Technische Universität Berlin

Big Data Management and Analytics
Database Systems and Information Management
Faculty EECS (IV)
Einsteinufer 17
10587 Berlin
http://www.dima.tu-berlin.de

Master Thesis

Positional data modeling and analysis for performance metrics in team sports

Marc Garnica Caparrós

Matriculation Number: 406074
15.08.2019

Supervised by
Prof. Volker Markl
Dr. Ralf Kutsche

Advisors:
Dr. Robert Rein - Prof. Fabian Wunderlich
Dr. Martí Casals - MSc. Jordi Cortés

Deutsche
Sporthochschule Köln
German Sport University Cologne

This dissertation originated from the cooperation with the Technische Universität Berlin (TUB) and the Deutsche Sporthoschule Köln (DSHS).

I would like to thank Prof. Volker Markl and Dr. Ralf Kutsche at TUB as well as the BDMA consortium for the immediate cooperation and allowing me to pursue this master thesis in collaboration with the DSHS in the field of Sports Analytics and Big Data.

Special thanks to Dr. Daniel Memmert for giving me the opportunity to carry out research in this field in the Computer Science in Sports institute at DSHS. Thanks to Dr. Robert Rein and Prof. Fabian Wunderlich for their supervision and guidance as well as all the team in Cologne for the acceptance and great environment.

Thanks Dr. Martí Casals and Jordi Cortés for trusting me as a potential collaborator and starting this journey in the field of Sports Analytics. Hopefully, we can continue this collaboration for a very long time.

Finally, I would like to thank my family and friends for their support and encouragement throughout my study, making this last two years the most outstanding and life-changing years of my life.

Hereby I declare that I wrote this thesis myself with the help of no more than the mentioned literature and auxiliary means.

Berlin, 15.08.2019

. . . . . .

(Signature [your name])

Abstract

Data science is a very interdisciplinary domain that in the last years has gained more impact and presence in so many sectors. With the proliferation of data, data science setups must adapt to the new technologies and Big Data management has become crucial to efficiently deal with the new incoming challenges in data science.

In this thesis, we introduce the topic of Sports Analytics and how sports clubs and research centers are using data science techniques to evaluate their players and understand the game. Studies using numerical metrics and simple statistics in sports have been used for several years. In the last 10 years, data sources and analytical models in sports increased in number and complexity due to the increasing number of data capture techniques and machine learning research. One of the most popular research tracks is the study of positional data in invasion sports: analyzing the game in sports like Football or Basketball by tracking each player and the ball position at a high frequency.

Positional data is a very simple data source containing the position of each tracked object at a certain frequency. The usage of this data source can provide crucial insights into the formation of collaborative teams and their dynamic behavior. Similarly to many other sectors, positional data is collected and provided by so many different organizations such as national leagues or software companies. The high number of providers is making the analytics teams spending most of the time dealing with the different data formats and reading procedures rather than on the statistic pipelines, result discussions, and applications.

We present a simple tool that aims to integrate all the different positional data sources by defining a common and integrated data model. The tool makes use of a very light language to define the structure of each data source and map it to the desired data model. The data source and its definition are ingested by the system to perform the appropriate transformations and load the data in the common repository. This tool tackles the specific challenge in sports positional data sources but aims to be easily replicated to any other sector with similar requirements.

The tool presented provides a single point of access for data sharing and querying. The usage of the tool will speed up the analysis with positional data by serving it in an agreed and standard format regardless of the data source. In the long term, this tool aims to be the base layer of a standard analytic stack for positional data studies providing data, software, and analysis as a service.

Zusammenfassung

Data Science ist eine Domäne, die sich durch Interdisziplinarität auszeichnet und deren Einfluss und Präsenz in vielen Bereichen in den letzten Jahren enorm zugenommen hat. Angesichts der weiten Verbreitung von Daten müssen Data Science Lösungen den neuen Technologien angepasst werden und der Umgang mit Big Data ist zu einer unverzichtbaren Anforderung geworden, um aufkommenden Herausforderungen im Bereich Data Science zu begegnen.

In der vorliegenden Thesis stellen wir das Thema der Sportanalytik und die Nutzung von Data Science Techniken durch Sportvereine und Forschungsinstitutionen zum Zweck der Spielerevaluation und Erforschung des Spiels vor. In den letzten zehn Jahren ist jedoch die Anzahl und Komplexität von Datenquellen und Analysemethoden drastisch gestiegen, was auf die Zunahme von Technologien zur Datenerfassung sowie Forschung im Feld des Machine Learning Lernens zurückzuführen ist. Eines der beliebtesten Forschungsfelder ist die Erforschung von Positionsdaten in invasiven Spielsportarten. Dabei werden Spiele wie Fußball oder Basketball analysiert, indem jeder Spieler sowie der Ball bei mit hoher Frequenz räumlich nachverfolgt werden.

Die Verwendung dieser Datenquelle kann wertvolle Erkenntnisse zur Formation kooperiender Mannschaften und ihres dynamischen Verhaltens ermöglichen. Ähnlich wie in anderen Sektoren, werden auch Positionsdaten von vielen verschiedenen Anbietern, wie nationalen Sportligen oder Software-Unternehmen erfasst und bereitgestellt. Die hohe Anzahl an Anbietern führt dazu, dass Analyseteams die meiste Zeit mit der Verarbeitung verschiedener Datenformate und dem Einlesen der Daten verbringen und wenig Zeit in die statistische Auswertung und Interpretation der Ergebnisse oder Anwendung der Erkenntnisse investieren.

In dieser Arbeit präsentieren wir ein einfaches Werkzeug, das die Integration aller verschiedenen Positionsdatenquellen durch Definition eines allgemeingültigen und integrativen Datenmodells zum Ziel hat. Das Werkzeug verwendet eine leichte Sprache, um die Struktur einer jeden Datenquelle zu definieren und sie auf das gewünschte Datenmodell zu übertragen. Die Datenquelle sowie ihre Definition wird durch das System eingelesen, um die erforderlichen Transformationen durchzuführen und die Daten in die Datenbank zu laden.

Das vorgestellte Werkzeug stellt einen einzelnen Zugriffspunkt für das Teilen und Abrufen von Daten zur Verfügung. Diese Funktion beschleunigt die Analyse von Positionsdaten, indem es sie, unabhängig vom ursprünglichen Datenformat, in einem abgestimmten und allgemeingültigen Format bereitstellt. Auf lange Sicht soll dieses Werkzeug die Basis eines standardisierten Analysemodells für Studien mit Positionsdaten bilden, um Daten, Software und Analysen bereitzustellen.

Contents

List of Figures

List of Tables

Listings

1 Introduction

In all level organizations and research fields, decision making has become a crucial asset. Data Analysis is becoming essential to assess, evaluate, and if possible, improve the performance of the organization in a wide range of sectors.

Nowadays in professional sports; clubs, management teams, coaching staffs, and gambling companies heavily rely on statistical models to predict sports outcomes and results. These models include metrics to measure player ability and team performance as key factors for decision-making with high valued aspects of sports management team design, recruitment or performance evaluation. Since the emergence of the baseball-related Moneyball phenomenon [Lewis, 2003], more and more changes have been introduced in professional sports impacting directly in a huge increase of data sources as well as a larger range of quantity, scope, and sophistication of models and data-oriented developments.

Not surprisingly, the number and level of sophistication of the available metrics in the edge of Sports Analytics have evolved proportionally to the recently advanced data mining and machine learning research as well as the increasing ability to collect more fine-grained data from the sports court. For instance, in the sport of Basketball so far, an exhaustive list of heterogeneous data-driven metrics has been introduced in several research projects like [Jose Sampaio, 2010] and [Franks et al., 2016] to quantitatively measure the productivity and performance of a player as well as the contribution to the team value.

1.1 Data Analytics in invasion sports

Invasion sports or also called collaborative team sports gather all the sports requiring a group of players performing simultaneously complex motions towards the same goal. The common structure of invasion sports includes two teams competing for ball possession in a spatially restricted area. Each team aims to score by putting the ball in the opposite goal while defending their goal, [Gudmundsson and Horton, 2016]. The most popular invasion sports are American Football, Basketball, Soccer, Handball or Hockey.

From a historical point of view, collaborative team sports have been most commonly represented as numerical data and aggregated counters. Management teams and sports analysts have been directly interacting with this data performing end-to-end data analysis while computing results and visualizations. The emerge of improved optical technologies has allowed professional teams in collaborative team sports like Basketball or Football gather team positional data at every single time frame of the game. **Positional** or **tracking data** contains the location of each player and the ball at every time frame in the game within a certain frequency. This data source represented a shift in how

collaborative sports performance was being analyzed and is allowing organizations and research academy introduce new metrics and frameworks to analyze in real-time the execution of the game and the impact of each action on the final goal. The specifications of the different data structures and sources currently used are explained in detail in Chapter 2.

The analysis of these different types of data has become a very promising competitive advantage for sports teams and a limitless source of knowledge. Teams collect and use data from biomedical tests, training activities and games to evaluate their players value, injury prevention, team tactics and performance. The eruption of data in many different formats from structured, semi-structured or video and image analysis made relevant the need for scalable and efficient tools to store and perform complex analysis.

1.2 Problem Statement

The German Sport University Cologne is one of the largest universities in the field of sport and exercise science with 21 academic institutes. The Institute of Training Sciences and Sport Informatics performs multidisciplinary research at the intersection of movement science and computer science, contributing in the area of data analysis regarding Big Data in sports.

One of the main research lines in the Sports Games Research sub-unit of the institute includes the collection, model, analysis and visualization of invasion sports tracking data or positional data. Due to its large network of connections and relations, the institute receives so many sources of data from different sports, countries, competitions, teams or other sports companies.

Research institutions are actively implementing studies looking for new performance metrics and insights to understand the sport better and increase players value and safety. Sports teams and associations rely on private or public studies to perform day-to-day critical decisions. Data access is usually one of the most difficult challenges and researchers need to deal with data ownership and format issues at large scale. Data sources in invasion sports vary their format and accessibility depending mainly on the provider, monitoring device and software manufacturer. Data sources and formats are explained in detail in Chapter 2.

Three main issues are raising in Sports Analytics and were also identified in the daily work of the unit as a consequence of the increasing data volume and heterogeneity. First, studies are usually unrepeatable, causing analytic teams to have isolated projects and dependencies. Ideally, research progress should rely on the concatenation of studies towards a bigger goal. However, each scientist is currently having their own, independent and isolated way to deal with their specific data sources and data transformations. This causes discrepancies in data management, difficulties in knowledge sharing and reusability of the work.

Moreover, studies usually require a bounded list of questions and specific outcomes while superficial exploratory analysis and general definitions are usually hard to obtain and maintain at a large scale for positional data. Data browsing or exploratory data

analysis can be trivial with relatively small data sizes and numerical data, but positional data exploration becomes challenging.

Finally, studies are becoming specific to answer a certain set of research questions or hypothesis but they do not include a framework or methodology to integrate the proposed analysis into a real-life scenario. Repeatability issues are continuously appearing in so many other data science use cases such as biomedical research, geographic data analytics or health-care. Some researchers such as [Baker, 2016] have already raised this concern. They highlighted that research should be a collaborative environment where studies benefit from each other and knowledge is continuously generated by contributing to previous work.

Unfortunately, in real scenarios such as the Sports Game Research unit working with positional data analysis, each scientist in the team develops and maintains their own set of tools, scripts and data sources. As a consequence, progress is hardly shareable and teams produce unhealthy dependencies. The collaborative research environment is probably more established in other research areas such as Computer Science. But in applied data science to sectors such as sports analytics or health-care this approach is not implemented intensively.

The main goal of this thesis is to create an initial tool that could start a switch towards a more collaborative environment where studies not only conclude some hypothesis but also present and standardize procedures and data transformations. This thesis aims to develop a flexible framework for positional data analysis in collaborative team sports. The resulting framework should target the mentioned challenges by providing:

- An integrated view of the data. Dealing with the heterogeneity of the data sources in a flexible and user-based manner.
- Accessible and efficient data serving layer for analysts. Enabling reusable pipelines and standardization of analytical processes.
- Best practices and documentation for data exploration and analysis. Facilitating the applicability of the studies into the research domain and boosting knowledge sharing.

Potentially, the proposed tool will optimize and speed up data collection and storage procedures and will let analysts and sports scientists focus on the actual conclusions.

The final tool should be cross-domain applicable and flexible. Thus, it could also be instantiated by any other data science team dealing with the proliferation of data sources and heterogeneity to optimize their analysis.

1.3 Thesis objectives

This research project focuses on Soccer and Basketball data due to their extensive literature in the research and the relatively higher data availability compared to other sports. The thesis is divided into the following objectives:

(a) **Review and analyze the most accepted and used data structures and metrics in Sports Analytics.** As an introduction, focus on documenting and justifying the evolution of the data analysis structures and their usage in the sports analysis, with the final step on positional data analysis.

(b) **Document the state of the art in positional data modeling in sports.** Establish an extended comparison between the different data models currently being used to represent and model positional data in sports. Define the heterogeneity challenges yielding the overlapping components and the main mappings between different data sources available.

(c) **Implement a flexible and integrated data repository to ingest, store and serve positional data.** The proposed tool should serve as a centralized repository for data collection and be the starting point for further analysis.

(d) **Provide exploratory data analysis for positional data.** Evaluate the proposed tool by providing a light serving layer to enable exploratory analysis of the data. An important goal of this thesis is also to enable the analytics team to interact, modify and optimize the tool with the arrival of new data sources and the design of new projects.

1.4 Research context

This research projects sources from the collaboration of three main advisory groups:

- TU Berlin - EECS Faculty - DIMA Group - Computer Science Academic Advice and Supervision.
- German Sports University of Cologne – Department of Sports Informatics and Sports Game Research – Positional Data Modeling and Analysis.
- External Academic Advice from the Sports Domain: Experts in Basketball Analytics and Sports Performance Metrics.

This research project has been based in Cologne as a research internship with the German Sports University of Cologne.

1.5 Outline

This thesis report is structured as follows.

Chapter 2 describes the related technologies and background used in this thesis. It includes a short overview of sports data analysis and positional data in detail. Finally, it also explores the related work on data integration research and methodologies in Big Data scenarios.

Chapter 3 focuses into documenting the proposed tool defining the different data sources, the data integration mechanism and implemented components.

Chapter 4 gives a detailed view of how the integrated repository can be exploited by the research group to explore positional data.

Chapter 5 concludes this thesis by stating the final contributions of this work, discrepancies with initial goals, potential future work and interesting extensions.

2 Foundations

2.1 Invasion sports data

This section gathers the main evolution of data structures and data models used to capture, document and analyze invasion sports. We focus on the data collection techniques and data structures being studied in the current state of the art in sports. Football and Basketball are two of the most popular invasion sports and gather most of the historical perspective in invasion sports data collection.

Systematic studies of sports data have been documented since the 1950s using manual notation methods [Gudmundsson and Horton, 2016] [Reep and Benjamin, 1968]. As presented in [Memmert and Raabe, 2018], football matches analysis started with quantitative evaluation. Simple data collection techniques were developed to manually gather all the events in the game. Thanks to the homogenization of the sport of football, standard counters and frequencies were easier to extract and analyze to understand the matches. This initial framework evolved to a better qualitative evaluation adding context to the frequencies with the work of experts and assessing match situation. At that moment most of the data collection procedures were manual and required a huge amount of post-game analysis hours.

First analysis and applications to the game of basketball were originated from the numerical statistics gathered through the game and summarized in the Box-Score [Justin et al., 2007], exemplified in figure 2.1. Box-Score related metrics along with the concept of possession in Basketball provides a discrete model representation of players and teams performance by quantifying the positive and negatives actions executed during the game. Adjusted and normalized methods were introduced along with concepts such as efficiency and regularity [Oliver, 2004] and [Mart and Ruiz, 2017] to improve the metrics due to the availability of more detailed play-by-play data where more actions and player decisions were reflected.

	Advanced Box Score Stats														
Starters	**MP**	**TS%**	**eFG%**	**3PAr**	**FTr**	**ORB%**	**DRB%**	**TRB%**	**AST%**	**STL%**	**BLK%**	**TOV%**	**USG%**	**ORtg**	**DRtg**
Bradley Beal	39:42	.652	.614	.409	.182	0.0	5.1	2.5	16.4	2.2	4.8	22.8	29.4	101	106
Trevor Ariza	34:54	.678	.577	.615	.692	8.1	20.5	14.0	19.2	2.5	2.8	10.5	20.6	140	104
Tomas Satoransky	31:29	.500	.500	.286	.000	6.0	0.0	3.1	38.7	0.0	0.0	22.2	10.8	122	115
Jeff Green	27:22	.697	.611	.333	.444	0.0	18.7	8.9	5.0	1.6	0.0	0.0	14.9	152	108
Bobby Portis	26:11	.388	.333	.333	.167	10.8	35.1	22.4	5.0	3.4	0.0	0.0	18.6	104	102

Figure 2.1: Basketball box score example from basketball-reference.com

Hand notation systems were extremely time-consuming. In the 90s, game analysis in

Football and Basketball became digitalized in the majority of the professional leagues. New technical tools and computing power enabled observers and analysts to enter the information into a database for faster analysis. Speech recognition systems and video analysis tools speed up the recording process of events in the game and improved the accuracy of the data recorded. In football, this enabled the analysis to start focusing on physiological and technical evaluation of the game: pass routes, distances, intensities, etc.

Such discretized numerical summaries are providing valuable insights to the current analysis and methodologies but are too dependent on categorized events with a lot of limitations reflected by the academy [Pelechrinis and Papalexakis, 2018] and lose track of key indicators such as player interaction, court distribution or defensive distribution. The emerge of improved optical technologies has allowed professional teams in collaborative team sports like Basketball or Football gather team positional data. This data source represented a shift in how collaborative sports performance was being analyzed and is allowing organizations and research academy introduce new metrics and continuous frameworks to analyze in real-time the execution of the game and each action's impact on the final goal. Real-time metrics computation using optical tracking have been highly active in the past years research with studies like [Cervone et al., 2014] or a similar approach for [Franks et al., 2016]. Now analysts study dynamic tactical evaluations such as patterns, network analysis, interaction or more complex indicators to measure intangibles in sports.

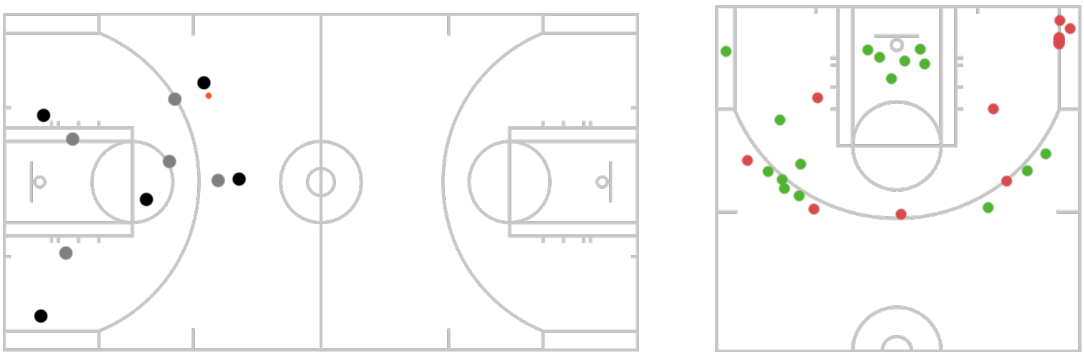
2.2 Positional Data in Sports Analytics

Sports game positional data is a landmark revolution in sports analytics as it tracks every aspect of a player's performance and enables scientific analysis on top of it. [Baca, 2008] and [Baca et al., 2009].

Team positional data is modeled as time series Spatio-Temporal data applied to the execution of a sports game. Spatio-temporal data is a well-defined component in any GIS application and its analysis at scalable and distributed way has been largely addressed by the academy, from a Big Data Analytics point of view in [Alarabi et al., 2018] and [Baumann, 2014] or GIS analysis applied to Sports Analytics in [Kotzbek and Kainz, 2015] and [Hollinger, 2005].

Two categories of Spatio-Temporal data are commonly captured in the sports sector and are illustrated in figure 2.2. Data is composed of object trajectories such as players and ball movement in the shared coordinate system based in the sports game court. These object trajectories are commonly enriched with event logs and detailed descriptions of the game (basketball dribbles or football touches for instance). The trajectories are produced by optical tracking systems and they are dense, uniform and frequent – usually in the range of 10 to 30 Hz. Events might be extracted from the trajectories or manually captured in post-game video analysis. [Thomas et al., 2017]. Events are distributed randomly in the time dimension and they are scarce. On the other side, trajectories are dense in the timeline; at every possible timestamp with a certain fre-

quency, there is a data point for each tracked object. Figure 2.2a shows a snapshot of basketball tracking trajectories, the position of the ball (orange) and each player position for both home (gray) and away team (black) at a certain timestamp in the game. Figure 2.2b shows the location of all shots made by a player in the court as an example of location-based event logs. Each shooting event is tagged with a hit or miss (green or red).



(a) Tracking trajectories (b) Distribution of events

Figure 2.2: Positional data examples in basketball games

Spatio-temporal analysis is currently being studied in different collaborative sports disciplines as a crucial source of individual and collective performance assessment as well as for knowledge discovering and pattern matching in player-to-player interactions, [Kempe et al., 2015] and [Memmert et al., 2017].

2.3 Data Integration: Dealing with data heterogeneity

As introduced in this report, many analytical pipelines in Data Science teams are composed of a set of heterogeneous data sources. Usually, these data sources are managed individually. In the last few years, more and more teams are moving towards better data management techniques and they expressed the need of having an integrated view of their data. Systems need to perform data integration tasks to ingest and query any of the data sources while dealing with different formats and models. The main challenge in data variety is how to seamless integrate all the existing data sources and potentially new ones while keeping the system simple and understandable. Data integration mechanisms enable transparent manipulation across multiple data sources.

2.3.1 Data heterogeneity

Data heterogeneity problems originate from distinct but overlapping characteristics of the data [Bishr, 1998]. In Big Data Science scenarios, data sources are provided usually in structured (database management systems export file), semi-structured (CSV, JSON,

XML) or unstructured (videos) formats, usually expressed as **syntactic heterogeneity**. Similarly, **semantic heterogeneity** refers to the difference between represented concepts by the data sources. Semantic heterogeneity management is a key feature for business models to combine and cross different background for richer and new analysis. Finally, the **schematic** or **terminological heterogeneity** refers to the differences in the data model. Same real-life concepts can be modeled differently depending on the data source or application. Integrated systems must identify the overlapping concepts of two different data models and be able to treat them as a single entity.

2.3.2 Current approaches and related work

Numerous applications are developing data integration mechanisms to gather and manage their data. According to [Chen et al., 2013], research lines trying to solve the data variety problem can be segmented in four big groups.

The first group consists of researchers working on how to involve the user and the wisdom of the crowd to enable systems to understand different sources and their corresponding relation. Secondly, probabilistic methods are being implemented to represent the data uncertainty and predict the schema of unstructured data. Some other contributions accept the fact that a perfect integration system for big data it is not feasible and focuses on practical services to ingest incoming sources. Finally, the last group of contributions focuses on data sources entity matching with real-world concepts for integration.

Ontology-based data integration systems

As a theoretical background, this thesis focuses on data integration systems based on an architecture maintaining a global schema and a set of sources [Lenzerini, 2002]. The implementation proposed is explained in detail in the following chapters.

Developments like [Touma et al., 2015], [Martínez-Fernández et al., 2018] or [Nadal et al., 2019] present ontology-based integration systems defining a *global* and *source schema*. The *global schema* represents the integrated view of the system incorporating entities represented in the data sources as well as new entities. The *source schema* represents per each data source its data model and their *mappings*, allowing the direct transfer of knowledge to the global schema. Data source mappings are metadata artifacts explaining data source models as instances of the global schema.

These integration systems can implement a virtual or physical integration; physical integration is the traditional approach adopted by common Data Warehousing applications were all the data from the sources is physically ingested into a centralized system who runs the storage and data exploitation requirements. Extract and load processes use data mappings to import data into the materialized common repository [Kimball and Ross, 2011]. On the other side, virtual integration avoids a physical copy of the data source and data remains at the sources. Queries into the integrated view are handled by data wrappers and the data source mappings to translate the global query into data source specific sub-queries.

Data mappings are the basic component to understand the integration strategy. Based on their definition, studies include two types of data source mappings [Lenzerini, 2002]:

- **Global-As-View mappings:** Mappings define the global schema as a set of operations on the data sources. Queries into the global view are unfolded into sub-queries to the sources.
- **Local-As-View mappings:** Mappings define each data source model as sub-instances of the global schema. Queries to the global schema need to be rewritten as queries into the sources.

Notation: RDF and JSON-LD

Global and source ontologies are commonly expressed through RDF[1], RDFS[2] and OWL[3] language notations as part of Semantic Web Technologies in knowledge representation for Linked Data. These XML-based formats are highly adopted by the majority of academia. Other developments are extending RDF to avoid the creation of a new serialization format. Instead, they propose a semantic layer on top of the data [Adida et al., 2008]. Enriching data with metadata definitions makes data extraction and intelligent knowledge discovery much easier to implement. JSON-LD[4] was created trying to fill this gap in RESTful web services discovery.

The solution proposed in this thesis takes into account that the target user of the platform does not implicitly know how to design, implement and maintain a functional ontology system. Several tools have been developed to help data scientists model their data through RDF [5] [6], but these systems have a higher learning curve for non-expert people and their maintenance requires specific knowledge for loading and querying the data. While still being aware of new developments in this research line like [Nadal et al., 2018], current tools are not mature enough and they are only valid for a fixed set of data sources. The proposed framework explained in the following chapters includes some tailored load solutions for the feasibility and applicability of the product but introduces the concept of global and source ontologies and leaves to the end-user the flexibility to model and define the data mappings with a simple language based on JSON-LD. It is expected that in the mid-term future this tool could be extended to follow the related work presented in Big Data Integration systems.

[1] https://www.w3.org/RDF/
[2] https://www.w3.org/TR/rdf-schema/
[3] https://www.w3.org/OWL/
[4] https://indienet.info/specs/json-ld-1.0.pdf
[5] http://usc-isi-i2.github.io/karma/
[6] https://github.com/serginf/MDM

3 Integrated positional data repository

In this section, we present the design and implementation of the core tool developed for this thesis along with the German Sports University for Positional Data Analysis in sports. A short of overview and justification of the solution is explained in section 3.1, after that the different data sources integrated into the repository are explained in detail in section 3.2. Finally, the Integration Layer responsible for the ingestion processing and storage of the data sources is presented in section 3.3.

3.1 Requirements overview

Positional data in sports can be expressed in different data formats and models. The overall goal of this project is to provide an integrated upper layer unifying all the existing and possible data sources. This integrated repository will enable the team with a flexible framework for each data source, an initial query point to run standard processes and analytic pipelines. The tool also provides a better understanding and control over the data.

Initially, an agile requirements elicitation was performed to understand the needs of the research group to design and implement the repository:

- The group is formed by several researchers working as Sports Data Scientists.
- Studies are elaborated directly from raw data. Scripts are being developed to manage I/O operations and usually are not shared between the group.
- Researchers propose a small set of research questions to test the data and run different tests to answer them.
- Several programming languages are usually in place: R[1] and Python[2] are the most dominant ones but also using Matlab[3] for iterative numerical analysis. The repository should be easily consumed by any programming language or system.
- Data processing and visualization tools are extremely linked to the programmer and the programming language but there is an increasing desire to plug tools such as Tableau[4] or Data Analysis tools.

[1] https://www.r-project.org/
[2] https://www.python.org/
[3] https://www.mathworks.com/products/matlab.html
[4] https://www.tableau.com/

- In any case, there is no code sharing and standards on how to deal with each data source or how to perform some common processing or visualization tasks. The tool should also help the group standardize and unify processing tasks.
- Data source attributes are highly variable and depend on each provider. The schema of the integrated repository should be flexible and the database management system should allow unknown types and different attributes without restrictions.
- Data from sources is hardly ever modified. However, joining different data sources to enrich the data is needed.
- Data source ingestion and data access are required to be available simultaneously among different members of the team.
- Data analysis is highly variable depending on each study. The repository must serve the data as raw as possible.

The overall project was designed to physically integrate each data source into a flexible data storage engine with a unified data model. This data model will be the entry point for all the research studies regarding positional data providing a single view of the data.

The state of the art technologies for Big Data integration and Semantic modeling presented in chapter 2 have been taken into account to design the data ingestion tasks and integration steps. We have implemented several sessions to introduce the concept of ontology to the research group and the benefits of semantic data management. As the first contribution of this thesis, we provide a conceptual target schema to model positional data in sports.

However, as mentioned in the previous chapter, the current semantic tools still lack usability and they were not suitable for a feasible integration. To simplify and open the tool to the end-user, we decided to implement custom ETL processes that unify the data model for the current existing sources. This ingestion framework has been implemented as flexible as possible in order to easily integrate new data sources in the future. As a result, the integrated repository includes a predefined light data model as a target schema and requires the user to define each data source with a mapping. Mappings are also managed within the data storage engine allowing semantic learning and re-usability of namespaces for new data sources. The ETL process is implemented in a generalized manner and uses both the data source and the appropriate mappings to ingest or update the repository.

Data mappings are global-as-view mappings consisting of queries to perform in the data sources to obtain the target schema entities. In other words, the data mappings translate the data sources into a set of target schema entities. For this thesis, we propose a light custom mapping language based on JSON-LD features. The details of this mapping notations can be found in section 3.3.3. We believe that with the correct maintenance of the target schema definitions and the mappings notation, the system proposed can be easily extended to apply more complex and flexible semantic integration concepts and feed on new developments in this area.

Technology stack

Python was chosen as the main programming language for two main reasons: First of all, for its high popularity in data science teams including the group, which makes the system closer to the user expertise. Secondly, because of the data science and machine learning resources for data ingestion and processing pipelines. To achieve the schema flexibility and generalized processing of different data sources, the system makes use of Apache Spark python library PySpark to deal with the load of files and processing of queries. The schema flexibility of the system makes use of the loading functionalities of PySpark into Dataframes[5] as the standard data structure for all the incoming files. Pyspark Dataframes data model is exploded by means of the data mapping files to keep a flat structure and processed by a common query processor. The query processor obtains the desired data and maps it to the target schema data model. The extracted data is available for import into the repository.

The physically integrated repository is implemented with MongoDB[6] document store due to its flexibility in the schema of the documents, its high availability of analytic tools and its accessibility through high-level communication protocols such as RESTful services. The whole architecture proposed is explained and justified in detail in section 3.3.1.

This document focuses on the research contributions and overall design of the system. For more detail in the Python implementation and software components the source code of the system is available and documented in the source code github repository.

3.2 Data sources

In this section, we present the data sources currently being managed by the research group regarding positional data for football and basketball. We highlight their main differences in size, data models and formats.

- **Trial-based data collection: KINEXON Precision Technologies, KINEXON ONE[7]**

  The institute of Exercise Training and Sport Informatics at the German Sport University Cologne has its own system to record positional data for amateur and professional teams in games, training or experimental set-ups. This enables the team to gather accurate positional data from very different sources and also perform trial-based experiments and record them by means of positional data.

  The recorded data is stored in CSV raw files. At a frequency of 25 frames per second, these CSV files contain each tracked object position as well as other variables such as speed, acceleration or direction of movement in degrees. Each row of the file belongs to one object at a certain time. Time is expressed in milliseconds format and also in a formatted string 'MM/DD/YYYY hh:mm:ss.sss'. To

[5] https://spark.apache.org/docs/latest/api/python/pyspark.sql.htmlpyspark.sql.DataFrame
[6] https://www.mongodb.com/
[7] https://kinexon.com/

print the locations of all the players at a certain time or the trajectory of a single object during a time span this data model will always require some grouping transformations on the data.

The institute has collected more than 230 minutes of positional data from experimental set-ups. This represents approximately 500 MB of data. The current plan involves more than 12 data collection per year to perform test-based analysis.

- **DFL German Football League: Positional Data**
  The institute has also positional data for more than 600 matches from professional soccer matches played in the German Bundesliga. Data is recorded at 25 frames per second for each player and the ball (23 tracked objects at a time). In a professional soccer match, 90 minutes of the game represents more than 135000 data points per tracking object. German Bundesliga official source gathers the data and preprocesses it to finally produce standard XML files of 300 MB of size per match. These files contain for each tracked object all the positions at each frame in a nested XML object. For instance, the file can easily serve the trajectory of a single object during the whole game but would require more processing time to print all the team locations at a certain frame.

- **NBA SportVU Movement tracking data and events**
  The National Basketball Association (NBA) is one of the leaders in research and sports development and was one of the first sport professional leagues who started preparing their facilities to gather as much data from the game as possible. The system has also been tested with NBA raw data from SportVU tracking systems as part of the data provided by STATS LLC. STATS installed optical tracking systems tracking two-dimensional locations of every player and three-dimensional locations of the ball at a resolution of 25Hz. Data was acquired via this repository as part of a student agreement with STATS.

  This source of data provides JSON files describing each game of the 2015-2016 NBA season. Each game file has an average size of 100MB, with a total size of 64000MB for the whole season. Each JSON file groups all the players and ball positions by recorded frame. In this case, the data model is useful to query instant snapshots with the locations of all the players at a certain point in time but will struggle to serve a single object trajectory during a certain period.

For detailed definitions of each data source files and examples, read the annex .1 of this report.

3.3 Integration layer

The Integration Layer is responsible for ingesting the data sources and manage the mappings accurately to perform the physical integration. It includes the management of mappings as well as the validation, processing, and import of the data sources.

3.3.1 Software architecture and components

The system implementation follows a scalable micro-services structure with small and robust modules. The tool can be used from any programming language by using its Application Program Interface (API) for data ingestion and access. Figure 3.1 shows the final deployment proposed for each component and the module interactions.

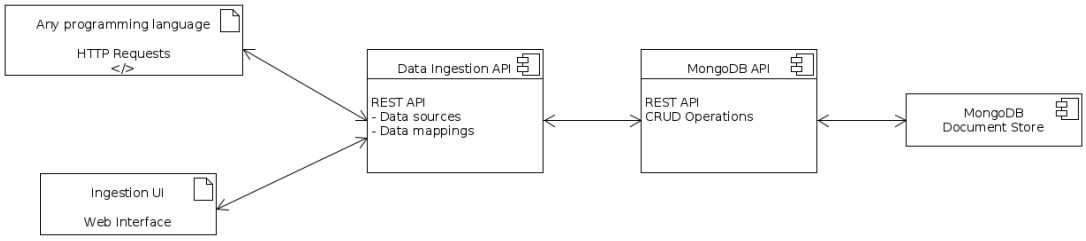


Figure 3.1: Deployment diagram.

The main component is the **Data Ingestion API**. It is a light API implemented with Python Flask[8] and gathers the basic functionality to import new data mappings and new data sources. Data source imports are handled with the appropriate data mapping and translated into the target schema. This data ingestion API can be consumed by any programming language or system able to make HTTP requests. This enables the team members to program bulk imports or pre-process data files before the import into the integrated repository in a flexible manner. As part of the system, we also provide a very light web user interface to easily import new data sources and manage the system mappings graphically.

The **MongoDB storage engine** can be accessed using the usual program interfaces of MongoDB with any programming language or by the MongoDB shell. For better accessibility the system makes use of a highly customizable Python REST API framework to easily build a Web Service for MongoDB, Eve[9]. Eve is encapsulated in the **MongoDB API** component where the target schema of the document store is defined as well as some minor settings restrictions. Eve opens automatically all CRUD operations to the defined entities of any MongoDB setup allowing the user to easily deploy a functional web service for MongoDB storage within seconds. The MongoDB API component will also be reference in Chapter 4 as the main actor in the serving layer.

3.3.2 Integrated conceptual model

The proposed data model aims to represent the main entities present in Positional Data including events and trajectories in sports data. This system allows this data model to be dynamic and changeable for prospective data entries.

[8]https://flask.palletsprojects.com
[9]https://docs.python-eve.org

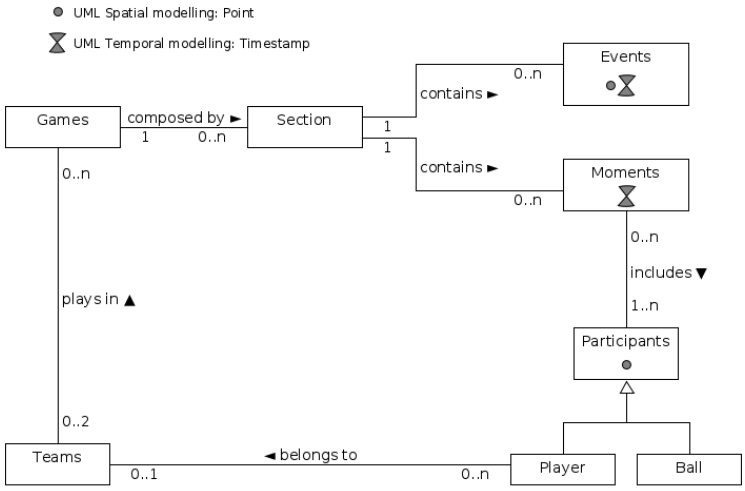


Figure 3.2: UML conceptual modeling of the target schema.

Figure 3.2 represents the basic UML class diagram of the target schema with the entities involved in the integrated view of the data. Class attributes are not defined representing the complete flexibility of schema for each entity of the domain. The main classes are *Events* and *Moments*. The class diagram also models that events are spatio-temporal points (time-series locations) and Moments are timestamped by means of UML extensions in Spatial and Temporal modeling. Each *Moment* includes *Participants*, a general class involving the *Ball* and *Player* in the field. *Participants* are tagged with its location as a point.

The rest of the diagram represents the basic data structure extracted from our study: Players belong to a certain *Team* and teams play in *Games*. Each game is composed by at least one *Section* and each section stores the events and moments at that segment of time. It is important to highlight that the requirements of this study were mainly towards the positional data analysis, for other studies such as teams or games databases this data model presented might not be the most suitable option. The project opens a flexible data structure that can be easily integrated with other specific repositories for richer knowledge discovery.

Target schema instantiation in MongoDB

The conceptual model presented is translated into a physical model in MongoDB document store. This physical model makes sure that no schema restrictions are present in the documents and can be easily managed using the MongoDB component in figure 3.1.

The basic structure of any document in the repository just has two mandatory attributes. The MongoDB-based *_id* attribute to identify each document a custom identifier named *schema_identifier* that is linked with the domain of the repository. In this

instance of the repository the attribute *schema_identifier* will be used by the user to semantically identify the documents. Proper indexing needs to be configured at the database level to achieve good performance when querying by this custom identifier.

The following collections are configured in the repository implementing the conceptual framework presented in figure 3.1.

- **Games:** Information of each game of the database.
- **Teams:** Information of the teams playing in games included in the repository.
- **Participants:** Information of the players involved in the games of the repository.
- **GameSections:** There is at least one game section per each game included in the database. Depending on the user, game sections might be real game parts such as first and second half in Football or some other partitions time or event-based. On each game section, moments and events are defined as nested collections depending on the source of data imported. Moments and events are not expected to be queried individually and the analysts' team always query them by chunks.

Each collection schema is kept as flexible as possible. The ingestion system allows the user to identify how each game section should be partitioned and the partition attribute.

3.3.3 Mapping language

A light pseudo-language is proposed to easily develop data source mappings. Data mappings are JSON files that contain all the information required to load the file into the system and transform the data into the integrated view defined by the target schema in figure 3.2.

A data mapping file consits of two sets of definitions expressed as JSON key attributes:

File information: Definitions of the file format and specifications of the raw data. Two mandatory attributes as *format* and *timestamp_format* referring to the file structured format (JSON, XML, CSV) and the syntax in which the timestamps are expressed in the file respectively. Depending on the *format* attribute, the file information may require more information as it can be seen in the listing 3.1.

```
##File definition for XML files
"file": {
  "format": "xml",
  "rowTag": "Positions",
  "valueTag": "_xml_value",
  "timestamp_format": "yyyy-MM-dd'T'HH:mm:ss.SSSXXX"
}
```

```
##File definition for JSON files
"file": {
  "format":"json",
  "timestamp_format": "ms"
}

##File definition for CSV files
"file": {
  "format": "csv",
  "delimiter": ";",
  "header": "True",
  "timestamp_format": "ms"
}
```

Listing 3.1: Data mapping file defintions examples

Data information: This part of the mapping contains all the different queries to perform to the data source to obtain the desired entities from the files. Data information is stored as a nested JSON object where each key maps directly with a set of elements to extract from the source file. Each set of elements can represent a collection, a nested collection, an array or a nested array. The processor will identify these types and execute the correct query to the file. The value of each set of elements definition are the pointer to the attribute or column in the file containing the information. The only mandatory field to add to each set of collections in this part of the mapping is the schema identifier pointer. A single data mapping file can contain more than one query definition to the same target collection, this features enables extracting information from the same concept in two different query strategies and combine them in the integrated view.

Incoming files can contain nested structures or complex fields that need to be processed correctly by the tool. The data mapping language includes simple methods to identify them:

- Nested attributes in the source file are expressed by a dot like: *attribute.nested_attribute*.
- Nested collections in the source file with a defined schema are expressed by means of '/' like: *array/attribute_of_the_array*.
- Nested collections with no defined schema are expressed like similar programming languages array indexing: *array/[position_in_the_array]*.

If the data section defines custom collections, the resulting document might result in large sizes due to the number of nested elements. To solve that the mapping language enables a custom partition of nested collections by a defined attribute.

The specifications and meaningful examples on the mappings language proposed and syntax can be obtained in .2 taking as reference the data sources of the system.

Semantic learning

The end-user of the system is responsible for manually editting the mappings for each data source. This approach can eventually come up with non-functional integration of data source where each data mapping is using their own namespaces and data elements. This is the reason why the tool includes a light semantic learning module that tries to unify the mappings with two main strategies.

A default mapping has been defined by the team with common data elements and namespace for the basic characteristics of the expected data in the system. This strategy provides a first template for a positional data source to be integrated into the system.

The system is also able to learn from new mappings by keeping a *merged mapping* using all the data elements and names from in all the mappings stored in the database. This merged mapping can also be downloaded and used as a starting point for a new mapping edition that will integrate the new elements with the already used names. The complete format of both the default mapping developed and a mapping recommendation with the data sources presented is also described in section .2 of the appendix.

3.3.4 Ingestion system

The ingestion of data into the system has four main components described in figure 3.3. Initially, the source file and the data mapping file are loaded into memory. The file format is validated for the loading process. Secondly, these files are validated by following the mapping specifications, structures and fields are checked for the data extraction. After that, the target repository is also validated for the data ingestion.

The validation process finally yields a validation report that can be used by the user to understand the potential result of the import. After the validation, the user can launch the final data import where data is extracted from the data source file and processed to map the target repository data model. Finally, data is imported into the MongoDB database.

Data validation

The validation of the data source ensures that the specified data mapping file is matching with the data file characteristics in both dimensions: format and schema. The file is loaded into memory with PySpark loading functionality and the specifications extracted from the mapping file. Using PySpark loading module the schema of the file is inferred and all the pointers and fields used in the mapping specifications are checked to be accessible. This validation process yields a validation report as well as the schema of the source file.

Data extraction and processing

After the validation process, the loaded data structures are processed in order to avoid nested structures and complex data types. With the light syntax expressed in section 3.3.3, the processing engine identifies nested structures or nested arrays in the source

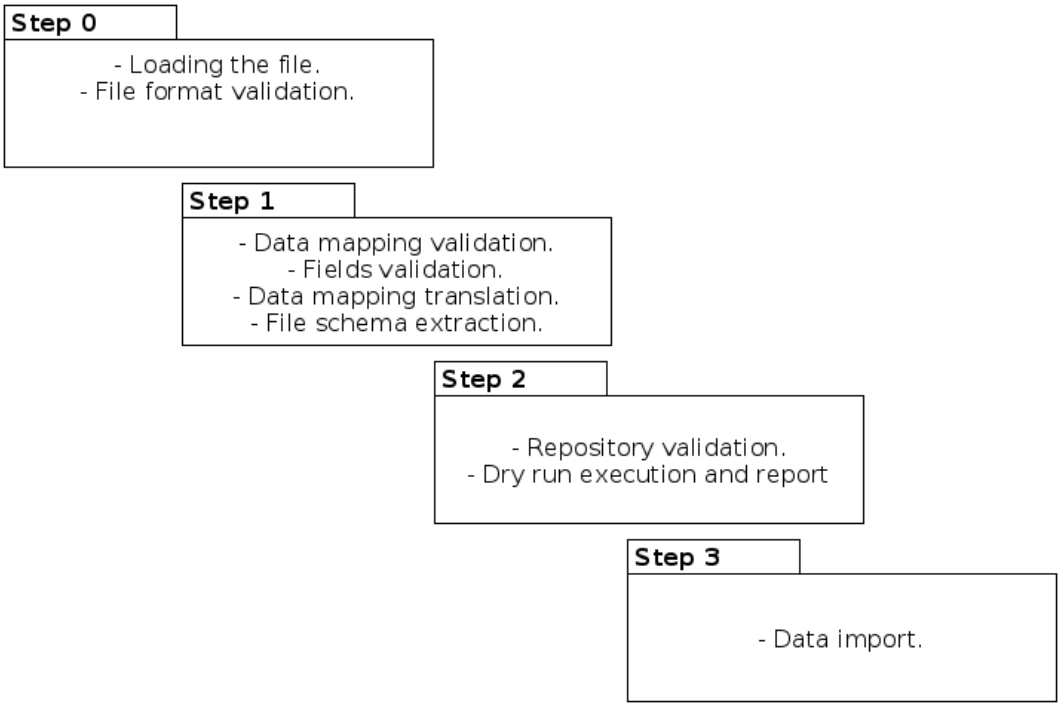


Figure 3.3: Ingestion steps.

file and prepares them for the data extraction by exploding methods to build a flat structure.

The flatten Dataframes are processed by a general query processor to obtain the data specified in the mapping files. In this initial version, the query processor only allows the discovery of a single, first-level nested object and a nested array per each object extracted. Each set of elements to extract defined in the **data** section of the mapping is translated into a query plan as shown in the listing 3.2. The mapping file is used to gather the select attributes, grouping attributes, nested collection attributes, and nested array attributes.

```
SELECT all_select_attributes

if nested_array_exists:
  GROUP BY non_nested_array_attributes
  AGGREGATE nested_arry_attributes into an array

if nested_collection_exists:
  GROUP BY non_nested_collection_attributes
  AGGREGATE nested_collection_attributes into an struct
```

```
if nested_collection_array_exists:
  GROUP BY non_nested_collection_array_attributes
  AGGREGATE nested_collection_array_attributes into an array
```

Listing 3.2: JSON File specifications with ms timestamp

The ingestion system is able to integrate any incoming data source by means of the schema inference of PySpark modules and the data mapping translations into the query plan shown in listing 3.2.

Repository validation

The data extracted from the Dataframes is used against the repository to validate the data import. This step is a dry-run execution of the import that yields a complete report of the documents that will be inserted or updated during the data import. This report is available for the user to have a complete understanding of the process and validate the data extraction process before updating the database.

Data import

Finally, data is imported into the database via the MongoDB API module and the result of each operation is also documented in a report and provided to the user. New data objects are inserted directly into the repository. Already existing data objects are updated increasingly by keeping the previous attributes, updating the overlapping ones and adding the new attributes.

After the data import, the *merged mapping* is also updated by analyzing the new documents and their attributes namespace.

All the steps defined in this section are available to the user as independent executions. As presented in figure 3.1, the user interface allows the user to sequentially execute all the desired steps and review the intermediary reports. More information about the developed user interface for the flexible ingestion of data files is going to be documented in the github repository of the tool.

On the other side, the ingestion steps are also available via the API. The API has been developed to enable team members to perform their own data management techniques and ingest dynamically the files into the repository. Two functions are available through the API:

- **/api/schema**
  Return the schema of the source file.
  Method: POST
  Parameters: Mapping name and source file as input.
  Requirements: The mapping name must exists in the repository.
  Success response: Code 200. Content: Data source schema.
  Error response: Code 500. Content: Empty.

- **/api/validate**
  Load and execute the validation of the file as well as the repository validation.
  Method: POST
  Parameters: Mapping name and source file as input.
  Requirements: The mapping name must exists in the repository.
  Success response: Code 200. Content: Validation report.
  Error response: Code 500. Content: Validation report if available.

- **/api/import**
  Load and execute the entire pipeline. Validating the file and executing the import.
  Method: POST
  Requirements: The mapping name must exists in the repository.
  Parameters: Mapping id and source file as input.
  Success response: Code 200. Content: Validation and import report.
  Error response: Code 500. Content: Validation and import report if available.

3.3.5 Performance

As shown in the listing 3.2, the performance of each data source extractions depends on the target set of elements. Each element defined inside the data section of the mapping is going to perform a query in the data file and an import. Extractions targetting complex structures to build and nested elements will be more time consuming than simple file access and queries. Additionally, the performance of the ingestion system is also linked to the data source schema itself. Nested schemas and complex structures in the data source will impact the execution of exploding and flattening procedures. The result of flattening nested structures will also impact on the final size of the Dataframe to query and will increase the responding time.

This study does not focus directly on the performance of the system but has been designed to scale up the process if needed. PySpark submits spark jobs to a configurable Apache Spark cluster that can make use of the parallelism of tasks to speed up the ingestion. We expect to be able to test this accurately when launching the first production deployments of the tool. All the tests have been run in a Dell XPS 9370 running Ubuntu 16.04 LTS 64 bit with 16GB of RAM and 8 Intel Core i7 CPU.

Table 3.1 shows the average validation time and import time for each of the data sources in this project use case. Each data source is described by the file size, the number of collections that are containing from the target schema, whether the structure of the file is flat or not and the number of nested collections and arrays to build from the source file.

The presented analysis in table 3.1 does not provide full dependencies on the amount of time needed to validate and import some of the data source files regarding source file size and nested structures. After a deeper study, we concluded that the main factor impacting on the cost of each ingestion is the difference between data models. The conceptual model is predefined and strictly fixed for the target repository but each potential data source might come with a different data model and data structures. The

Source	Size (MB)	Collections	Schema	#Nested	Validation time (s)	Import time (s)	DB size (MB)
Kinexon	103	All	Flat	2	79	102	44
DFL	295.4	All	Complex	2	5610	375	104
NBA	103	All	Complex	1	1564	87	54

Table 3.1: Performance analysis.

ingestion time is strictly dependent on how different each schema are. If the processing of each data source requires to completely unfold the data source schema and mount the target data model, the ingestion will be time-expensive.

For instance, German Bundesliga (DFL) data files group data by tracking object: For each tracked object (i.e a player or the ball) it creates nested arrays with all the tracked frames, the position of this object during the whole game 25 frames per second. On the other side, the target model aims to group data by frame: for each frame of the game collects all the tracked objects positions in a nested structure. Clearly, both strategies are orthogonal and the transformation from one to another requires several file scans.

4 Exploratory positional data analysis

This project also aimed to create a flexible framework on top of the integrated repository for exploratory analysis in the analysis of positional data in sports. The integrated repository has been developed to serve as the main entry point for positional data querying and serving positional data to the analyst in different granularities employing the **serving layer**.

To enable exploratory data analysis, the serving layer also includes basic computations and aggregating to speed up the data access process of the data analysis.

4.1 Serving layer

The serving layer is implemented as a REST API in the MongoDB API component presented in figure 3.1. Although this implementation is more specific to the sports analysis use case which has been the focus of this thesis, the serving layer remains flexible to the domain and can be also useful for other framework instances in other sectors.

The serving layer exploits the repository with two types of endpoints: Entity endpoints and aggregated endpoints.

Entity endpoints are basic API resources to access each document of the database. These endpoints are used to query and filter by document. Each defined entity in the target repository can be queried in the following syntax:

- **/api/serving/<entity>**
  Get any document or set of documents from any collection in the database. In the use case of this project entities are: Games — Participants — Teams — GameSections.
  Method: GET
  Requirements: <entity>must exists as a collection in the database.
  Parameters: Query filters are supported for filtering and sorting as well as Mongo DB queries and Python conditional expressions. Any field present in the integrated repository is allowed to be added as a query filter
  Request example: /api/serving/Games?month=Jan or
  /api/serving/Games?where={"month": "Jan"}
  Response: Returns the set of documents matching the specified conditions.

- **/api/serving/<entity>/entity_Id — entity_schema_identifier** Resource endpoint to directly access a document of the database by object id or schema iden-

tifier.
Method: GET
Requirements: <entity>must exists as a collection in the database and the identifier must be present in the collection.
Parameters: -
Request example: /api/serving/Games/e11111 or /api/serving/Games/Match
Response: Returns the specified document.

Aggregated endpoints have been added to provide a transparent layer allowing the data analyst query joined documents and richer information. For instance, for some studies it might be useful to query a certain period of time in the game or to add the players age to their positional data. This use case have been covered by providing a custom **Games** endpoint to the API.

- **/api/serving/aggregated_game/<game_id>**
  Query positonal data from a game with two additional functionalities: Time interval and joining data.
  Method: GET
  Requirements: The game identifier must be an existing identifier in the repository.
  Parameters:
  - gameSectionId = id of the game section to provide. Mandatory.
  - initial = lower bound of the time interval in minutes. Optional.
  - final = upper bound of the time interval in minutes. Optional.
  - join<entity>= 'True' if the result must join with the provided entity fields, with entity being all the available collections in the repository: Games - Participants - Teams. Optional.

  Request example: /api/serving/aggregated_game/DFL-MAT-0002UK? gameSectionId=secondHalfinitial=2final=22joinTeams=True
  Response: Returns the game positional data frame for the specified game resource matching the provided time interval and the additional data specified.

5 Conclusions

In this thesis, we have presented a flexible framework to achieve an integrated repository for positional data in the field of sports analytics. The project has been carried out together with the requirements of the Institute for Computer Science in Sports at the German Sports University of Cologne. This framework is expected to serve as a data homogenization layer as well as a single point of query for data sharing and data access. The complete list of contributions, conclusions and future work planned is presented in the following sections.

5.1 Contributions

Sports analytics is a research track that originated from the need for sports teams to start evaluating their performance more objectively and automatically. We have provided a short overview of the impact of data science techniques for sports performance analysis: from the very simple statistical procedures and data structures to the complex data sources and machine learning pipelines. The emergence of positional data and larger data sources made impossible for the research and clubs to keep traditional methods for data capture, storage, and analysis. The proposed framework in this thesis helps data science teams to quickly deploy an efficient system for seamless data integration into a predefined target model through a simple mapping development.

We have proposed an entire architecture as well as a light mapping language to represent each data source and enable the system understand it. Non-technical users can now define their data sources regarding the target model and easily integrate them into a single storage engine. Data ingestion includes the raw data and the definitions to perform the appropriate transformations and finally integrate them in a common repository. The system has been tested with the current most used data sources achieving total integration.

Furthermore, we have presented a light serving layer for data access. This serving layer enables data scientists and analysts query the desired data in a fix and standard format from a single point of query, without dealing with each data source and each data model. Aggregated methods have been added to this serving layer for better performance specifically to the sports analytics field.

Although this project has been related to the sports analytics use case, the core contribution of this flexible framework aims to serve as a tool for any public or

private research center focusing on data science and statistical analysis. The proliferation of syntactically and semantically heterogeneous data sources is a challenge in so many sectors and Big Data management is part of the solution. The framework is domain-independent and can be instantiated with any target model and data sources.

5.2 Future work

As future work, we plan to incrementally add more positional data sources into the repository from many other national leagues and providers. The tool is ready for early stages of deployment and we are aiming to start an end-to-end performance evaluation to build the appropriate infrastructure. With an accurate setup, the integrated repository managed by the German Sports University of Cologne can develop into one of the most extensive, documented and standard positional data sources in the current state of the art.

In parallel, we are aiming to publish the tool as a domain-independent framework and provide documentation and user manual to instantiate the system in any use case. Several other use cases as bio-medical research gather the same requirements as the use case studied in this project and they are a good example to test the replication of the tool.

5.2.1 Analytics Hub

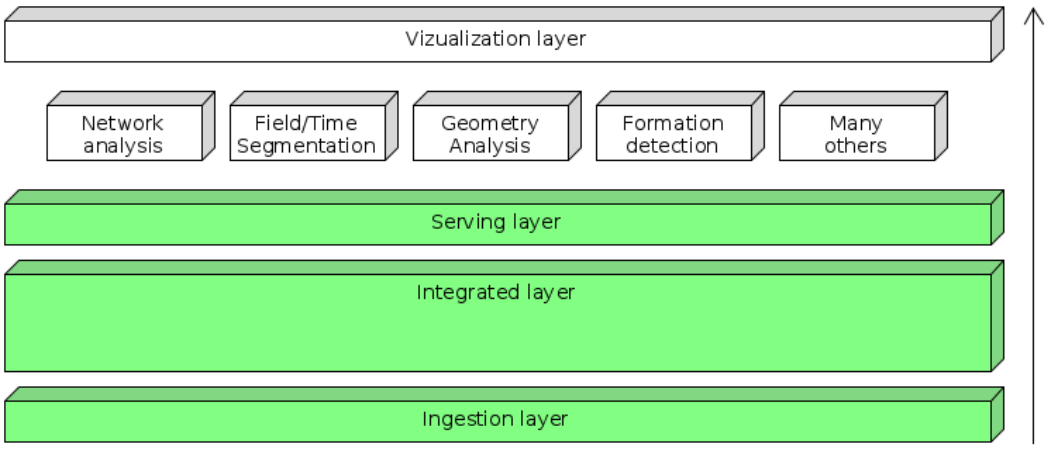


Figure 5.1: Analytics hub complete stack.

The tool presented layer aims to be the initial step into a robust and rich analytics repository for sports positional data. The implementation and deployment of this initial framework will allow the research team to speed up their studies as well as start developing standard procedures and computations for the most popular research algorithms and metrics. In the mid-term, this repository and the upper serving layer can serve not only as a data-sharing point for research but also as a dynamic code repository to share and improve analytical pipelines.

Figure 5.1 shows the desired stack culture which this thesis is trying to kick off. This thesis focuses on the homogenization and integration of the different data sources and the implementation of a light serving layer for data access and exploratory analysis. New studies and research will start developing new procedures to ingest more data and build on top of this layer to provide standard algorithms and procedures for different research paths and applications.

Other data science tracks and domains can also benefit from the integration layer provided in this thesis report and start building their analytics hub. The main goal is to provide **analytics as a service** for knowledge sharing, standardization, and re-usability of studies. This three main features will speed up data science track research and will let the analysts work in the results and applications in the sector.

Bibliography

Ben Adida, Mark Birbeck, Shane McCarron, and Steven Pemberton. Rdfa in xhtml: Syntax and processing. *Recommendation, W3C*, 7:41, 2008.

Louai Alarabi, Mohamed F Mokbel, and Mashaal Musleh. St-hadoop: A mapreduce framework for spatio-temporal data. *GeoInformatica*, 22(4):785–813, 2018.

A Baca. Tracking motion in sport—trends and limitations. In *9th Australasian conference on mathematics and computers in sport, Math Sport (ANZIAM)*, 2008.

Arnold Baca, Peter Dabnichki, Mario Heller, and Philipp Kornfeind. Ubiquitous computing in sports: A review and analysis. *Journal of Sports Sciences*, 27(12): 1335–1346, 2009.

Monya Baker. Is there a reproducibility crisis? a nature survey lifts the lid on how researchers view the'crisis rocking science and what they think will help. *Nature*, 533(7604):452–455, 2016.

Peter Baumann. Rasdaman: Array databases boost spatio-temporal analytics. In *2014 Fifth International Conference on Computing for Geospatial Research and Application*, pages 54–54. IEEE, 2014.

Yaser Bishr. Overcoming the semantic and other barriers to gis interoperability. *International journal of geographical information science*, 12(4):299–314, 1998.

Dan Cervone, Alexander D'Amour, Luke Bornn, and Kirk Goldsberry. Pointwise: Predicting points and valuing decisions in real time with nba optical tracking data. In *Proceedings of the 8th MIT Sloan Sports Analytics Conference, Boston, MA, USA*, volume 28, page 3, 2014.

Jinchuan Chen, Yueguo Chen, Xiaoyong Du, Cuiping Li, Jiaheng Lu, Suyun Zhao, and Xuan Zhou. Big data challenge: a data management perspective. *Frontiers of Computer Science*, 7(2):157–164, 2013.

Alexander M Franks, Alexander D'Amour, Daniel Cervone, and Luke Bornn. Meta-analytics: tools for understanding the statistical properties of sports metrics. *Journal of Quantitative Analysis in Sports*, 12(4):151–165, 2016.

Joachim Gudmundsson and Michael Horton. Spatio-Temporal Analysis of Team Sports – A Survey. 2016. ISSN 03600300. doi: 10.1145/3054132. URL `http://arxiv.org/abs/1602.06994{%}0Ahttp://dx.doi.org/10.1145/3054132`.

John Hollinger. Pro basketball forecast: 2005-2006. *Dulles, VA: Potomac*, 2005.

A Jose Sampaio. Revista Internacional de Derecho y Gestión del Deporte , 10,. (I):1–34, 2010.

Kubatko Justin, Oliver Dean, Pelton Kevin, and Dan Rosenbaum. A starting point for analyzing basketball statistics. *Journal of Quantitative Analysis in Sports*, 3 (3):1–24, 2007. URL `https://EconPapers.repec.org/RePEc:bpj:jqsprt:v:3:y:2007:i:3:n:1`.

Matthias Kempe, Andreas Grunz, and Daniel Memmert. Detecting tactical patterns in basketball: Comparison of merge self-organising maps and dynamic controlled neural networks. *European Journal of Sport Science*, 15(4):249–255, 2015.

Ralph Kimball and Margy Ross. *The data warehouse toolkit: the complete guide to dimensional modeling*. John Wiley & Sons, 2011.

Gilbert Kotzbek and Wolfgang Kainz. Gis-based football game analysis – a brief introduction to the applied data base and a guideline on how to utilise it. 2015.

Maurizio Lenzerini. Data integration: A theoretical perspective. In *Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 233–246. ACM, 2002.

Lewis. Who's on first moneyball: teh art of winning an unfair game. 2003.

Jose A Mart and Manuel Ruiz. Regular point scoring by professional basketball players. (November), 2017.

Silverio Martínez-Fernández, Petar Jovanovic, Xavier Franch, and Andreas Jedlitschka. Towards Automated Data Integration in Software Analytics. 2018. doi: 10.1145/3242153.3242159. URL `http://arxiv.org/abs/1808.05376{%}0Ahttp://dx.doi.org/10.1145/3242153.3242159`.

Daniel Memmert and Dominik Raabe. *Data Analytics in Football: Positional Data Collection, Modelling and Analysis*. Routledge, 2018.

Daniel Memmert, Koen A.P.M. Lemmink, and Jaime Sampaio. Current Approaches to Tactical Performance Analyses in Soccer Using Position Data. *Sports Medicine*, 47(1):1–10, 2017. ISSN 11792035. doi: 10.1007/s40279-016-0562-5.

Sergi Nadal, Alberto Abelló, Oscar Romero, Stijn Vansummeren, and Panos Vassiliadis. MDM: Governing Evolution in Big Data Ecosystems. pages 682–685, 2018. doi: 10.5441/002/edbt.2018.84. URL `http://schema.org/SportsTeam`.

Sergi Nadal, Oscar Romero, Alberto Abelló, Panos Vassiliadis, and Stijn Vansummeren. An integration-oriented ontology to govern evolution in Big Data ecosystems. *Information Systems*, 79:3–19, 2019. ISSN 03064379. doi: 10.1016/j.is.2018.01.006. URL `https://doi.org/10.1016/j.is.2018.01.006`.

Dean Oliver. *Basketball on paper : rules and tools for performance analysis*. Washington, D.C. : Brassey's, Inc, 1st ed edition, 2004. ISBN 1574886878 (hardcover : alk. paper). Includes bibliographical references and index.

Konstantinos Pelechrinis and Evangelos E. Papalexakis. Athlytics: Winning in sports with data. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM 2018, Marina Del Rey, CA, USA, February 5-9, 2018*, pages 787–788, 2018. doi: 10.1145/3159652.3162005. URL `https://doi.org/10.1145/3159652.3162005`.

Charles Reep and Bernard Benjamin. Skill and chance in association football. *Journal of the Royal Statistical Society. Series A (General)*, 131(4):581–585, 1968.

Graham A. Thomas, Rikke Gade, Thomas B. Moeslund, Peter Carr, and Adrian Hilton. Computer vision for sports: Current applications and research topics. *Computer Vision and Image Understanding*, 159:3–18, 2017.

Rizkallah Touma, Oscar Romero, and Petar Jovanovic. Supporting Data Integration Tasks with Semi-Automatic Ontology Construction. pages 89–98, 2015. doi: 10.1145/2811222.2811228.

Annex

.1 Data sources specifications

In this first part of the annex we present in detail each data source used as a base example for the integrated repository. These data sources are currently being studied by the institute of Computer Science in Sports at the German Sports University of Cologne.

- **Trial-based data collection: KINEXON Precision Technologies**
  CSV files downloaded from the data capture system with the following structure:

  root
  - ts in ms : long
  - sensor id : integer
  - mapped id : integer
  - number : integer
  - full name : string
  - group id : integer
  - group name : string
  - x in m : double
  - y in m : double
  - z in m : double
  - speed in m/s : double
  - direction of movement in deg : double
  - acceleration in m/s2 : double
  - total distance in m : double
  - metabolic power in W/kg : double
  - acceleration load : double
  - _c17 : string

  The basic structure of the CSV files allows a simple load into the system but it requires the processing engine to perform groupings by timestamp and

sensor id to gather the target model. Sensor ids cover ball sensors and player sensors.

– **DFL German Football League: Positional Data**
XML files provided by the German Football League for each official match. The schema is the following:

root
- FrameSet : array
  - Frame : array
    - _BallPossession : long
    - _BallStatus : long
    - _M : long
    - _N : long
    - _S : double
    - _T : string
    - _X : double
    - _Y : double
    - _Z : long
  - _GameSection : string
  - _MatchId : string
  - _PersonId : string
  - _TeamId : string
- Metadata : struct
  - PitchSize : struct
    - _X : double
    - _Y : double
  - _MatchId : string
  - _Type : string
- _EventTime : string

These files gathers the information in a more complex structure. For each tracked object represented by *_PersonId* it collects a *Frame* array with all the positions of this object on each frame. For each of this tracked frames it provides the ball status and possession (home or away team) as well as many other numerical variables such as *_M* for the minute of the frame, *_N* for the frame number, *_S* for the speed in km/h, *_T* for the timestamp with format 'yyyy-MM-dd'T'HH:mm:ss.SSSXXX' and *(_X, _Y, _Z)* for the coordinates of the object

– **NBA SportVU Movement tracking data and events**
Positional data from the National Basketball Association for the 2015-2016 NBA season. This data sources contains the following structure in JSON files:

root
- events : array
  - eventId : string
  - home : struct
    - abbreviation : string
    - name : string
    - players : array
      - firstname : string
      - jersey : string
      - lastname : string
      - playerid : long
      - position : string
    - teamid : long
  - moments : array
    - _element : array(string)
  - visitor : struct
    - abbreviation : string
    - name : string
    - players : array
      - firstname : string
      - jersey : string
      - lastname : string
      - playerid : long
      - position : string
    - teamid : long
- gamedate : string
- gameid : string

These files contain one event where they introduce some information on each (home — away). The main concept of this data sources is the array *moments*. Without an explicit structure, this array frames with a timestamp in milliseconds and each player and ball position.

.2 Data mappings syntax

In this section we gather different examples on the mapping language proposed for both file and data information. The data mappings are processed by the ingestion tool to understand each incoming file and extract the data.

```
{
  "name": "",
  "file": {},
  "data": {}
}
```

Listing 1: Data mapping file basic structure

The **file** key of a data mapping contains the information to load the file into the system.

```
{
  "file": {
      "format": "csv",
      "delimiter": ";"
      "header": "True",
      "timestamp_format": "ms"
  }
}
```

Listing 2: CSV File specifications with milliseconds timestamp

```
{
  "file": {
      "format": "xml",
      "rowTag": "Positions"
      "header": "_xml_value",
      "timestamp_format": "yyyy-MM-dd'T'HH:mm:ss.SSSXXX"
  }
}
```

Listing 3: XML File specifications with string timestamp

```
{
  "file": {
      "format": "json",
      "timestamp_format": "ms"
  }
}
```

Listing 4: JSON File specifications with ms timestamp

The **data** key contains all the different set of elements the tool should extract from the source file.

```
{
  "data": {
    Games : [
      {
        "a": "collection",
        "schema_identifier": "game_id"
      }
    ],
    Teams : [
      {
        "a": "collection",
        "schema_identifier": "team_id"
      }
    ],
    Participants : [
      {
        "a": "collection",
        "schema_identifier": "participant_id"
      }
    ],
    GameSections : [
      {
        "a": "collection",
        "schema_identifier": "gamesection_id",
        "game_id": "game_id",
        "moments": {
          "a": "nested_collection",
          "n_id" "moment_id",
          "section_id": "gamesection_id",
          "participants": {
            "a": "nested_array",
            "a_id": "participant_id",
            "coord_x": "x",
            "coord_y": "y",
            "coord_z": "z"
          }
        }
      }
    ]
  }
```

```
}
```

Listing 5: Basic structure of the data specifications: Default mapping

```
{
  "data": {
    "Games": [
        {
          "a": "collection",
          "schema_identifier": "Metadata._MatchId",
          "additional_fields": [
              "Metadata.PitchSize._X", "Metadata.PitchSize._Y"
              ]
        }
      ]
  }
}
```

Listing 6: Extracting games from the German Bundesliga file. Struct access example

```
{
  "data": {
    "Participants": [
      {
        "a": "collection",
        "schema_identifier": "FrameSet/_PersonId",
        "team_id": "FrameSet/_TeamId"
      }
    ]
  }
}
```

Listing 7: Extracting participants from the German Bundesliga file. Array access by attribute name

```
{
  "data": {
    "Participants": [
      {
        "a": "collection",
        "schema_identifier": "FrameSet/[0]",
        "team_id": "FrameSet/[1]"
      }
    ]
  }
```

```
}
```

Listing 8: Extracting participants from the German Bundesliga file. Array access by position

```
{
  "data": {
    "GameSections": [
      {
        "a": "collection",
        "schema_identifier": "dshs#mapping_1",
        "game_id": "dshs#mapping_1",
        "moments": {
            "a": "nested_collection",
            "n_id": "ts in ms",
            "timestamp": "ts in ms",
            "section_id": "dshs#mapping_1",
            "participants": {
                "a": "nested_array",
                "a_id": "sensor id",
                "coord_x": "x in m",
                "coord_y": "y in m",
                "coord_z": "z in m",
                "additional_fields": [
                    "speed in m/s", "direction of movement in deg",
                    "acceleration in m/s2", "total distance in m",
                    "metabolic power in W/kg", "acceleration load"
                ]
            }
        }
      }
    ]
  }
}
```

Listing 9: Extracting game sections and moments from Kinexon files

```
{
  "data": {
    "GameSections": [
      {
        "a": "collection",
        "schema_identifier": "events/moments/[0]",
        "game_id": "gameid",
        "moments": {
```

```
            "a": "nested_collection",
            "n_id": "events/eventId",
            "section_id": "events/moments/[0]",
            "timestamp": "events/moments/[1]",
            "quarter_timestamp": "events/moments/[2]",
            "shot_clock": "events/moments/[3]",
            "participants": "events/moments/[5]"
          },
          "partition" : {
              "interval": "5m",
              "ts_field": "events/moments/[1]"
          }
        }
      ]
    }
}
```

Listing 10: Extracting game sections and moments from NBA files. Partitioning the result in groups of 5 minutes

More syntax and mapping examples can be found in the github repository of the project.