data model takes large, complex document files and attempts to organise them. It has the flexibility to add as many fields as it wants at any length (Goli-Malekabadi *et al.*, 2016). In a typical structured table, uniform fields are typically required; however, with these databases, they can add any number of fields to a document as they would like (Leavitt, 2010).

The document-oriented data model is designed to store and retrieve documents within a database. These documents can take multiple formats such as XML, JSON, or BSON, and typically follow a self-describing, hierarchical tree structure composed of maps, collections, and scalar values. Each document, usually associated with a unique key, contains a set of fields that can be individually indexed. Documents are generally organised into collections, and a document-oriented database consists of multiple such collections. These collections are created dynamically as new data is inserted into the system (Hashem and Ranc, 2016).

Column-Oriented Data Model

Rather than storing sets of information in structured tables with multiple rows and columns, the column-oriented data model uses one extendable column of closely related data (Leavitt, 2010) – as classic DBMS does. In the columnar approach, each attribute is stored in a different table, so successive values of an attribute are stored consecutively. This advantage is

important for data warehouses, where information is generally taken by aggregating vast volumes of data (Matei, 2010).

Since column-oriented DBMS stores each attribute independently, Abadi et al. (2007) explains that a mechanism should be used to stitch together multiple attributes within the same logical tuple into a physical tuple. All proposed column-oriented architecture accomplishes this by attaching tuple identifiers or positions to column values. This can then be reconstructed by finding the tuple identifier/position. Modern column-oriented systems store columns in their position order. This accelerates the reconstruction process (Stonebraker *et al.*, 2005).

Graph-Oriented Data Model

The graph-oriented data model represents information as a network of nodes and edges, where nodes act as entities and edges define the relationships between them (Levene and Poulovassilis, 1990). This model is designed to capture the interconnected nature of data rather than focusing solely on individual records. It is particularly suited to applications where relationships are as important as the data itself, such as social networks, recommendation systems, and knowledge graphs (Angles and Gutierrez, 2008).

Unlike traditional models that require predefined structures, the graph-oriented model allows for flexible and dynamic representations. Both nodes and edges can contain properties, providing context and meaning to the connections within the data (Robinson *et al.*, 2015). Levene and Poulovassilis (1990) explain that this model enables a more semantic approach to data organisation, as relationships can be explored directly without the need for complex join operations.

In practice, the graph-oriented data model provides a clear advantage when dealing with highly connected information. It focuses on relationships as first-class elements within the data, supporting flexible schema evolution and efficient traversal between related entities.

Non-Relational Databases in the Video Game Industry

Non-relational databases have become particularly important in the video game industry, where developers must manage massive volumes of dynamic and real-time data generated during gameplay. In online and multiplayer games, data such as player profiles, inventory states, leaderboards, match results, and behavioural analytics are continually updated and queried, placing significant demands on backend systems. NoSQL databases are frequently preferred in these scenarios because they offer greater scalability, faster performance, and flexibility compared with traditional relational systems, enabling efficient handling of frequent data modifications and high concurrency without the overhead of predefined schemas. In fact, the rapid pace of user interactions and large user bases typical of modern games make NoSQL solutions well-suited to real-time applications where relational databases struggle to keep up with throughput and latency requirements. Economies of scale in the storage and processing of this big game data are essential as developers seek to extract actionable insights for player retention and game optimisation (Costan, 2025).

4.1.3 RDBMS & NoSQL Comparison

The relational model structures data into multiple linked tables made up of rows and columns. These tables are connected through foreign keys stored within specific columns. When data is queried, the system retrieves information from several tables to provide the complete result — essentially answering the user's question through combined data relationships. In contrast, non-relational data models are typically designed around the application's specific queries and access patterns, rather than starting from a fixed relational structure (Hashem and Ranc, 2016).

Although the relational data model turned out to be a very simple but effective way to represent data in databases, this led to the need to incorporate more semantics into the data model to distinguish between entities and relationships (Hidders, 2001).

Conventional relational database systems use two-dimensional data sheets with properties to interact, interpret, and query; however, they are not effective for huge data queries (Tauro *et al.*, 2013). Although relational databases have been improved over their prolonged existence, they unfortunately started to show their age in modern software design. They do not perform well under large amounts of data, giving less good performance for dynamic schemas (Kaur and Rani, 2013).

Query Models

When looking at the difference between RDBMS & NoSQL database management systems, there are substantial differences in their querying capabilities. Document datastores (such as CouchDB & MongoDB) allow for complex queries. This is not surprising due to the design of many NoSQL databases, dynamic querying features being removed for performance and scalability (Strauch and Kriha, 2011).

When comparing the query models of RDBMS and NoSQL database management systems, the differences are both structural and philosophical. Relational databases rely heavily on SQL, a declarative language that supports complex joins, aggregations, nested queries, and transactional guarantees, making it highly effective for applications requiring strong consistency and multi-table relationships. In contrast, many NoSQL systems deliberately remove or limit dynamic querying capabilities to optimise for distributed performance and horizontal scalability.

Strauch and Kriha (2011) note that while document-oriented databases such as MongoDB and CouchDB do support rich, flexible queries over their semi-structured data, other NoSQL models – particularly key-value and wide-column stores – restrict querying to primary-key lookups or limited secondary indexes to maintain low latency across distributed clusters. This design trade-off reflects the broader NoSQL philosophy: queries are shaped around data access patterns defined by the application, rather than relying on general-purpose querying tools. As a result, developers must often design NoSQL schemas with query patterns in mind, embedding or duplicating data to avoid expensive cross-collection or cross-node operations.

While this approach improves performance at scale, it shifts a portion of the query optimisation burden from the database engine to the application layer. Ultimately, the contrast in query models reflects the different priorities of each system – relational databases prioritising expressive, ad hoc querying, and NoSQL prioritising speed, scalability, and predictable access patterns (Strauch and Kriha, 2011).

Schema

A fundamental difference between RDBMS and NoSQL systems lies in how each handles schema design and schema evolution. Relational databases enforce a strict, predefined schema, meaning that the structure of tables – including data types, constraints, and relationships – must be defined before data can be inserted. This rigid schema ensures consistency and integrity but makes structural changes costly, particularly in large-scale systems where schema migrations can disrupt performance or require complex reconfiguration (Kaur and Rani, 2013).

In contrast, NoSQL systems are typically schema-less or schema-flexible, allowing data to be stored without requiring adherence to a fixed structure. This flexibility enables developers to

adapt their data models rapidly as application requirements evolve, making NoSQL highly suitable for agile development environments and applications handling diverse or rapidly changing data formats. Costan (2025) explains that schema flexibility is a key reason why NoSQL systems are increasingly favoured in domains such as gaming and big data analytics, where incoming data may vary significantly in structure, volume, and velocity. Furthermore, the ability to add fields on demand – without modifying existing records – allows NoSQL databases to support dynamic content, scalable architectures, and fast iteration cycles, characteristics that traditional RDBMS struggle to match when dealing with modern, data-intensive workloads.

	RDBMS	NoSQL
Structure	• Relational database model. • Special unduplicated key to identify data stored in each row. • Uses Structure Query Language (SQL).	• Non-relational database model. • Stores data in flexible, schema-less models. • Uses various query language models.
Schema	• Fixed schema changes require careful planning and migrations.	• Dynamic or schema-less allows easy modification and adaptation to changing data structures.
Efficiency & Effectiveness	• Renowned for being a slower, more structured approach. • Fixed schema makes changes complex	• Known for its speed and efficiency with little structure. • Used to handle high data volumes. • Dynamic schema makes it easier to modify.
Query Language	• Uses Structured Query Language (SQL) for defining and manipulating data.	• Uses various query models depending on database type (e.g., document queries, key-value lookups, graph traversals).
Use Cases	• Applications requiring strong consistency, complex relationships, and transactional integrity (e.g., banking, inventory management, player account data).	• Applications requiring flexibility, fast writes/reads, and scalability (e.g., real-time analytics, telemetry, social interactions in games).

Table 1 RDBMS & NoSQL Comparison Sheet

4.1.4 GoLang (Go)

GoLang (The Go Programming Language, 2009, hereafter Go) is a modern programming language with its design philosophy emphasising simplicity, expressiveness, robustness, and efficiency. It has strong support for concurrency and dependency management – it is a compelling choice for systems programmers (Rios, 2024).

GoLang features a Unix-minded design by checking all the boxes for simplicity. Rios (2024) explains that many Python and Ruby proficient developers transition to Go, as it allows them to retain their level of expressiveness while still achieving better performance and the capability to work with concurrency. A big element of Go's philosophy is its choice to aim for convenience for developers over a primary aim for CPU usage.

Uzayr (2023) describes Go as having an unmatched simplicity, despite it being a very advanced language with a robust feature set – it stands out from the crowd due to its ease of use and fundamental approach.

GoLang Advantages	
No Generics	• Generics and patterns add to the ambiguity and difficulty of comprehension. • Designers simplified things by choosing against them.
No Dynamic Libraries	• Go opted to omit any dynamic libraries. • Plug-in packages can be added through plug-in packages in the newer versions of the program (*Go 1.10 version*)
Single Executable	• GoLang does not come with a runtime library. • A single executable can be produced and can be released simply by copying. • It eliminates any worries about making mistakes through dependencies or version incompatibilities.

Table 2 GoLang Advantages (Uzayr, 2023)

Overall, GoLang presents a strong, developer-friendly option for building performant and concurrent applications, with a clear goal of simplicity and maintainability. When compared to working directly with RDBMS and NoSQL databases, Go provides the tools to efficiently interface with them but cannot replace them. While Go's simplicity and single-executable design make it sound like a proficient pathway for backend development, using it as a full-stack solution that manages complex data storage entirely in code could quickly become expensive and difficult to maintain.

4.1.5 Redis

Redis (2009) is a popular in-memory data store that is fast and works with lots of different programming languages. It runs in the background and can be used on the same machine or over a network, which makes it great for caching or keeping track of game stats in real time (Seguin, 2012). Redis is renowned for how easy it is to learn. Through a couple of hours of learning, you can get the basics if using simple commands such as SET and GET to store and read data. It works with Python, C/C++, R, and much more (Eddelbuettel and Balamuta, 2018).

For game analytics, Redis is useful for tracking player scores, session stats, and leaderboards. Because it is in-memory, it is fast at updating and giving live results. That said, Redis is not meant to replace a full database due to its complexity (Eddelbuettel and Balamuta, 2018). It is best as a helper, sitting alongside your main database to make parts of the system quicker and easier to work with.

4.2 Multiplayer Games Development

Multiplayer games development has a rich history dating back to the earliest of computer games. In the earliest days of multiplayer gaming, there existed no networking between players. Multiplayer functionality was achieved by having players interact with the same

computer. Players would manipulate their avatars through shared, common screens or the screen could be partitioned into a split-view video screen, with each player owning a section of the screen (Armitage *et al.*, 2006).

With the introduction of online games played over or on an internet connection also came a surge of online multiplayer games. This took the local & LAN multiplayer systems already developed and took them to a global scale. Crawford et al. (2011) describe this transformation as a significant cultural phenomenon – especially when looking at the most obvious and frequently discussed, large number of massively multiplayer online role-playing games, which attract player communities larger than many small- to medium-sized countries' populations.

This section is a review of the brief history of multiplayer games before opening it up to the wider scale, and how data systems are used and applied to multiplayer games development.

4.2.1 History of Multiplayer Games Development

Multiplayer Games Development has a rich history that is vital in evaluating the steps that have been made to reach this point. Throughout chapter 1, Glazer and Madhav (2015) give a brief history of multiplayer gaming, dating back to the earliest personal computers of the 1970s.

Serial ports allow data to be transmitted one bit at a time, with their typical purpose being to communicate between external devices, such as printers & modems. In the 1980s, an issue of *BYTE Magazine (1980)* featured an article on how to develop the then-named *'Multimachine Games'* in BASIC by Wasserman and Stryker (1980). A drawback of using serial ports was due to typical computers only contained 2 ports. This meant that daisy chains had to be used to connect more than two computers. This meant that although the technology had been around since the early 1980s, it was not properly taken advantage of until the 1990s, when it really gained traction.

Networked games would eventually expand to Multi-User Dungeons (MUD). These were typically a text-based style of game where several players could connect to the same virtual world.

In the early 1990s, Local Area Network (LAN) games were created, led by games such as *Doom (1993)*, which took gaming by storm. When the internet began to be adopted, games such as *Unreal (1998)* became very popular.

Online games started to be adopted through consoles in the early 2000s. One type of game is the Massively Multiplayer Online (MMO) game model, which supports thousands of players simultaneously in the same game session. The game *Starsiege: Tribes (1998)* implemented a networking architecture relevant to modern-day technology – the client-server model, allowing for clients to connect to a server that coordinates the game.

At a similar time, *Age of Empires (1997)* was developed to allow for computers to connect in a peer-to-peer manner. Instead of sending information about each unit over a network, instructions are sent to each peer to evaluate the commands individually. To ensure that machines are synchronised, a turn timer is used that saves commands over a period before sending them over a network. These commands will not be executed until two turns later, which gives enough time for players to execute and receive their own commands.

4.2.2 Importance of Data Systems in Competitive Multiplayer Design

Over the past few decades, the composition of player bases has undergone significant shifts. Communities are now populated with a vast and varied spectrum of people, many of whom are unaligned with the attributes of a ‘traditional gamer’ (Shaw, 2012). This is a key example

of how differences in preference make data analytics vital in the success of multiplayer gaming. Making community-curated decisions helps in creating a thriving user experience for all parties interested in a product.

Tom Clancy's Rainbow Six Siege (2015) Development Logs

Quantitative and qualitative data analytics help to understand the user base of your game, a significant area that is useful for making informed design choices during development. A clear example of the importance of data-driven decision-making in multiplayer design can be seen in Ubisoft Montreal (2015) ongoing development of *Tom Clancy's Rainbow Six Siege (2015, hereafter R6S)*.

The studio publishes monthly *Designer's Notes*, which outline upcoming balance changes and the reasoning behind them. These updates rely heavily on quantitative data such as operator pick rates, win-rate deltas, weapon performance metrics, and broader behavioural trends within the player base.

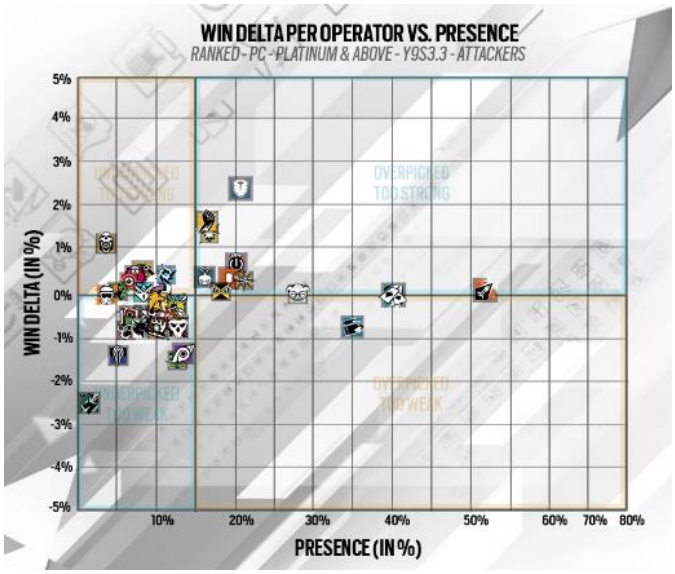

Figure 1 Ubisoft Montreal (2024) 'Win Delta Per Operator vs. Presence'

By making design decisions on real-time metrics rather than intuition alone, the developers ensure that changes reflect the needs and behaviours of a diverse and evolving community. R6S's development demonstrates how analytics enable designers to identify imbalances, anticipate meta shifts, and maintain long-term competitive integrity – all of which are essential for sustaining a healthy, player-centred multiplayer player base.

Figure 2 Ubisoft Montreal (2024) 'Y9S4 Rank Distribution: PC

Graphs are used to present rank distribution because they show how players are spread across ranks in a clear visual format. This enables the identification of population concentrations, gaps between tiers, and changes in distribution over time. These visual patterns help developers verify how ranking and matchmaking systems to ensure they are functioning as intended. This shows how important telemetry can be to help in identifying balance changes within gameplay – the more reason why smaller studios/independent developers should have access to such tools.

Max Payne 3 Rockstar Social Club Match Report System

Although data analysis & telemetry are very useful for design improvements, players may find insights into their individual progress similarly important for competitiveness, general game understanding, and more. Some game developers offer in-game data analytics systems for players, allowing them to analyse previous matches, view graphed data, and more. These systems are very important for creating competitive integrity and allow players to improve their skills and understanding through game data.

Rockstar Studios (2012b) implemented data systems for Max Payne 3 singleplayer and multiplayer modes through their Social Club platform, where players could access game statistics from previous matches, compare data to their friends, and much more. These systems helped elevate competitiveness between players, showing leaderboards that tailor their statistics to their friends. These systems are completely automated and are gathered through data collection while users are playing – creating a 'side-by-side comparison' report that allows you to view your single-player and multiplayer data alongside fellow players (Rockstar Studios, 2012a).

Figure 3 Rockstar Studios (2012) Side-by-side comparison

Rockstar Studios (2012a) released a news article explaining how to use this feature. They explain the tool as *"a quick and indisputable end to any Max Payne 3 verbal one-up manship".* This reverberates the use of this tool, allowing users to compare themselves with their friends. Furthermore, they offer tools to generate unique tables of players selected by the user, adding a level of customisation.

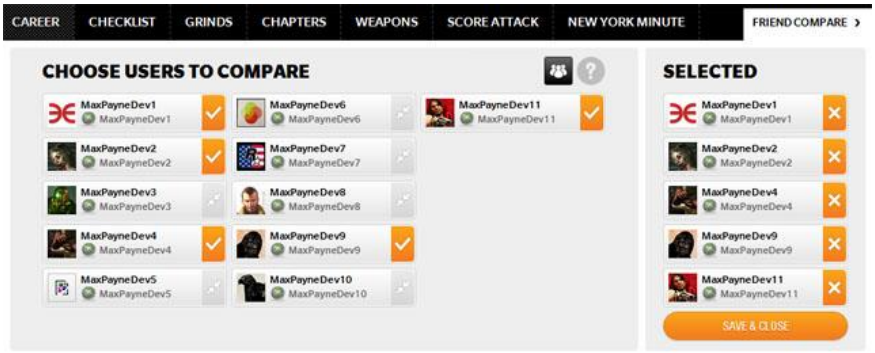

Figure 4 Rockstar Studios (2012) Choose Users to compare

In another article, Rockstar Studios (2012c) explains a tool allowing players to view their previous matches and stats within them. An interesting element of these match reports is the heatmaps that allow players to view where they shot their weapon the most throughout the match. This allows players to view where actions occurred throughout the match, giving a better insight into how they played. Players can use these insights when analysing how to improve their playstyle, especially for competitiveness.

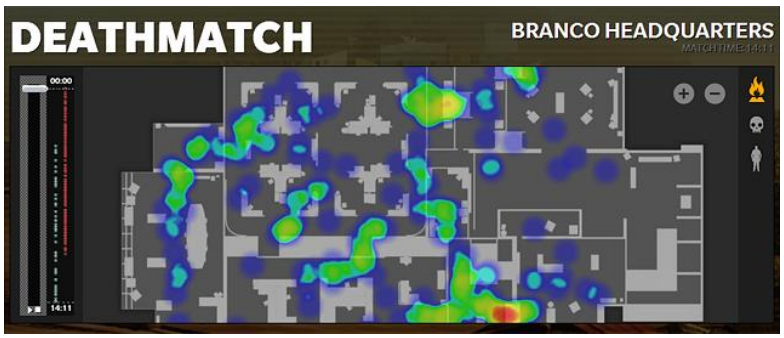

Figure 5 Rockstar Studios (2012) Deathmatch

Features like these are important in creating a competitive community, as they allow players to gain an edge in their playstyle while attempting to improve. Rockstar Studios (2012b) uses the Rockstar Social Club features constantly throughout their titles as a way for players to check their statistics without physically logging into the game. They use this platform in parallel to a typical gaming experience as an extra for users who want to view more information from their playthrough.

4.2.3 Telemetry in Multiplayer Gaming

Telemetry systems are extremely useful in elevating communities and are widely used by most triple-A and, in some cases, double-A modern multiplayer game developers. They assist in giving players an insight into both how their progress tracks globally, as well as by tracking personal progression. With modern gaming becoming more competitive in its communities, players must strive to use as much data as possible to elevate themselves above the level playing field.

Telemetry also serves as a foundation for more sophisticated analytical practices that help developers understand not only what players do, but why they do it. Wallner et al. (2014) highlight how modern telemetry pipelines enable studios to collect high-resolution behavioural data, which can then be transformed into models of player engagement, skill progression, and decision-making. These insights enable developers to refine game balance, identify problematic design patterns, and support data-driven iteration throughout a title's lifespan.

Empirical evidence helps to shape the environment in which competitive players inhabit, creating ecosystems where being strategic and fairness are continuously monitored and adjusted. By grounding design decisions in telemetry-derived analytics, developers create multiplayer spaces that reward mastery while remaining responsive to the evolving behaviours of their communities (Wallner *et al.*, 2014).

4.3 Unity Engine Development

The Unity (Unity Technologies, 2026) Engine is a renowned tool made for professional studios and indie developers alike. It is a powerful integrated game engine and editor that enables you to efficiently create objects, import external assets, and link them all together with a customised coding syntax. The editor is visually driven and built in a way that allows you to do everything with simple drag-and-drop motions. This enables you to connect scripts, assign variables, or create multi-part assets (Menard *et al.*, 2015).

The engine has a plethora of multiplayer games connected through its self-hosted Unity Game Services (Unity Technologies, 2026). This tool offers game developers & companies a selection of processes to assist in setting up for multiplayer gaming. These multiplayer systems enhance the development of multiplayer games by enabling developers to utilise professional-grade systems with as little as a panel and a few lines of code. Naturally, it is possible to create custom systems; however, with the Unity Game Services, that is not necessary.

4.3.1 Unity Engine for Game Development

Unity is widely used because it supplies developers with everything needed to build full games in one place. Singh and Kaur (2022) explain that the engine includes systems for graphics, audio, physics, animation, and more, all working together inside the Unity Editor. This setup makes it easier for developers to test ideas quickly, change things on the fly, and keep both design and programming closely connected throughout development.

Haas (2014) shows that Unity's popularity also comes from how accessible it was when it was first released. At the time, many engines were difficult to use or locked behind expensive licenses. Unity focused on being easier to learn and supporting many platforms, which helped it grow among indie developers as well as larger studios. This early focus on accessibility shaped how the engine is still used today.

More recent work, such as Tikhomirov (2023), highlights how the Unity Editor is organised into different windows that each serve a specific purpose. The Scene and Game views help with layout and testing, while the Hierarchy, Inspector, and Project windows handle objects, components, and assets. This layout keeps everything visible and manageable, even in large projects. Taken together, these sources show how Unity's tools, history, and editor design all contribute to its ongoing role as a major engine in modern game development.

4.3.2 Unity Engine for Data Analytics

The core of market competition has shifted from fighting for the product market to fighting for the consumer market. Unity has been accepted as an excellent development tool for building interactive systems that can be used to observe customer behaviours (Zhong and Xiao, 2015). Zhong and Xiao (2015) explain that Unity provides a rich networking interface, data communication and data collection. They later describe these networking interfaces as 'well encapsulated', enabling Unity to be utilised in big data collection and analysis.

The growth of information and communication technologies (ICTs) has led to large amounts of data being generated across websites, mobile apps, and social media platforms (Psaltoglou and Vakali, 2021). Similarly, in multiplayer gaming, player actions, in-game events, and

system behaviour produce continuous streams of data that are a good advantage to collect and analyse. Unity's networking and data systems are designed to manage this type of information, allowing developers to track player activity, gather statistics, and use the data to make informed decisions (Unity, 2021). By integrating these features, Unity helps developers organise and visualise data, supporting improvements to gameplay, balance, and user experience.

Beyond tracking and analysing player behaviour, Unity can also serve as a platform for developing more advanced decision support systems in gaming and simulations. The Unity analytics engine has been used to support the creation of reusable analytic models, which can be applied to different tasks without needing to manually rebuild lower-level models (Nachawati *et al.*, 2017). By combining symbolic computation, reusable models, and a structured analytics engine, Unity provides a flexible framework for decision guidance, enabling developers to perform analyses, optimise gameplay mechanics, and make data-backed design choices without much technical development.

4.4 Insights

This section's primary goal is to summarise key elements understood through the literature review. Considering the research findings, it evaluates different key elements which are valuable in the implementation of real-time metric systems and telemetry.

4.4.1 Big Data for Strategic Planning & Execution

Big data in multiplayer games is a form of mass data gathering used in strategic design and embedded within live service game development. The framework proposed by Moorthy et al. (2015) consisting of Business Assessment, Data Assessment, Solution Development, and Insight, aligns closely with the operational structure of modern AAA companies. Business Assessment reflects the identification of competitive goals and long-term strategies, while Data Assessment relates to evaluating telemetry through the 4Vs: Volume, Velocity, Variety, and Value (Parashar, 2013). Multiplayer game environments naturally generate high-volume, high-velocity, and highly varied datasets, including behavioural logs, match statistics, and ranking distributions. As Cai and Zhu (2015) explain, value density decreases as data scale increases, reinforcing the need for filtering and analytical processing. Solution Development and Insight are realised when this data is transformed into dashboards and reports, demonstrating that telemetry systems must be designed to support analytical interpretation rather than functioning solely as storage mechanisms.

4.4.2 Visualisation and Competitive Integrity

Telemetry is most useful in multiplayer games development when data is translated into clear visual graphs that can be used to support interpretation and decision-making. The examples of Rainbow Six Siege's win-delta graphs and rank distributions, alongside Max Payne 3's match reports and heatmaps, illustrate how visual analytics can be used to convert raw datasets into personal user performance indicators for both developers and players. Matthew N. O. Sadiku et al. (2016) emphasise that visualisation opens up pattern recognition and informed thinking. Weiming Zhu (2021) highlights its growing importance within the modern data-driven world.

In competitive multiplayer contexts, visual analytics can operate in two ways: internally and externally. Internally, they allow developers to balance systems and monitor player behaviour to improve development. Externally, they provide players with transparent feedback on performance and progression. This dual function strengthens competitive integrity by ensuring that design decisions are grounded in evidence and that players can clearly understand ranking and matchmaking systems.

4.4.3 RDBMS and NoSQL as Complementary Systems

The comparison between RDBMS and NoSQL database management systems suggests that modern multiplayer telemetry architectures benefit from a hybrid approach rather than reliance on a single structure.

RDBMS provide structured schemas, strong consistency, and transactional integrity, making them suitable for managing player accounts, authentication, inventories, and ranking systems where accuracy is essential (Nance *et al.*, 2013). However, relational systems may struggle with highly dynamic schemas and extremely large-scale telemetry data (Kaur and Rani, 2013).

NoSQL systems, offer flexible schema design, horizontal scalability, and improved performance under high concurrency, making them appropriate for real-time event logging, behavioural analytics, and large-scale data aggregation. Together, these characteristics show that effective multiplayer telemetry systems should separate structured gameplay-critical data from analytical data, integrating relational and non-relational technologies according to their strengths.

5. Research Methodologies

5.1 Research Application

Through the development of a real-time data-integrated artefact, it is important to evaluate prior knowledge to sustain correct practice. Within the literature review, research has been conducted to understand current methodologies for integrating real-time metrics and telemetry, all of which are useful to understand the current processes used across all industries, therefore allowing the application of these methodologies to more (smaller) multiplayer game projects.

As evaluated in the literature review, the artefact is to be created considering the different elements of both relational and non-relational databases. Data that is not relational should be used by a NoSQL database, while data that is relational to users should be used by a relational database. This is to evaluate their efficiency through practice and to showcase how they can be implemented into most multiplayer games.

5.1.1 Unity

Inside the literature review, evaluation of the Unity Game Engine for Multiplayer games development was an important step in understanding why Unity is a good choice for artefact development. Unity provides lots of powerful tools which aid in the development of any multiplayer game. This includes their Unity Services package, which provides various elements that help build multiplayer communities.

In the development of the artefact, the Unity Services *'Networking for Game Objects 2.1'* package should be used within engine version 6000.1.11f1. This features functionality which enables developers to create simple multiplayer experiences using Unity's own modern networking package.

This artefact contains its own backend system; the Unity Services hub is not necessary for the success of the project. The project should be built using WebGL to allow players to access the game through the final website.

5.1.2 PostgreSQL (RDBMS) & MongoDB (NoSQL)

While developing the artefact, using both RDBMS and NoSQL is important in evaluating both relational and non-relational databases for integration into game development. Game analysis

data should be stored through relational means when it refers to a specific user. Relational databases will be used to relationally store data for specific players and their in-game stats.

These stats are stored through private means and are only accessible by participants of the study. This data is accessed through an API that must have special access to the data via private means. PostgreSQL (The PostgreSQL Global Development Group, 1996) & MongoDB (MongoDB Inc., 2009) are used in the development of an artefact as a better-protected, more efficient, and widely adopted variant of an RDBMS and a NoSQL database. Strong passwords must be applied to databases to ensure the protection of any participant data storage.

5.1.3 Express.js Backend

An API must be created to interact between the website, database, and game project. Node.js is a good base for API development - using the *JavaScript* language, widely adopted for how accessible it is. Express.js is a package extended from the Node.js framework, allowing developers to prototype with ease, consisting of lots of simple packages that can assist in setting up quickly.

Express.js should be used to ensure the protection of participant data. The backend must be privately hosted, password-protected, and have limited accessibility globally. Participants must be able to run API functions to start sessions at runtime, access data, and more.

5.1.4 Next.js (graph.js)

Next.js is an appropriate choice for website development in this project as it provides a robust framework for building modern, dynamic web applications. Its support for server-side rendering and static site generation ensures that pages load quickly and efficiently, which is essential for presenting real-time game analytics and player data collected during playtesting.

Furthermore, Next.js simplifies routing, component management, and integration with JavaScript libraries such as Graph.js, allowing the creation of interactive dashboards and visualisations that update automatically as new data is received. This combination of performance, flexibility, and ease of development makes Next.js a practical solution for delivering a responsive and data-driven interface for the artefact.

5.2 Participant Data Collection & Questionnaire

Player data will be collected in two categories: global gameplay data and user-specific performance data. Global data refers to information gathered across all participants and focuses on overall activity within the game environment. This includes the locations on the map where shots were fired, where players were hit, and where deaths occurred. These spatial points help identify common engagement zones, high-risk areas, and general movement patterns across the entire player group. None of this information is linked to individual participants.

User-specific data will also be recorded to capture each participant’s individual performance. This includes the number of deaths, the number of kills, and the total matches played. These metrics provide insight into how each participant interacted with the artefact and help evaluate difficulty, pacing, and engagement on a per-player basis. As with the global data, no personal or identifying information is collected.

After the gameplay session, participants will be presented with a short questionnaire designed to gather their views on data collection within games. The questionnaire will ask whether they noticed any form of tracking during play, how aware they felt of data being recorded, and how comfortable they were with the idea of gameplay data being collected in the background. The questions will be presented clearly and neutrally, encouraging participants to reflect on their

experience without influencing their responses. This allows the study to compare the actual tracking methods used with participants' perceptions of being monitored, offering insight into how unobtrusive the data-collection process felt during gameplay.

6. Results and Findings

This section presents the results obtained from the gameplay telemetry and participant questionnaire conducted during the investigation. All findings are reported factually and without interpretation, focusing solely on the observable outcomes generated during the study. The gameplay data collected from the artefact comprises user-specific performance metrics, providing a clear overview of how participants interacted with the environment. These results include spatial data points, such as where players' deaths occurred, and where shots are fired – as well as user-specific performance indicators such as total matches played, kills, deaths, kill-death ratios, and more. Any tables or graphical representations used to illustrate these findings are referenced within the text and included in the appendix

In addition to the gameplay metrics, this section also presents the results of the post-session questionnaire, which gathered participants' views on the presence and perception of data tracking during gameplay. The findings summarise how aware players felt of background data collection, whether they noticed any tracking mechanisms, and how comfortable they were with the idea of gameplay telemetry being recorded. These results are presented objectively, highlighting a factual representation of responses and any clear trends that emerged from the dataset.

6.1 Metrics and Telemetry Integration & Results

Matches begin once both players are present and end when one player is eliminated, which is recorded as a completed match. Across the investigation, 161 completed matches were logged. Ten participants took part, and the system generated 14 user profiles as players loaded the game. These profiles were created automatically to maintain anonymity while still allowing gameplay activity to be linked to individual sessions.

The telemetry system recorded several categories of gameplay data throughout these matches. Spatial data points were logged for every shot fired, hit registered, and player death. This produced a complete set of coordinates showing where these events occurred across all sessions. Below, *Figures 6-11* showcase the dataset results through participant gameplay. This includes a list of all users generated within the time, and the heatmaps which contain every data point recorded through the study.

Leaderboard

14 players

Kills | Deaths | K/D

#	Player	Kills	Deaths	K/D
	FrostHunter8102	26	11	2.36
	IronBear6682	24	16	1.50
	IronBear8435	19	12	1.58
4	SilentKnight3649	16	24	0.67
5	FierceDragon2227	16	15	1.07
6	ShadowHawk8800	15	16	0.94
7	SwiftHunter1524	14	6	2.33
8	SwiftGhost3377	12	19	0.63
9	ShadowBear3255	11	26	0.42
10	SwiftDragon4230	6	14	0.43
11	SwiftWolf5856	1	0	1.00
12	MightyRaven8547	1	0	1.00
13	MightyBear9667	0	1	0.00
14	ShadowHunter6050	0	1	0.00

Figure 6 Participant User Leaderboard including Kills, Deaths, and Kill Death Ratio.

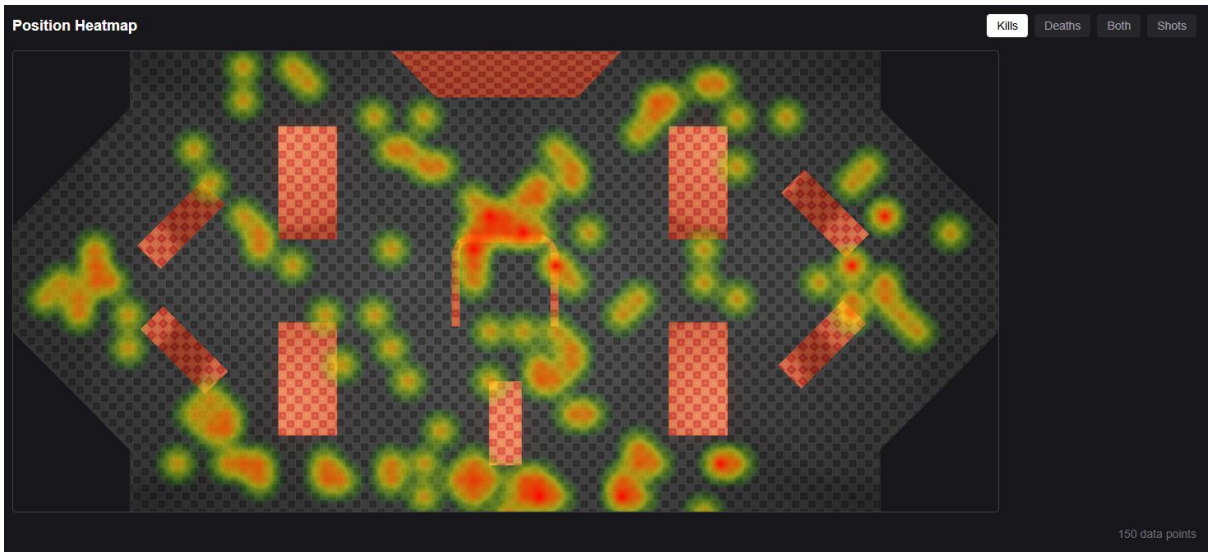

Figure 7 Heatmap of Kill Locations

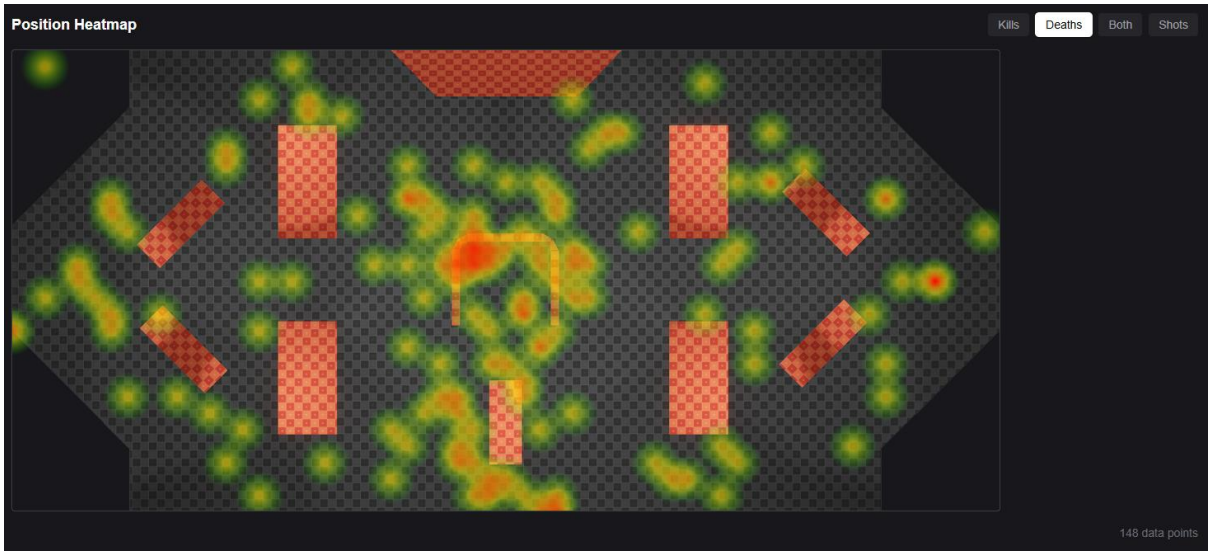

Figure 8 Heatmap of Death Locations

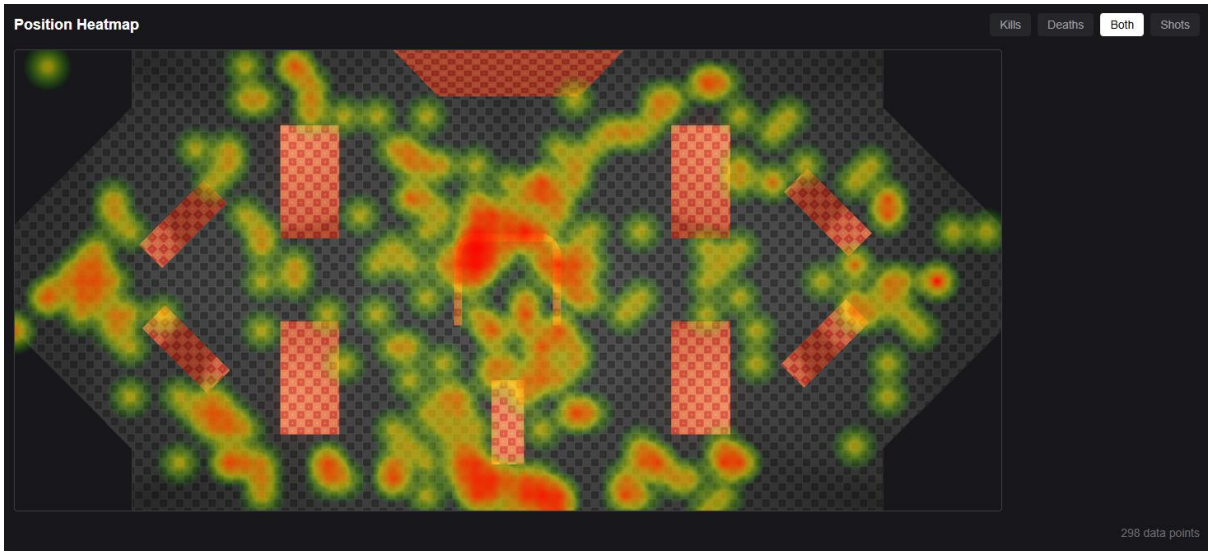

Figure 9 Heatmap of combined deaths and kill locations.

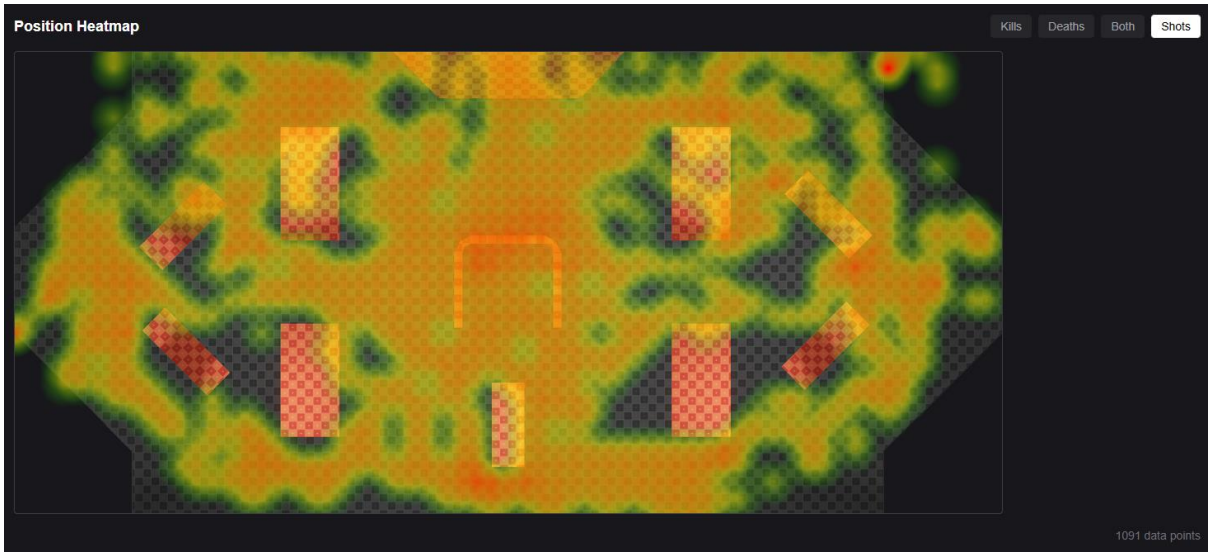

Figure 10 Heatmap for Locations of Shots Fired

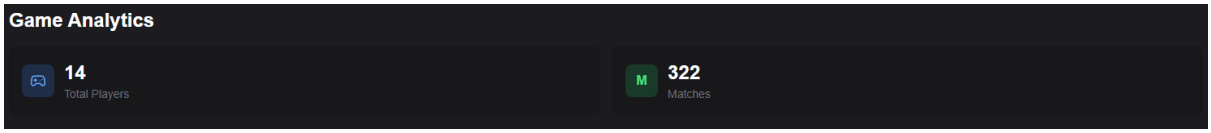

Figure 11 General Analytics throughout the playtest.

User-specific performance data was also collected for each generated profile. This includes the total matches played per user, kills, deaths, and calculated kill-death ratios. The dataset also records repeated participation, showing which users completed multiple matches within the same session. All numerical values and per-user breakdowns are included in the appendix for reference.

In addition to performance and spatial data, the telemetry system logged the number of user creation events, the number of active users per match, and the total volume of interactions recorded during the investigation. These results provide a factual overview of match activity, user participation, and in-game events, forming the complete output of the telemetry system for this study.

6.2 Participant Ethics Questionnaire

Throughout the investigation, participants are asked to complete a questionnaire following their playthrough of the artefact. This questionnaire evaluates the ethics involved in these systems to understand participant-specific thoughts when it comes to the tracking of audience real-time data in videogames.

Would you typically select to have data tracked on videogames?

The first questionnaire item asked participants whether they would typically choose to allow data tracking in videogames. This question was included to establish a baseline understanding of participants' general attitudes towards gameplay data collection before their involvement in the investigation.

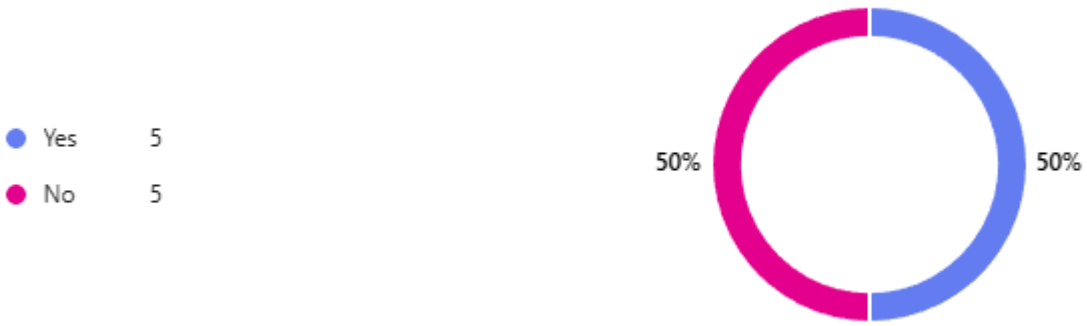

Figure 12 Question 1 Results

Responses indicated a 50/50 split between the participants. Five out of ten participants reported that they usually enable data tracking features when playing videogames, while the remaining five stated that they typically disable this option.

These findings provide an overview of participants' existing behaviours in relation to data tracking and serve as a reference point for questionnaire responses concerning awareness and comfort with telemetry systems used within the artefact.

What makes your decision to enable data tracking?

The second questionnaire item asked participants who reported enabling data tracking to explain the reasons behind their decision. This question was included to record participants' motivations for allowing gameplay data to be collected and to identify any common factors influencing this choice.

5 Responses

ID ↑	Name	Responses
1	anonymous	Well because I understand the value for developers in identifying bugs and optimising performance in games.
2	anonymous	Being able to observe data like this interests me. This kind of data is also extremely useful as a Game Designer and be used to understand how players are interacting with the game.
3	anonymous	Helps developers improve the game
4	anonymous	As long as the data collection is by a trusted source, it wouldn't bother me
5	anonymous	i like feeling like im contributing to projects i like

Figure 13 Question 1.1 Results

Five participants responded to this question. A summary of these responses is presented in *Figure 13* above. All participants explained what they believed the benefits of data collection were for game development and improvement in game companies. Three participants stated that data tracking could help developers in identifying bugs, optimising performance, and improving overall game quality. Two participants specifically highlighted the value of gameplay data for understanding player behaviour and informing game design decisions.

In addition, one participant explained that their willingness to enable data tracking was dependent on how trustworthy they deemed the organisation collecting the data. Another participant indicated that enabling data tracking allowed them to feel that they were contributing positively to projects and games they enjoy.

What makes your decision to disable data tracking?

Within this question, participants who would typically disable data tracking were asked for an insight into their reasoning. This question was included to record participants' concerns regarding gameplay data collection and to identify any common factors influencing this choice.

5 Responses

ID ↑	Name	Responses
1	anonymous	privacy concerns
2	anonymous	Most games don't use recorded data constructively. Further more they don't typically ensure the player base knows why that data is being recorded or what elements of gameplay it could effect.
3	anonymous	like being incognito
4	anonymous	I find it a little too intrusive
5	anonymous	I dont like the idea of my data being retrieved without my knowledge or knowing what it is being used for.

Figure 14 Question 1.2 Results

Five participants provided responses to this question. A summary of these responses is presented in *Figure 14* above. Every participant referred to concerns related to privacy, personal anonymity, and the transparency of data usage. Two participants specifically mentioned worries about how their data is collected and whether it is used constructively by game developers. These participants also stated that games often do not clearly communicate the purpose of data collection or how recorded information may affect gameplay.

In addition, two participants indicated that they preferred to remain anonymous while playing and expressed a desire to remain “incognito” within game environments. One participant described data tracking as intrusive – another reported discomfort with data being collected without clear knowledge of what information was being retrieved or how it would be used.

While playing through the artefact, was the data tracking noticeable in any way?

In this question, participants were asked whether they noticed any form of data tracking during their playthrough of the artefact. This question was included to determine whether the telemetry system operated without being noticed and whether participants were consciously aware that gameplay events were being recorded in real time.

Figure 15 Question 2 Results

All ten participants reported that they did not notice any data tracking activity during gameplay. This response showcases a direct comparison between real game production and the actual tracking methods implemented during the study, giving us a clear record of participant awareness levels without interpreting this as fact.

When playing online multiplayer games, what are your expectations of the data that developers are tracking? (i.e. game data, user data)

This question asked participants what types of data they would expect developers to collect while playing online multiplayer games. The aim was to understand participants' general assumptions about what information is usually recorded during gameplay.

10 Responses

ID ↑	Name	Responses
1	anonymous	I expect developers to track gameplay metrics such as win/loss ratios, in game events and performance data like latency. For user data I'd expect basic hardware specifications, crash logs and connection quality.
2	anonymous	levels, k/d/a, elo, time played, kills per gun, weapon/class usage rates
3	anonymous	They typical contemporary use case for tracked data is marketing, however it can still be used to improve design elements and the overall experience of the game.
4	anonymous	I expect data will be used to improve the game I'm playing in some way. Heatmaps are great for this!
5	anonymous	used anonymously to improve map design/ gameplay
6	anonymous	gameplay data, user data
7	anonymous	I would typically think they are tracking my movements within the game. Maybe more specific things like what weapons & characters I use the most too.
8	anonymous	I would expect they would be tracking most gameplay functions I use
9	anonymous	I assume they track information to do with how i interact with gameplay
10	anonymous	developers are most likely using user data to understand more about their target audience and build the game around that

Figure 16 Question 3 Results

Most participants expected developers to track gameplay and performance data, such as win/loss ratios, kill-death statistics, time played, movement patterns, and weapon or character usage. Several participants also mentioned technical data, including latency, connection quality, crash reports, and hardware specifications.

Some participants also referred to how this data may be used. One participant suggested that data tracking is often linked to marketing, while others focused on how it can help improve game design, map layouts, and overall player experience. Multiple participants mentioned heatmaps and anonymous data being used to support development decisions. Overall, responses show that participants generally expect data collection to be a normal part of online multiplayer games and believe it is mainly used to improve gameplay and understand player behaviour.

How would you expect your data to be used within released products?

This question asked participants how they expected developers to use their data in finished and released games. The aim was to understand participants' views on how collected information should contribute to ongoing development and player experience.

10 Responses

ID ↑	Name	Responses
1	anonymous	I'd expect data to be used for matchmaking improvements, game balancing, identifying common bugs or crash points and even how the data influences the game to have improved build releases.
2	anonymous	to optimise and update the game, to increase player satisfaction
3	anonymous	Ideally the data recorded will be used to better the game and help the developers make patches and iron out small imperfections.
4	anonymous	Data should be used to refine the project before release and be used to back up design decisions.
5	anonymous	improve balance and gameplay
6	anonymous	Look at how players are playing the game and adapt it to the different playstyles to make it more fun to play
7	anonymous	When updating/improving the game.
8	anonymous	Improvements in gameplay and other functionality
9	anonymous	To improve the user gameplay experience.
10	anonymous	id assume they use them to actively understand the current playerbase as we contribute to the information base

Figure 17 Question 4 Results

Most participants explained that they expected their data to be used to improve gameplay, balance, and overall performance. Common responses included using data for matchmaking improvements, game balancing, bug fixing, and optimising updates. Participants also mentioned that data could support regular patches and help refine games after release.

In addition, some participants highlighted the role of data in understanding player behaviour and different playstyles. Some people suggested that developers could use this information to adapt gameplay systems and make games more enjoyable for a wider range of players.

Did you feel that the data tracking presented worked well for the current gameplay?

This question asked participants whether they felt the web telemetry and data tracking systems functioned effectively during gameplay. The aim was to understand how players perceived the performance and usefulness of the telemetry system within the artefact. This is also a good way for developers alike to implement better systems following feedback from existing work.

10 Responses

ID ↑	Name	Responses
1	anonymous	Yes, the data tracking felt seamless and didn't notice a drop in performance or an impact to the overall gameplay.
2	anonymous	yeah
3	anonymous	I feel the way the data was displayed was really insightful and really helped
4	anonymous	Yes, I was able to see data live updating during run time. The system works perfectly for how I would want it to if I were using it to collect data.
5	anonymous	yes
6	anonymous	It worked well, it kept track of how well i performed in the game
7	anonymous	Yeah, I think it worked quite well.
8	anonymous	Yeah, works pretty well
9	anonymous	Worked reasonably well for what Id expect people would use systems like this
10	anonymous	i like the idea of the data tracking system and it has a lot of potential in what it can do for the gameplay

Figure 18 Question 5 Results

Overall, responses were positive, with all participants indicating that the data tracking worked well during gameplay. A couple participance reiterated that they could not tell they were being tracked, stating that it did not affect performance or disrupt their experience. Others mentioned that the live updating data and visual displays were useful and provided helpful insight into gameplay.

Participants also commented on the potential of the system beyond its current state. One participant noted that the system met their expectations for collecting gameplay data. Another highlighted its future potential for enhancing gameplay. Overall, responses suggest that participants found the data tracking system effective and can see the use for such systems in assisting in development or for more personal stat tracking.

In the interest of using these systems to improve your skills, what would you suggest being presented to players?

This question asked participants what types of gameplay data they would like to see if telemetry systems were used to help improve their individual skills. The aim was to understand which forms of feedback players believe would be most useful for supporting learning, reflection, and performance improvement.

10 Responses

ID ↑	Name	Responses
1	anonymous	The response times for my shot to the other play for example the time it took for me to shoot the other play the delay and the reasons to why i was unable to shoot them in the allocated time.
2	anonymous	death heatmap, aka places they die the most in is pretty useful cus that just tells them that they should try another method
3	anonymous	Data on most likely areas to be killed so that players can improve their movement. Potentially show them data of what areas people get the most kills from.
4	anonymous	Data relating to the player directly would be most useful. Being able to see what you might be doing wrong is great for improving your skills. Being able to see where enemies were when they killed you could be useful too. For future development having heatmaps that track where players move could be useful too although consideration would need to be made if this should be shown to players or not.
5	anonymous	most common deaths and where first shots are landed
6	anonymous	Depends on the type of game but something like the system in arc raiders where it shows you where you've been, what you fought and how much damage you did/ enemies did to you is pretty neat. Helps to look at where you may be going wrong.
7	anonymous	Maybe a way of tracking my specific match history - more catered towards me.
8	anonymous	I would suggest something to be able to properly scope how you are compared to other players. Maybe a way to compare yourself to other better players
9	anonymous	Maybe present the data more specifically to me. A dashboard which allows me to login specifically and view my own data would work really well
10	anonymous	i think i could see myself using these systems to develop my skills within the game, especially when starting to understand hotspots for players in mobility. i could also use these systems to understand where i am most at risk.

Figure 19 Question 6 Results

Most participants focused on the value of positional data for improving gameplay. Several participants explained that this type of information could help players understand their poor positioning, adjust their routes through the map, and change strategies when approaching the enemy player.

Although not a feature within this artefact, participants highlighted their desire for personalised performance data. Some suggested systems that allow players to view their own match history, individual statistics, and detailed breakdowns of their performance. Examples included response times, damage dealt and received, first-shot accuracy, and common death locations. These features were seen as useful for helping players understand mistakes and identify specific areas for improvement.

In addition, some participants suggested that comparison tools could support skill development. This included features that allow players to compare their performance with stronger or more experienced players, as well as systems that show how individual performance relates to wider player trends. Others proposed dedicated dashboards where players could securely log in and view their own data over time.

Overall, responses indicate that participants value clear, visual, and personalised feedback systems. Participants suggested that combining heatmaps, performance statistics, and comparison tools could help players better understand their weaknesses, track progress, and use gameplay data more effectively to improve their skills.

7. Discussion and Analysis

7.1 Real-Time Metrics and Telemetry Discussion

The real-time metrics system developed for this project shows how valuable data collection can be when analysing player behaviour. By tracking death locations, kill locations, and shooting activity during playtesting, the system provides a clearer picture of how players move through the map and where major interactions take place.

The heatmaps produced by this system make these patterns easier to see. For example, *Figure 7* highlights a concentration of kills in the centre of the map. This suggests that players naturally engage in this area more than anywhere else. *Figure 8* supports this finding, as deaths are also heavily clustered in the same region. Together, these results show that the centre of the map acts as a major conflict zone where players frequently clash. The fact that both kills and deaths occur there in high numbers shows that the area is contested rather than one-sided.

More importantly, these heatmaps demonstrate how real-time metrics can help designers understand how well different parts of the map are working. Areas with high activity may reveal choke points, strong sightlines, or popular routes. Areas with little activity may suggest that players are avoiding spaces for different reasons. This information is difficult to gather, but important to review; telemetry systems make these trends more approachable and measurable.

These systems also support iterative design. Because the data is collected in real time and in the same way across multiple tests, designers can compare results before and after making changes to the map. If a redesigned area begins to show more balanced behaviour, or if a previous hotspot becomes less overwhelming, the heatmaps provide clear evidence that the changes are influencing gameplay. This enables designers to make decisions based on actual player behaviour data rather than assumptions or feedback.

The heatmap system shows a good example of how real-time metrics can be used to evaluate map layout, identify problem areas, and guide design decisions. It turns players' gameplay into information that helps designers understand how players interact with the environment and whether the map is supporting the intended style of play.

All participants within the investigation reported that they did not notice any data tracking during their playthrough of the artefact. From a functional perspective, this suggests that the telemetry implementation was unobtrusive and did not negatively impact the gameplay experience. While invisible systems support immersion and performance, complete invisibility may contribute to trust concerns. Developers should therefore aim to balance unobtrusive data collection with clarity and disclosure at appropriate points in the user experience to avoid ethical concerns.

7.2 Associated Ethical Discussion

When looking at any data tracking system, it's always important to understand the ethical implications involved with it. Throughout testing of the artefact, participants have been asked questions related to the ethics involved in data tracking systems. This should give developers a better idea of the reasons people disable these systems, thereby identifying opportunities to improve them and effectively entice users to change their typical decision.

Within the first question, participants were asked whether they would typically allow or disallow data tracking systems within the video games that they play. *Figure 12* showcases that 50% of people would typically choose to disable any data tracking. Although reasoning varied, a common denominator between them was the level of trust which is required when allowing game companies to utilise user data.

Building on this, the responses gathered throughout both *Figures 13 & 14* suggest that player attitudes towards telemetry systems are not strictly opposed, but instead dependent on context and transparency. Participants who disabled tracking did not reject the concept of data collection outright; instead, they referenced uncertainty around how their information would be handled and whether organisations could be trusted to use it responsibly. This highlights an important ethical implication for developers. Trust appears to function as the primary gatekeeper for user consent.

Where participants felt that data collection clearly contributed towards improving game quality, optimisation, or balancing, they were more willing to enable tracking. This suggests that ethical telemetry design is not purely a technical challenge, but also a communication challenge.

The findings from participants who typically enable data tracking reinforce this position. These individuals acknowledged the benefits of telemetry for development, particularly for bug fixing, monitoring, and understanding player behaviour. This demonstrates that a portion of the player base already accepts data collection as a normal part of modern game development and its use in the development cycle. For developers, this presents an opportunity to strengthen ethical acceptance by clearly surfacing the direct link between collected data and tangible game improvements. This was an approach to Tom Clancy's: Rainbow Six Siege (Ubisoft Montreal, 2015) in their public development logs – as reviewed within research.

In contrast, the participants who reported disabling tracking consistently raised concerns surrounding anonymity and personal privacy. The desire to remain "incognito" (*Figure 14*) suggests that players may perceive telemetry systems as possibly intrusive when reassurance is not provided. Additionally, concerns about clear communication showcase how the current industry's approach to disclosure may not be fully effective. From an ethical standpoint, simply providing a privacy policy may not be sufficient if players do not feel confident that they understand what is being collected.

Participants already assume that developers track gameplay metrics such as performance statistics, session data, and technical diagnostics. With this expectation already being present within their mind, it indicates how telemetry systems have become largely normalised within AAA multiplayer development. One participant mentions potential marketing uses (*Figure 16*); this could suggest ethical concern when data collection is perceived to extend beyond core gameplay improvement. Developers designing these systems should therefore be cautious and ensure that any secondary data usage is clearly justified and communicated to the population of players.

Responses relating to how players expect their data to be used in released products (Figure 17) were largely aligned with development-focused purposes, including matchmaking improvements, balancing, bug fixing, and post-launch optimisation. This shows that players are generally supportive of telemetry when there is a clear player-facing benefit in mind. When the exchange of data has an emphasis on use for gameplay improvements, users are more comfortable sharing data when they can see how it contributes to their experience.

Feedback on the artefact's telemetry systems was positive; participants did not notice any disruption to gameplay. Some participants also recognised the useful nature of real-time metrics and telemetry on players. This suggests that when telemetry is implemented carefully,

it can coexist with player experience without generating negative press. For developers, this reinforces the importance of ensuring that tracking systems remain performative and non-intrusive.

7.3 Personalised Statistics

A major point of interest throughout the questionnaire was personalised statistics and comparative performance tools (Figure 19) in real-time data game systems. Participants value access to personal data when it is presented in a presentable way. The desire for dashboards and long-term performance tracking further emphasises the importance of telemetry systems for gameplay improvements. Providing players with controlled access to their own telemetry may help turn data collection into something that actively benefits them.

Building on this, the responses suggest that players are more comfortable with telemetry when they feel involved in the process rather than observed by it. When data is only collected in the background, it may contribute towards the trust concerns highlighted earlier. However, when similar data is surfaced back to players through clear visualisations or performance breakdowns, it becomes easier for users to understand the purpose behind the collection. This reinforces the idea that transparency and player-facing value should work alongside technical implementation.

One participant (*Figure 19*) showed interest in comparative performance tools, particularly when used to understand weaknesses or improve decision-making during matches. As mentioned, this indicates that telemetry systems have the potential to support not only developers but also player learning and reflection. If implemented carefully, these systems may shift player perception from passive tracking towards active skill training.

For developers, this highlights the importance of designing telemetry with both internal analysis and external player visibility in mind. Systems that provide optional dashboards may help maintain player trust while still supporting development needs. Overall, personalised and accessible feedback appears to be a key factor in aligning real-time telemetry systems with player expectations and ethical considerations.

Overall, the findings from the investigation suggest that resistance to gameplay telemetry is largely conditional rather than absolute. Ethical concerns typically appear from a lack of trust, transparency, and perceived loss of control. For developers, this highlights several key considerations that need to be evaluated when designing and developing real-time metrics and telemetry systems. Clear and concise communication to a game’s community of consent, optional data participation, visible player benefits, and secure access to personal performance data all appear to contribute positively towards user acceptance. By prioritising these factors, developers may be able to reduce player hesitation and build greater trust in real-time data tracking systems.

8. Conclusion

To conclude, a review of each section is to be utilised to answer the questions presented within the paper's aims and objectives. At the start of the paper, a review of relevant literature on data system integration was conducted, firstly reviewing general big data terminology and modern implementation techniques, before examining how existing organisations implement their own systems to track users' data across the community. This included understanding some of the telemetry systems behind them and showcasing how visual aids are created to inform design decisions about the direction of the game.

Within the early stages of the paper, general business terminology has been understood to better evaluate existing descriptions of big data & data analysis. Research on modern development methodologies evaluated a range of data accumulation techniques.

Relational Database Management Systems (RDBMS) and Non-Relational Database Management Systems (NoSQL) are the two primary methodologies for big data storage structure within the modern day. RDBMSs greatly benefit from structure (Antonio *et al.*, 2025), and are greatly valued for their use of the Structured Query Language (SQL) – a modern Data Manipulation Language (DML) (Ali *et al.*, 2019). Not only SQL (NoSQL) databases primary use for big data (Martinez-Mosquera *et al.*, 2019) – but are also known for their use in improving application productivity. (Choi *et al.*, 2014)

Modern caching solutions such as Redis (Redis, 2009) are a common way for modern game developers to obtain data quickly. GoLang (The Go Programming Language, 2009) is a modern data development language modelled at simplicity and efficiency; its focus is on obfuscating complexity while remaining a leader for data interpretation. Systems such as these are often adopted by developers looking for the fastest data transfers available – especially when dealing with high volumes of user data.

In research, Unity Engine (Unity Technologies, 2026) is analysed in its strengths in data analytics systems. Using Unity's Netcode for GameObjects (Unity Technologies, 2026), an open-source reproduction of industry-standard data tracking systems was created, including an instance of a third-person shooter game, and a web data tracking project that visualises tracked data in real-time. This utilises modern data structures and serves as a manual for the general implementation of data technologies within multiplayer games development.

Through experimentation, multiple data points were collected. This included positional data for kills, deaths, and shots, and user data such as kill counts, death counts, and K.D. This was placed into a NoSQL database, where data could be depicted through a Next.js front end.

Wallner et al. (2014) emphasised the importance of data to assist in the understanding of an ever-evolving community. Accompanied by these systems and experiments, a questionnaire was formed to understand participants' feelings towards data tracking systems, asking questions regarding their typical reaction when prompted to accept or disallow data tracking on games they play. The questionnaire received a direct mixed response, with 50% of users saying they would typically enable data tracking systems, while the rest would not. As shown in *Figures 6-11*, data retrieved started to highlight weak spots within the map.

9. Recommendations

Beware of inaccurate data or obfuscators.

- When tracking with these systems, it is important to verify data collected is correct and accurate
- Through artefact testing, players understood the data implications of their actions and therefore mostly avoided abusing these systems.
- In the development of any metric system professionally, it is important to implement data verification to generate an accurate dataset and avoid obfuscation.

Outline data tracking system goals.

- Before investing time into developing real-time metrics and telemetry systems, understand exactly what you want to learn from them.

- By planning these systems around specific data tracking goals, it makes structuring them a lot easier.
- Develop towards an end goal and ask questions about how these systems can be best implemented within live multiplayer games development.

Importance of Ethics.

- When developing these systems, it is important to share this data tracking interest with the stakeholders of an individual project.
- Shady data tracking requests do not end with positive results and must be considered when developing any real-time metrics system.
- Have an action plan to ensure you are getting people interested in these systems and, therefore, to get the best out of them.

10. Bibliography

Abadi, D.J., Myers, D.S., DeWitt, D.J., Madden, S.R. (2007) *Materialization Strategies in a Column-Oriented DBMS*. Cambridge, MA.

Abbas Fadhel, S., Ali Jameel, E. (2022) A comparison between NoSQL and RDBMS: storage and retrieval. *MINAR International Journal of Applied Sciences and Technology*. **04**(03), 172–184.

Abdelhedi, F., Brahim, A.A., Atigui, F., Zurfluh, G. (2016) Big data and knowledge management: How to implement conceptual models in NoSQL systems'. In *IC3K 2016 - Proceedings of the 8th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management*. SciTePress, pp. 235–240.

Ahamed, B., Ramkumar, T. (2016) Data integration - challenges, techniques and future directions: A comprehensive study. *Indian Journal of Science and Technology*. **9**(44).

Ali, W., Shafique, M.U., Majeed, M.A., Raza, A. (2019) Comparison between SQL and NoSQL Databases and Their Relationship with Big Data Analytics. *Asian Journal of Research in Computer Science*, 1–10.

Amazon AWS (2006) Amazon S3. *https://aws.amazon.com/s3/*.

Angles, R., Gutierrez, C. (2008) Survey of graph database models. *ACM Computing Surveys*. **40**(1).

Antonio, N., Amato, R., Amato, N. (2025) *Comparing XML and Relational Databases: advantages, disadvantages, and use cases for information storage and retrieval*. Edmonton.

Armitage, G., Claypool, M., Branch, P. (2006) Recent Online and Multiplayer Games. In *Networking and Online Games: Understanding and Engineering Multiplayer Internet Games*. Swinburne: Wiley, pp. 0–232.

Baxendale, P., Codd, E.F. (1970) Information Retrieval A Relational Model of Data Large Shared Data Banks. *Communications of the ACM*. **13**(6), 377–387.

Blizzard Entertainment (2004) World of Warcraft.

Cai, L., Zhu, Y. (2015) The challenges of data quality and data quality assessment in the big data era. In *Data Science Journal*. Committee on Data for Science and Technology.

Carpenter, J., Cassandra, E.H. (2016) *Cassandra: The Definitive Guide*. Sebastopol.

Chen, H., Chiang, R.H.L., Storey, V.C., Lindner, C.H., Robinson, J.M. (2012) *Business Intelligence and Analytics: From Big Data to Big Impact Quarterly-Business Intelligence and Analytics: From Big Data to Big Impact*.

Choi, Y.L., Jeon, W.S., Yoon, S.H. (2014) Improving database system performance by applying NoSQL. *Journal of Information Processing Systems*. **10**(3), 355–364.

Choudhary, V., Mehta, A., Patel, K., Niaz, M., Panwala, M., Nwagwu, U. (2024) *Urenna Nwagwu, Integrating Data Analytics and Decision Support Systems in Public Health Management*. New Jersey.

Côrte-Real, N., Oliveira, T., Ruivo, P. (2017) Assessing business value of Big Data Analytics in European firms. *Journal of Business Research*. **70**, 379–390.

Cosic, R., Shanks, G., Maynard, S. (2012) *Association for Information Systems Towards a Business Analytics Capability Maturity Model Towards a Business Analytics Capability Maturity Model*. Geelong.

Costan, C. (2025) *An Overview of Big Data and NoSQL in the Video Game Industry*. Bucharest.

Crawford, G., Gosling, K. V, Light, B. (2011) *Online Gaming in Context*. New York.

Cuzzocrea, A. (2014) Privacy and security of Big Data: Current challenges and future research perspectives. In *PSBD 2014 - Proceedings of the 1st International Workshop on Privacy and Secuirty of Big Data, co-located with CIKM 2014*. Association for Computing Machinery, Inc, pp. 45–47.

Cybulski, J.L., Keller, S., Nguyen, L., Saundage, D. (2015) Creative problem solving in digital space using visual analytics. *Computers in Human Behavior*. **42**, 20–35.

Dynamix (1998) Starsiege: Tribes. *Sierra On-Line*.

Eddelbuettel, D., Balamuta, J.J. (2018) Extending R with C++: A Brief Introduction to Rcpp. *American Statistician*. **72**(1), 28–36.

Ensemble Studios (1997) Age of Empires. *Microsoft*.

Epic Games, Digital Extremes (1998) Unreal. *GT Interactive*.

Glazer, J., Madhav, S. (2015) *Multiplayer Game Programming: Architecting Networked Games*. Indiana.

Goli-Malekabadi, Z., Sargolzaei-Javan, M., Akbari, M.K. (2016) An effective model for store and retrieve big health data in cloud computing. *Computer Methods and Programs in Biomedicine*. **132**, 75–82.

Google (2005) Google Analytics. *https://developers.google.com/analytics*.

Haas, J. (2014) *A History of the Unity Game Engine*.

Hashem, H., Ranc, D. (2016) Evaluating NoSQL document oriented data model. In *Proceedings - 2016 4th International Conference on Future Internet of Things and Cloud Workshops, W-FiCloud 2016*. Institute of Electrical and Electronics Engineers Inc., pp. 51–56.

Hidders, A.J.H. (2001) *A graph-based update language for object-oriented data models*. Eindhoven: Eidenhoven University of Technology.

id Software (1993) DOOM. *GT Interactive*.

Jin, X., Wah, B.W., Cheng, X., Wang, Y. (2015) Significance and Challenges of Big Data Research. *Big Data Research*. **2**(2), 59–64.

Kantamani, R., Kantamani, N., Constantin, R. (2025) Amazon Aurora DSQL for gaming use cases. *AWS Database Blog*.

Kasenides, N., Paspallis, N. (2019) A systematic mapping study of MMOG backend architectures. *Information (Switzerland)*. **10**(9).

Kaur, K., Rani, R. (2013) *Modeling and Querying Data in NoSQL Databases*. 1st ed. California: IEEE.

Lawrence, R. (2014) Integration and virtualization of relational SQL and NoSQL systems including MySQL and MongoDB. In *Proceedings - 2014 International Conference on Computational Science and Computational Intelligence, CSCI 2014*. IEEE Computer Society, pp. 285–290.

Leavitt, N. (2010) Will NoSQL Databases Live Up to Their Promise? *IEEE*, 12–14.

Levene, M., Poulovassilis, A. (1990) *The hypernode model and its associated query language*. London.

Linden Lab (2003) Second Life.

Martinez-Mosquera, D., Lujan-Mora, S., Navarrete, R., Mayorga, T.C., Herrera, H.R.V. (2019) An approach to Big Data Modeling for Key-Value NoSQL Databases. *Revista Ibérica de Sistemas e Tecnologias de Informação*. **19**, 519–530.

Matei, G. (2010) Column-Oriented Databases, an Alternative for Analytical Environment. *Database Systems Journal*. **1**(2), 3.

Matthew N. O. Sadiku, Adebowale E. Shadare, Sarhan M. Musa, Cajetan M. Akujuobi (2016) Data Visualization. . **2**(12).

Menard, Michelle., Wagstaff, Bryan., Smith, Emi. (2015) *Game development with Unity*. Cengage Learning PTR.

Meta Platforms Inc. (2004) Facebook. *Meta Platforms Inc.* [online]. Available from: https://www.facebook.com [Accessed November 10, 2025].

Microsoft Corporation (1989) Microsoft SQL Server.

MongoDB Inc. (2009) MongoDB.

Moorthy, J., Lahiri, R., Biswas, N., Sanyal, D., Ranjan, J., Nanath, K., Ghosh, P. (2015) Big Data: Prospects and Challenges. In *Vikalpa*. SAGE Publications Ltd, pp. 74–96.

Nachawati, M.O., Brodsky, A., Luo, J. (2017) Unity decision guidance management system: Analytics engine and reusable model repository. In *ICEIS 2017 - Proceedings of the 19th International Conference on Enterprise Information Systems*. SciTePress, pp. 312–323.

Nance, C., Losser, T., Iype, R., Harmon, G. (2013) NOSQL VS RDBMS-WHY THERE IS ROOM FOR BOTH. In *SAIS 2013Proceedings*. Georgia: Association for Information Systems, pp. 111–116.

Oracle Corporation (1995) MySQL.

Oracle Corporation (1979) Oracle Database.

Parashar, Manish. (2013) *2013 sixth International Conference on Contemporary Computing (IC3-2013) : 8-10 August 2013, Jaypee Institute of Information Technology, Noida, India*. IEEE.

Psaltoglou, A., Vakali, A. (2021) An exploratory approach for urban data visualization and spatial analysis with a game engine. *Multimedia Tools and Applications*. **80**(10), 15849–15873.

Redis (2009) Redis / Redis Documentation.

Rios, Alex. (2024) *System Programming Essentials with Go*. Birmingham: Packt Publishing Ltd.

Robinson, Ian., Webber, Jim., Eifrem, Emil. (2015) *Graph Databases*. 2nd ed. Marie. Beaugureau, ed. California: O'Reilly Media Inc.

Rockstar Studios (2012a) Compare Max Payne 3 Stats Across Single Player and Multiplayer with the New Social Club Friend Compare Feature. *https://www.rockstargames.com/newswire/article/ak14o883822447/compare-max-payne-3-stats-across-single-player-and-multiplayer-w.html*.

Rockstar Studios (2012b) Max Payne 3.

Rockstar Studios (2012c) New at Social Club: Multiplayer Match Reports. *https://www.rockstargames.com/newswire/article/9k124883873711/new-at-social-club-multiplayer-match-reports.html*.

Rosenblatt, B. (1994) *Unix RDBMS: The next generation What are the Unix relational-database vendors doing to survive in the next generation of client/server environments **.

Rossel, G., Manna, A. (2017) A Modeling methodology for NoSQL Key-Value databases. *Database Systems Journal*. **8**, 12–18.

Samanta, A.K., Sarkar, B.B., Chaki, N. (2018) *Query Performance Analysis of NoSQL and Big Data*. Kolkata: IEEE.

Seguin, K. (2012) *The Little Redis Book*.

Sen, S., Chand, S., Mazumdar, T., Roy, M., Chakraborty, A., De -Asst Librarian, S., Roy, A., Hossain, J. (2015) *Chief Academic Advisor Journal Circulation Layout and Cover Design: Globsyn Management Journal (GMJ)*.

Shaw, A. (2012) Do you identify as a gamer? Gender, race, sexuality, and gamer identity. *New Media and Society*. **14**(1), 28–44.

Shevchenko, V. V. (2025) *Integration of Telemetry into Strategic Management of IT Products*. Kyiv.

Da Silva, M.A.A., Sadovykh, A., Bagnato, A., Cheptsov, A., Adam, L. (2014) JUNIPER: Towards modeling approach enabling efficient platform for heterogeneous Big Data analysis. In *ACM International Conference Proceeding Series*. Association for Computing Machinery.

Singh, S., Kaur, A. (2022) Game Development using Unity Game Engine. In *ICAN 2022 - 3rd International Conference on Computing, Analytics and Networks - Proceedings*. Institute of Electrical and Electronics Engineers Inc.

Stonebraker, M., Abadi, D.J., Batkin, A., Chen, X., Cherniack, M., Ferreira, M., Lau, E., Lin, A., Madden, S., O'neil, E., O'neil, P., Rasin, A., Tran, N., Zdonik, S. (2005) C-Store: A Column-oriented DBMS. In *Proceedings of the 31st VLDB Conference*. Trondheim: VLDB.

Strauch, C., Kriha, W. (2011) *NoSQL Databases Lecture Selected Topics on Software-Technology Ultra-Large Scale Sites*. Stuttgart.

Tauro, Clarence., Patil, Baswanth., Prashanth, K. (2013) *A Comparative Analysis of Different NoSQL Databases on Data Model, Query Model and Replication Model*. Bangalore.

Tencent Cloud (2025) What database does the client game use? *Tencent Cloud*.

The Go Programming Language (2009) Go Programming Language / Go Documentation. *https://golang.org/*.

The PostgreSQL Global Development Group (1996) PostgreSQL.

Tikhomirov, A. (2023) *Degree title Bachelor of Engineering*.

Ubisoft Montreal (2015) Tom Clancy's Rainbow Six Siege. *Ubisoft*.

Ubisoft Montreal (2024) Y9S4 Designer's Notes. *https://store.steampowered.com/news/app/359550/view/4478361902017151024?l=english*.

Unity (2021) Analytics Dashboard. *Unity Documentation*.

Unity Technologies (2026) Unity (6000.1.17f1).

Uzayr, S. bin (2023) *GoLang: The Ultimate Guide*. 1st ed. Boca Raton: CRC Press.

Venkatraman, S., Kaspi, K.F.S., Venkatraman, R. (2016) SQL Versus NoSQL Movement with Big Data Analytics. *International Journal of Information Technology and Computer Science*. **8**(12), 59–66.

Wallner, G., Kriglstein, S., Gnadlinger, F., Heiml, M., Kranzer, J. (2014) Game user telemetry in practice: A case study. In *ACM International Conference Proceeding Series*. Association for Computing Machinery.

Wang, Y., Kung, L.A., Byrd, T.A. (2018) Big data analytics: Understanding its capabilities and potential benefits for healthcare organizations. *Technological Forecasting and Social Change*. **126**, 3–13.

Wasserman, K., Stryker, T. (1980) BYTE Magazine. *McGRAW-HILL*, 24–24.

Weiming Zhu (2021) A Study of Big-Data-Driven Data Visualization and Visual Communication Design Patterns. . (2).

X Holdings Corp. (2006) X. *X*. [online]. Available from: https://x.com [Accessed November 10, 2025].

Yin, S., Ray, I. (2005) *Relational Database Operations Modeling with UML*. Taipei.

Zhong, H., Xiao, J. (2015) *Apply Technology Acceptance Model with Big Data Analytics and Unity Game Engine*.