UNIVERSITY OF TROMSØ UIT

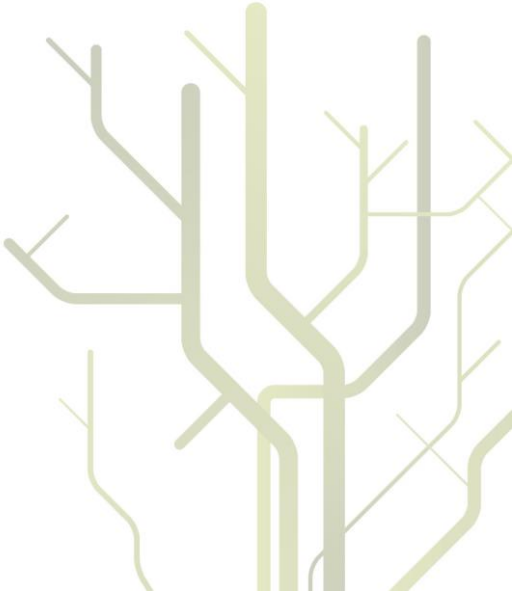
FACULTY OF SCIENCE AND TECHNOLOGY
DEPARTMENT OF COMPUTER SCIENCE

Ruoksat

A system for capturing, storing and presenting the digital footprints of coach knowledge and execution

Kim-Edgar Sørensen

INF-3981 Master's Thesis in Computer Science
June 2013



Abstract

Soccer teams and athletes are constantly looking for possibilities to gain advantages over opponents. In soccer your next opponent is analyzed down to the smallest details to find weaknesses and strengths. All this to be able to take advantage of your opponents weak points and handle their strengths. An example of a typical weakness is a team playing 4-4-2 and gives up space between the lines (back-four and midfield). An strength can be finding this space, typical with a number 10 player.

There are several ways to gather information about your opponent. From looking through whole matches to advanced tools, which can highlight key information for you.There are two main aspect of the analysis process; First you need to gather the information and secondly is how to present the information, usually for the coaching staff.

Acknowledgements

Table of Contents

List of Acronyms

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Sports science is an increasingly hot topic (ref) and more and more teams are investing time and money into strategic development of big data and sports medicine. All this to increase the players performance on the field and reduce the risk of injuries. One of the pioneers in this field for a long time AC Milan established in as early as 2002 the MilanLab. The goal for the program was to get better and more concrete information about the players physicality by collecting data over time of players performance. In 2008 they took it to the next level partnering up with Microsoft to create more specialized software tools for analyze and to organize the data better.

As AC Milan was one of the first to use big data, the use of big data today has exploded. Services like Opta, Prozone and Match Analysis serves player statistics, heat maps, and video analytics of players performance. Fans gets exposed by these statistics by clubs and TV which uses it actively. The awareness of big data statistics for clubs and fans has possible never been higher than now.

Soccer teams and athletes in general are constantly looking for possibilities to gain advantages over opponents. In soccer your next opponent is analyzed down to the smallest details to find weaknesses and strengths. All this to be able to take advantage of your opponents weak points and limit their strengths. MAY REMOVE An example of a traditional weakness is a team playing 4-4-2 and gives up space between the lines (back-four and midfield). An strength can be finding this space, typical with a number 10 player.

There are several ways to gather information about your opponent. From looking through whole matches to advanced tools, which can highlight key information for you. The information out there is enormous. Tools for filtering out the useless data is valuable in a field where the next soccer match usually is in 3 days. There are two main aspect of the analysis process; First you need to gather the information and secondly is how to present the information, usually for the coaching staff.

1.2 Problem definition

This thesis will develop a system complementing the Muithu and Bagadus systems. Focus will be on soccer opponent analytics, where a data repository need to be developed capturing important events relevant for this type of analytics. Specially we want to identify the breakthrough players in a attack. Also a user interface component providing the core information about the opponent should also be developed.

1.3 Interpretation

Our project is that providing an infrastructure for capturing events like attacks that leads to an attempt on the opponent goal, and presenting this information through an user interface. We are interested in how long it typically takes to capture all relevant events from a single match as this has to be done manually. As the system complementing other systems we limit the amount of data captured from each match. A data model that reflects what we want to capture from each attack needs to be developed. This for being able to do relevant queries on the data afterwards.

First a requirement specification shall be developed. Then a prototype and that fulfills the requirements. At last the system shall be evaluated by specialist in the domain of soccer.

The design and development processes will be performed in collaboration with staff of the Norwegian soccer club Trms IL. An end-user comparison of the system against currently available tools will perform a final evaluation, and the result of this evaluation will be used to conclude the thesis.

1.4 Methodology

The final report of the ACM Task Force on the Core of Computer Science [1] divides the discipline of computing into three major paradigms:

- *Theory*: Theory: Rooted in mathematics, the approach is to define problems, propose theorems and look to prove them in order to find new relationships and progress in computing.
- *Abstraction*: Rooted in the experimental scientific method, the approach is to analyze a phenomenon by creating hypothesis, constructing models and simulations, and analyzing the results.
- *Design*: Rooted in engineering, the approach is to state requirements and specifications, design and implement systems that solve the problem, and test the systems to systematically find the best solution to the given problem.

For this project the design process seems to be the most suitable out of the three paradigms. The design process consist of 4 steps and is expected to be iterated when tests reveal that the latest version of the system does not satisfactorily meet the requirements.

- *State requirements and specification*: A need or problem is identified, researched, and defined.
- *Design and implement the system*: Data models and a system architecture are designed. Prototypes are implemented.
- *Test the system*: Assessment and testing of the aforementioned prototypes.

1.5 Outline

- *Item 1*: Presents some background information related to the project
- *Item 2*: Describes the requirement specification
- *Item 3*: Describes the general system model

Chapter 2

Related work

In this chapter we present some background related to analytic in the domain of soccer . We look at types of analysis out there. This spans from pre-match to post-match. Then we go into how data is gathered and presented. There are two main approaches for capturing data: manually often by human annotating it, and automatic capturing often by using sensors.

2.1 Types of analysis

In soccer there are several phases where you use analytic to help you gain insight. Not only the soccer team uses analytics, but TV and fans also.

2.1.1 Prematch

Pre-match you use analytic to find weaknesses and strengths in the opponent team, on a individually level or as a team. You look at your team matched up against your opponent. A typical situation is that the manager gets an video summary of the opponent highlighting the opponents strengths and weaknesses. The video summary is often made up by the coaching staff who may use tools like Interplay.

Typically in TV you have pundits bringing you analytic of key battles during the build up to the game.

2.1.1.1 In match

During match the coaching staff continuously analysis the match and makes adjustment. Of course it is the players who makes all the decisions in the end, but the coach is the boss and most of the time players listen to what hes says. An adjustment to the formation can potentially be the tipping point in the game.

Troms IL uses a system Muithu that lets you annotate sequences of a game with entity's like player, comment. This information will then be time synchronized with the video feed. Later, like in the break or in the game even, you can search on entity's to get the corresponding video feed. As the system is available on an tablets players can in the middle of the match come to sideline to see a involvement. It can be anything that is tagged like a player involvement to a team move.

Using systems like ZXY or MiCoach you can get real time information about a players performance. Rather than guessing that a player is tiering during a game you can get information metrics. You get an evidence based on the metrics you get that the player in fact looks fatigued. In soccer you only have 3 substitutions. Making the right ones is crucial.

Using data gathered from sports data company like Opta you can get statistics live during the game. Its popular in TV to show statistics like ball possession percent, how far players have run or passes played and so on.

2.1.1.2 Post match

During the post match the coach team goes through the game to evaluate the team performance. This is valuable as you get very concrete information about good and bad. You can highlight situations in the match to help players understand tactical aspects.

2.2 Manually capturing

2.2.1 Opta

One of the big players Opta uses manual input to create their data repository. They have editorial teams across the world that captures data manually for

the most popular soccer leagues and other sports. For example to capture statistics for one match, 3 humans have to be involved to be able to annotate all data. The data is captured via an application specifically created for the purpose of capturing data as quick and easily as possible. The editorial teams of Opta need to be able to identify a player, registrate a pass his made or a tackle, in a very short time to be able to keep up with the pace of the game. They study things like which shoe color a player has to be able to quickly identify a player.

Opta capture all types of actions like passes, type of pass, attacks, and interceptions. For each action they log they add a series of description tags like pitch coordinate, player, team and time-stamp. For every single pass they registrate if it was a through ball, normal ball or even a headed flick on from a long ball. For shots they registrate the foot it was kicked with, if it was a volley and so on . All this is done while the match is playing. About 1600 individual events are recorded in a standard match.

```
<Event id="290575408" event_id="5" type_id="1" period_id="1"
min="0" sec="5" player_id="20856" team_id="810" outcome="1"
x="44.6"y="61.1" timestamp="2007-08-12T13:00:24.827"
last_modified="2007-08-12T13:00:25" >
<Q id="1774596260" qualifier_id="141" value="91.6"/>
<Q id="1429253465" qualifier_id="140" value="49.9"/>
<Q id="1084400575" qualifier_id="56" value="Back"/>
</Event>
```

Above is an example of an event registrated in the Opta database. The event has a series of qualifiers describing it. Except from the obvious as timestamp and last modified dates we see that the player id, team id, time of event, x and y coordinates and the outcome of the event are registrated. Also we see that some extra details are included. In this example it maps to a pass from [44.6, 61.1] to [91.6, 49.9].

```
<Q id="1774596260" qualifier_id="coordX" value="91.6"/>
<Q id="1429253465" qualifier_id="coordY" value="49.9"/>
<Q id="1084400575" qualifier_id="56" value="Back"/>
```

2.2.1.1 Prozone

Prozone is a video-based system that tries to track players in team sports. Their data capture system incorporates 8-12 cameras, which is strategically

positioned throughout the stadium to cover 100 percent of the ground, but with some redundancy in case of a faulty component. They also incorporate the TV-camera feed, which always follows the ball. All cameras are hooked into one server and uploaded at the end of the game before sent to undertake the tracking process. As different football grounds houses different pitch sizes the pitch dimensions has to be taken into account by calibrating the cameras. This is done when the system is installed by an operator.

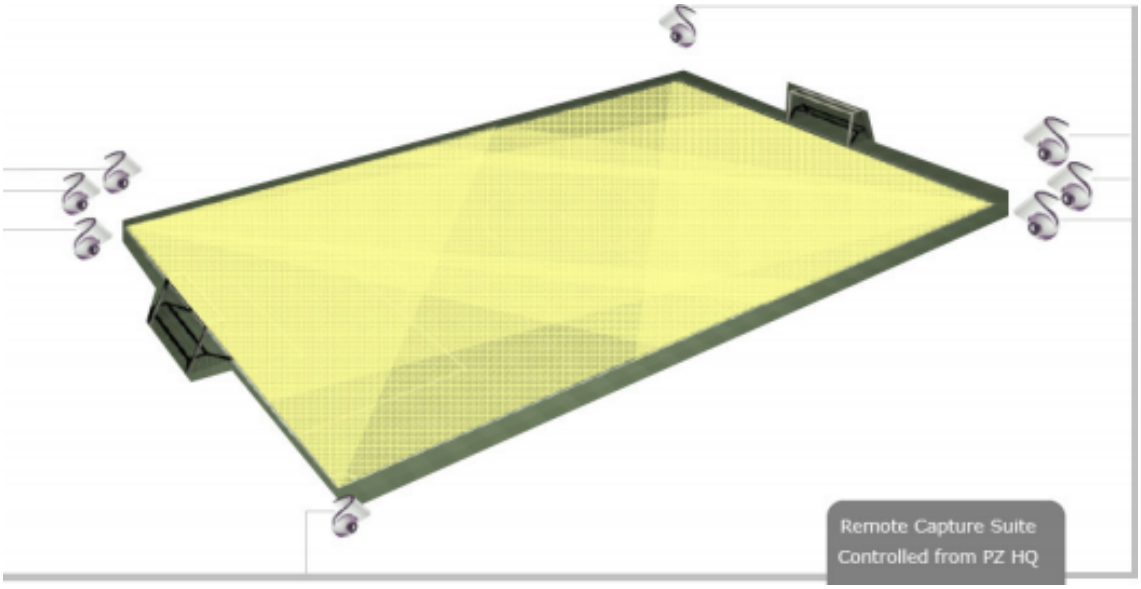


Figure 2.1: The Prozone camera system illustrated

The coding and tracking process of the players movement and actions is a sort of manual process at present time. First each camera feed goes through each own tracking process determining image co-ordinates and continuous trajectories for each player.The output from all the cameras is then combined into one dataset. Going into the algorithm for this is out of the scope of this project. In the final stage the manual work comes into the loop. There has to be a quality control group of operators dedicated for post processing the game. First the operators has to map the players identity and with the their corresponding start location. The operators will then follow the video feed checking that the identification of all players remain constant during the match. The tracking process may run into problems when two players collide or have any other physical contact as it becomes unclear who is who. To map video image co-ordinates to pitch co-ordinates Prozone uses computer vision homography calibration process.

The whole input process is done in a own software which helps you minimize

the amount of manual work for the operators. The software follows rules created by basic machine learning algorithms to validate and verify the input. A simple example would be if the ball goes out of play the system will now that the next event will be a throw in, corner kick or goal kick [4].

Prozone claims to be able to track every movements of every player on the pitch every 10th of a second without using any physical equipment on the players. Di Salvo et al. [3] conducted an empirical evaluation of deployed ProZone systems at Old Trafford in Manchester and Reebook Stadium in Bolton, and concluded that the video camera deployment gives an accurate and valid motion analysis. The data is after a match available through the PROZONE3 interface for analysis.

2.2.2 Automatic capturing

A system that uses sensors is the ZXY sport trackingsystem [**?**]. The system is in used by premier league soccer teams in Norway, including Troms IL and Rosenborg BK. Data captured is stored in Sybase databases with each match requiring about 500-700MB storage The players have to wear a belt around their waist for the system to be able to track their movements. The ZXY system is able to track the players movement very detailed with an accuracy of 0.5m. It has a resolution of 20 samples per second. The technology behind it relies on a radio-based signaling substrate to provide real-time high-precision positional tracking, also including acceleration and heart rate. A installation of receivers is required for the system to work. The home arena for Troms IL, Alfheim, is currently equipped with 10 receivers . A receiver tracks an specific area of the soccer field and combined they cover the whole pitch with some redundancy areas. The communication from the belt to the receivers goes on a 2.45-5.2 G Hz frequency radio signal. To compute the positional data the stationary radio receiver uses an advance vector based processing of the received radio signal. The data is aggregated and stored into a relational database. Including the positions of the players the ZXY also gives you the step frequency and speed.

2.2.3 Wrap up

The main problem with tracking systems that uses physical sensors is that usually only one of the teams wears the sensors. This limits the functionality

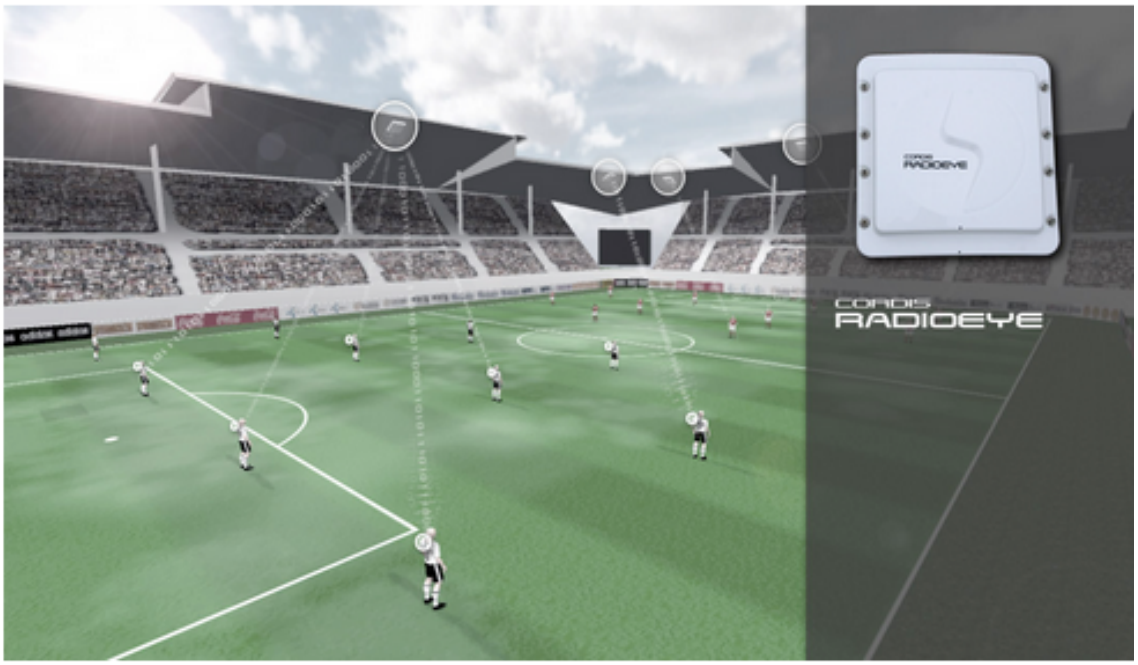


Figure 2.2: Overview of the ZXY system with receivers placed at the stadium and players wearing sensors

of the system as a opponent analysis system. You only get data for one team.

On the other hand you have the manually systems that requires some human annotation. These system are able to track both teams. As they rely on human annotation of some degree they get more rich data as well, but requires strict rules to not mess up the semantics.

2.2.4 Presenting data

This sections gives an short overview of some traditional ways to illustrate opponent analysis.

2.2.4.1 Prozone

Prozone comes with several softwares to illustrate the data. The most relevant for this project is the opposition analysis system.

2D animation Single player analysis Team analysis Pressing analysis Success/direction Player tempo Passing movements Receving the ball Player events

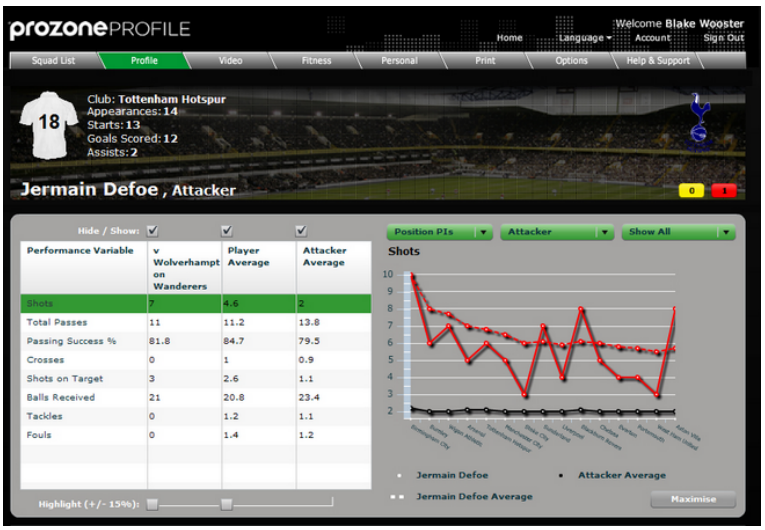


Figure 2.3: The Prozone software - individual player analysis gives you statistics of performance over time

On individual level you can get basic tactics like shots, total passes, passing success, crosses, shots on target, balls received, tackles, fouls.

Doing queries on the data can give you all sprints for a certain player. Players in certain areas of the field have more intensive sprints when they first are involved thus are more vulnerable to injuries. Knowing the actual physical load on players can prevent injuries by regulating the training intensity and amount of time on each exercise.

2.2.4.2 ZXY

ZXY provides you with a 3D graphic interface. This interface lets you see the players action in real time by reading the data stream to reproduce the players action. The data is streamed in real time into the database as the match goes on. While watching you can produce timestamps of different events and produce manual input which is time synchronized with the automatic data. Naturally you can build your own software on top of the Sybase database. Troms IL in collaboration with University of Troms has made several systems to complement the ZXY software. Muithu and Bagadus.

Figure 2.4: The ZXY software - tracking of players gives you information of distance covered, average speed and max speed. You also get a heat map of the players movement on the field.

[2]

Chapter 3

Requirement Specification

This chapter outlines the requirement specification of the system. This section describes the requirement analysis process. The process of gathering the requirements was done in collaboration with Troms IL.

3.1 Overview

The requirements evolved during the process of developing the system. Initially a requirement specification was designed from our perspective. We looked at the different analyze systems out there and what was in use at Alfheim today. Rather than going very wide providing all kind of analyses and statistics we narrowed it down to a very concrete system. A system that tries to to for many things may fell between two chairs, and at the end of the day not providing anything. You spend less time on each feature as you have more features thus reducing the quality on each feature.

The imagined system shall give you the key players in the offensive play of a given opponent soccer team. You shall be able to search on teams and individual players. Additionally you shall be able to see which areas of the pitch players are creating goal chances from. The system also needs a way of capturing data. This shall be an interface that enables you to store successful attacks for any match.

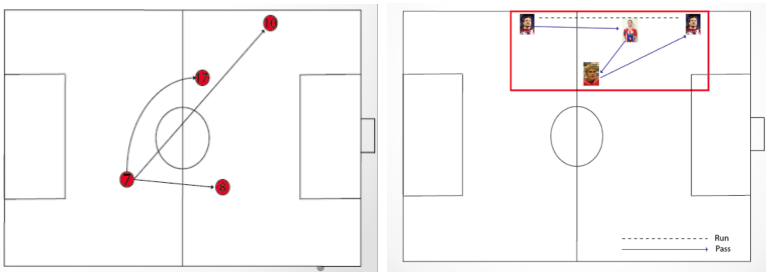


Figure 3.1: Visioned illustration displaying key players and how they combine given you have queried on a team

3.2 Capture

A domain model for the captured data is crucial to set as early as possible in the process. Changes to the model slows the development process. Captured data has to be re-captured and previous queries on data may fail due to missing fields. The domain model should reflect what we want to get out of the system. It sets the boundary's for which information we can pull from the system afterwards. When defining requirements Troms IL gave us a domain model they have used for a previous analytic project. It presented a analytic breakdown of all goals for Manchester United in the 2011/2012 season in Barclays Premier League. This domain model was used became our foundation for our domain model.

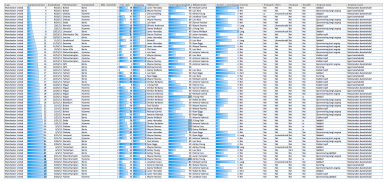


Figure 3.2: A screencapture showing some data Troms IL captured in a previous analytic project

The visioned system need a datastore. The data should be stored in database that has a rich query language. This for being able to support the large range of queries on the data. Good support for aggregate functions is crucial. The data will consist of text and integers.

Additionally a interface for input will be required. This interface should ensure that the input data is correct. For example when defining what type of attack the attack is, only the predefined options should be valid as input.

Initially we wanted to build a database of all the matches in the Norwegian premier league. From each match every successful attack a team makes should be captured. From the attack started you registrate where the attack started, every pass including from/to co-ordinates, and what type of attack the attack can be categorized as. At last if there is a breaking point in the attack this should be captured. The breaking point of the attack is stored with a breakthrough player and what type of breakthrough it was.

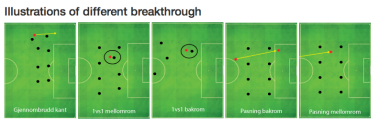


Figure 3.3: Illustrations of different breakthroughs we want to registrate

A definition of what a breakthrough player is: A player that does something extra that unbalance the other team. This can be a dribble past 1-2 players or a genius pass that opens deference of the opponents team.

First problem was how to divide the pitch into zones. As we are looking for which zones the breaking point of the attack this is crucial for the searches on the data captured later on. During the development of the system several types of dividing was presented to the coaching staff.

3.3 Present

The presentation of the data is a important aspect of the system. This will be where the end-users will spend their time. The presentation of data needs to be as simple as possible to understand. The coaching staff is no technical experts. The system is aimed for them to use. Therefor the system will need to present statistics and other valuable information in clean and understandable way. In the end it is the 11 players the coach selects that performs all the actions, but the coach sets the style of play boundary's for players. The goal is to give the coach the opportunity to use the system, both analytic illustrations and statistics provided in the system, so he can better reach out to his players with his ideas. The better a coach can sell his ideas to the players the more they will believe in the philosophy, style of play and follow the game plan when out on the field.

We purpose a web interface here as it is accessible from many devices as long as you have a Internet connection and a up to date web browser.

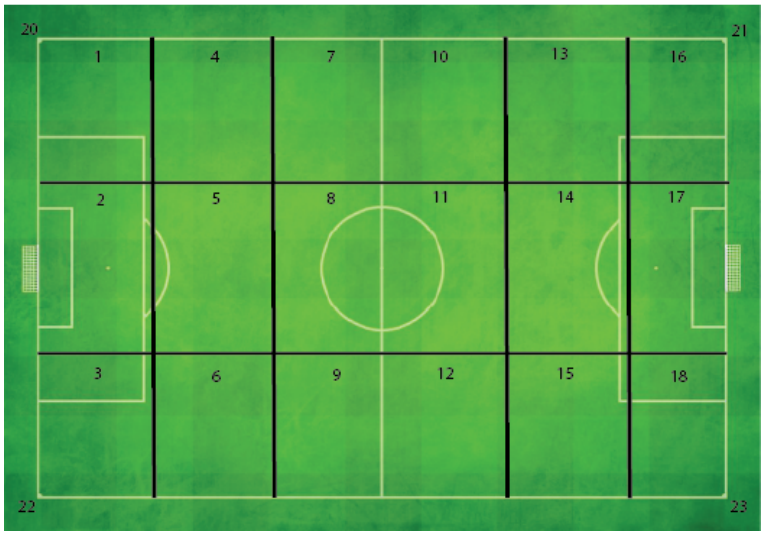


Figure 3.4: Dividing of pitch - the first suggestion had 18 zones

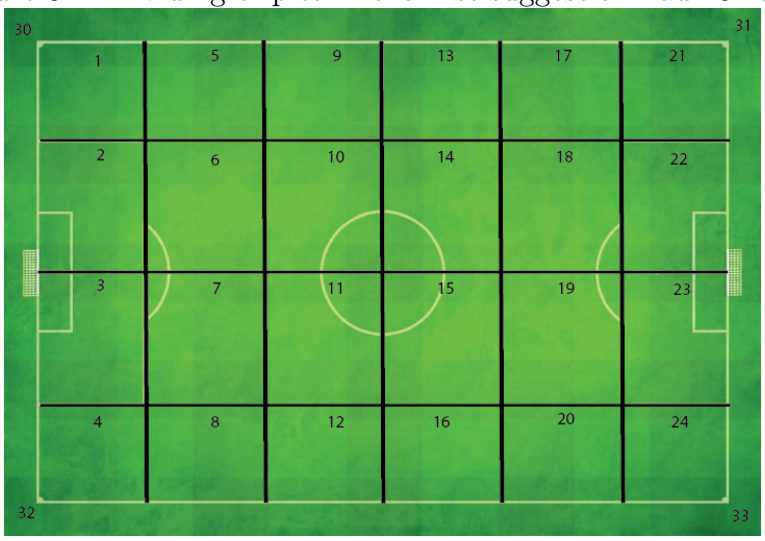


Figure 3.5: Dividing of pitch - the second suggestion had 24 zones

3.4 Summary

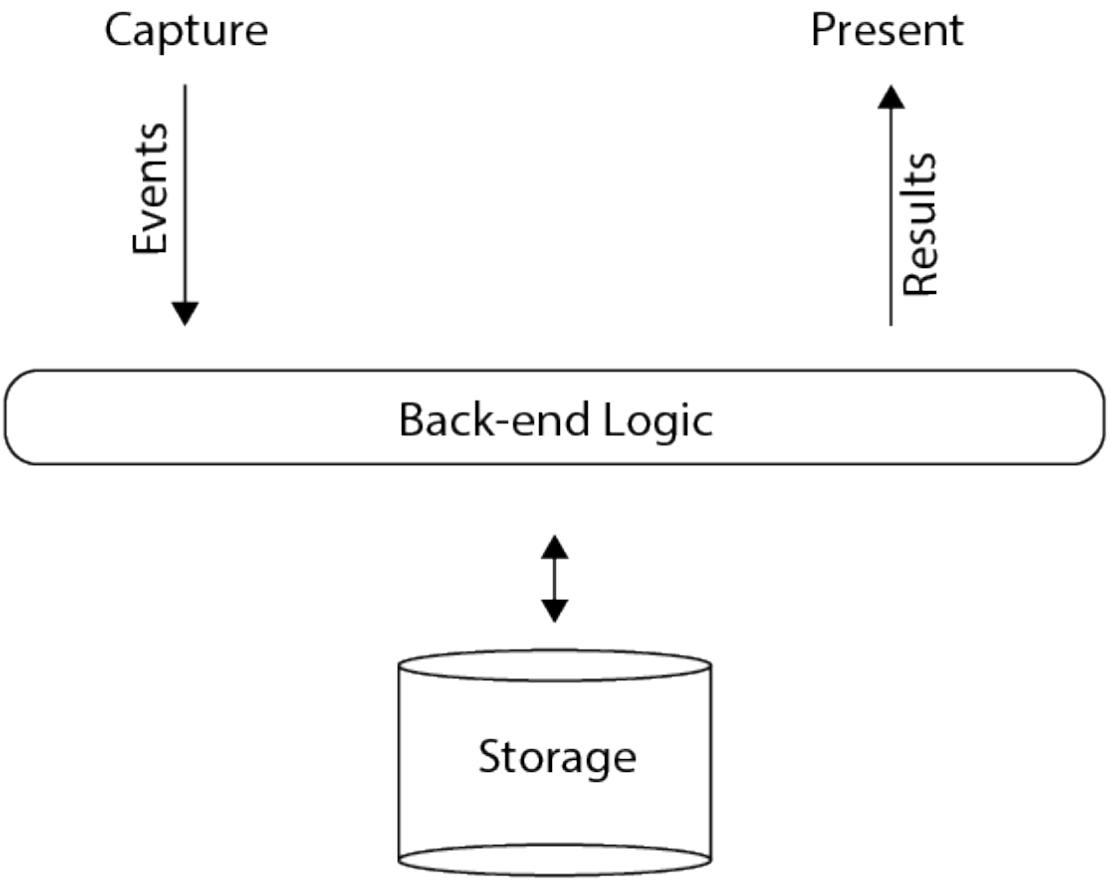


Figure 3.6: Conceptual arhitecture

Chapter 4

Soccer Analytic toolkit

4.1 System Architecture

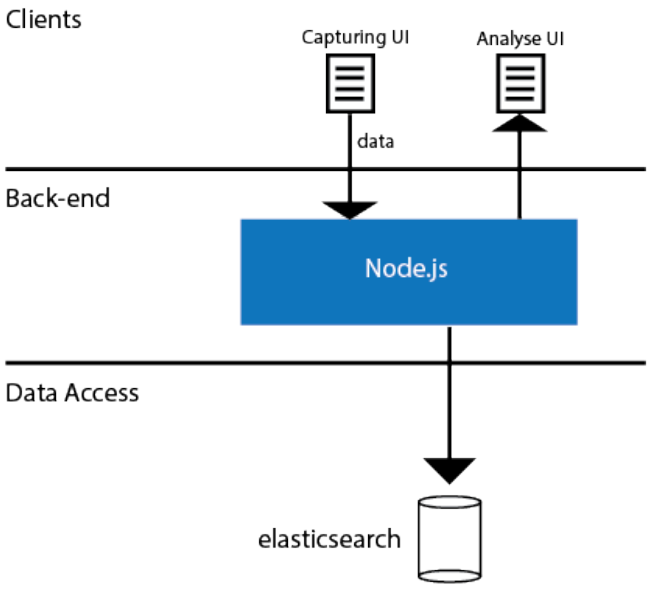


Figure 4.1: Overall architecture of the system

The system architecture can be layered in to three layers; client, back-end, database. The control flow flows from the clients requesting something or inserting data, to the back-end and to the third layer, the database, and back up again in reverse order.

4.1.0.3 Interface

The Soccer Analytic tool has one user interface - the web browser interface. Both the input and the analytic interface can be reached from this single web browser interface.

4.1.0.4 Back-end

The back-end is the middle layer between the client and the storage. Its main task is to serve static files to the clients, handle data insertions or handling web request by mapping them to database operations. New data insertions will possible be inserted into several database indexes. The back-end will then ensure that all indexes are updated before returning success to the client.

4.1.0.5 Storage

The storage layers task is to persist data and handle search queries on the data. It consist of several indexes that each store a part of domain model.

4.1.1 Domain model

The domain model is based around matches. All attacks is wrapped into a match root. Attacks can be seen as subdocuments of the match document. Inside the attacks all passes lies with other information. This gives a easy way of understanding how all the data is related together. Below is a complete example of how it all is structured. The number of attacks and passes is stripped down to one in the example.

```
{
    "hometeam" : "Troms",
    "awayteam" : "Rosenborg",
    "score" : "1-0",
    "date" : '2013-15-09',
    "attacks": [
        {
            "time": 4,
            "touch" : 1,
```

```
            "team" : "Troms",
            "breakthrough" : "None",
            "breakthroughPlayer" : "None",
            "typeOfAttack" : "D dball",
            "attackStart" : {
                "pos" : 17,
                "typeAction" : "Frispark",
                "player" : 403,
            },
            "passes": [
                {
                    "fromPlayer": 403,
                    "toPlayer": 393,
                    "fromPos": 17,
                    "toPos": 23,
                    "action": "CROSS"
                }
            ],
            "finish" : {
                "player": 393,
                "pos": 23,
                "action": "SHOTMISS"
            }
        },
}
```

4.2 Implementation details

Software Stack	
Web server	Node.js
Database	Elasticsearch
Web Interface	HTML5, CSS3
Client Side Technology	Backbone, Bootstrap, jQuery, Mustache, Highcharts, Lightbox, async

Figure 4.2: Software stack

4.2.1 Storage

The data storage is an elasticsearch database[**?**]. Elasticsearch is document oriented, schemaless and works well with JSON[1]. As our server is built on JavaScript working with JSON is easy. JSON-objects can be inserted right into the storage and elasticsearch will map fields and value accordingly, and make it available for search. Elasticsearch takes advantages of embedded documents meaning we can store related data together. As an attack is usually made up of several passes. With elasticsearch you can store the passes as an embedded document embedded in the attack document so they can be retrieved in one query.

The main reason for using Elasticsearch is its search capability. In the starting phase of the project MongoDB [2]was the storage engine. However as some aggregation queries was hard to figure out how to do a change to elasticsearch was made. It should be noted that MongoDB supports map reduce operations. With some extra work the queries causing problems could possible have been done with MongoDB. However, with elasticsearch you can in a single, simple to write, query get counted how many passes all players for a team has played and received, the number of times all players has been the breakthrough-player in an attack, count type of attacks, count the most used zones for passing and finishing and so on. This makes it very easy and efficient to do queries for analyses on teams and players.

4.2.1.1 Indexes

Indexes in much like tables in a relational database in the way that they is a container for data. An index stores documents which is a bunch of key/value pairs like JSON. You can let elasticsearch automatic analyze the field data or you can use mappings. For supporting querying on players and other fields where the value is more than one word, you have to tell elasticsearch that it is a multi field. For example searching for player name "Stefan Johansen" when the field is not set as multi field will give you two results and not one as you would expect.

In our project we have 5 different indexes. Team index which stores all the teams. Player index which stores all the players. Match index stores all data from a match including all attacks. Attack index stores information

[2]http://www.json.org/
[2]http://www.mongodb.org/

about attacks and last the pass index which stores passes. As may be noted there is data redundancy. You can argue that storage has become so cheap and if you can use a little bit extra space to gain performance you would do it. In this case it was done to be able to support aggregation functions on subdocuments.

4.2.2 Back-end

The back end is the middle-ware between the clients and the data layer. It exposes a RESTful interface over HTTP for the client to communicate. A request coming in is transformed to a database query based on the resource it tries to access. On answer from the database the result is transformed before returning it to the client.

Similar if the client sends new data for a match the middle-ware inserts the data into the appropriate indexes. The server will respond will HTTP status code 201 if all goes well or 400 on an error. The server uses HTTP code actively to tell the client the result of requests.

In principal, since the data input is generated in the web browser (with JavaScript), it could have been inserted right away into the database without going through an extra middleware. However, this limits us as we cant combine multiple queries by going through the back-end. Cleaning of data before serving to the client would neither be possible. Normally you also do validation of the data at the back-end before inserting into your database.

4.2.2.1 API

4.2.3 Front-end

Front end is consist of a single page JavaScript application using Backbone.js [3]as under-supporting framework. Backbone uses a MVC model to structure the code. With Backbone your views will update automatic when data changes. In the following sections the architecture of the client and how different concepts is used will be described.

[3]http://backbonejs.org/

Web server API	
GET /matches	Get all matches
GET /match/:id	Get specific match
POST /match	Post new match
GET /teams	Get all teams
GET /team/:name	Get team statistics
GET /team/:name/finalthird	Get passes into final third of the pitch
POST /team	Post new team
GET /player/:id	Get player statistics
GET /player/:id/breakthroughs	Get breakthroughs for player
GET /player/:id/finalthird	Get passes into final third of the pitch
POST /player	Post new player
POST /attack	Post new attack
POST /pass	Post new pass

Figure 4.3: Overview of the web servers API

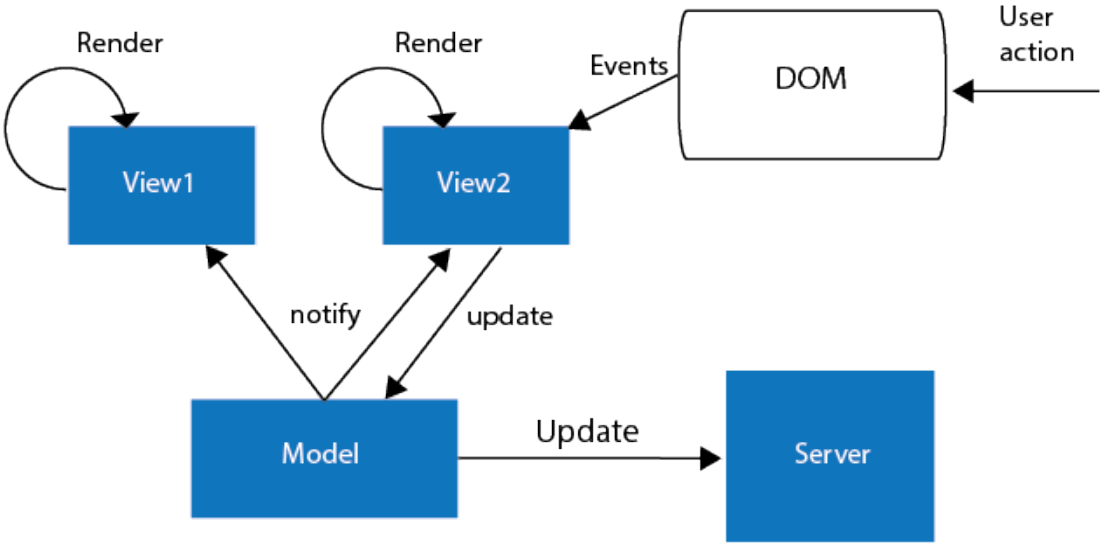


Figure 4.4: Architecture of the client side

4.2.3.1 Models

Data is represented as models in backbone. The model has mainly to responsibilities. First is whenever a update on a models data occurs the model notifies the views that has subscribed for update events for that particular

model. The second is that models is responsible for AJAX communication with the back-end. An example is when a user creates registrates a new attack for a match. He fills out a form and press submit. Then a new Attack Model is created. Calling save on the instance of the model will send an AJAX post request to the server with the models data in the HTTP body.

Similar to a Attack Model we have Player Model that handles everything around players. When you click yourself into a players profile the model will fetch statistics from the back-end, notify the view that the data is ready, and the view will be rendered.

There is also a Match Model (fetching and registrating matches), Team Model (viewing statistics), and a Pass Model (saving passes). They work it the same way as the models described above.

4.2.3.2 Views

For a analytic toolkit to be useful a good UI is critical. Here several helper library is used to present the data. Highcharts.js[4]is JavaScript library for illustrating graphs. A query on team generates a lot of statistics and rather than listing them up they are presented using charts. This also gives us the advantage of displaying several numbers for each player and plot it in the same graph. In the image below we show the number of times a player has been involved in all attacks, number of passes into the final third of the pitch and the number of times a player has been the breakthrough-player.

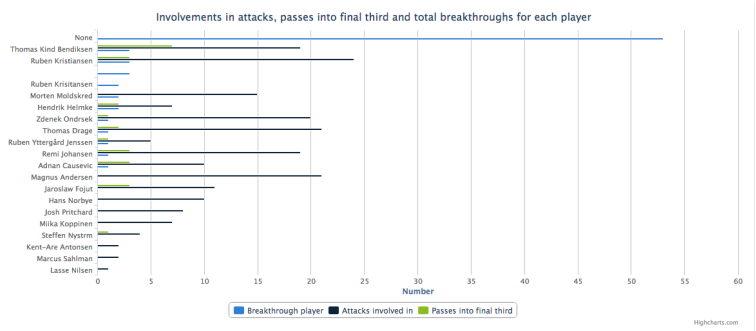


Figure 4.5: Shows how the passing statistic is illustrated on the client by using Highcharts.js

[4]http://www.highcharts.com/

Positional data is created using the a new feature of HTML5, canvas element. It lets you draw graphics on the fly in the web page. In this system it is used to create a element that symbols the different zones in our domain model. In the Team Model you have all zones with a number that symbols shots taken from that zone. This is then plotted into the respective zones.

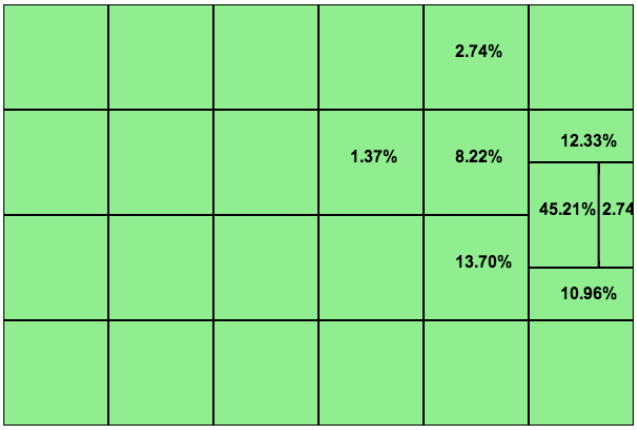


Figure 4.6: Illustrations of which zones the team has finished off their attacks from, with percent (team: Troms IL)

Backbone comes with a library Underscore.js[5]that makes creating HTML pages with dynamic content easily. When you are rendering new content on the site you can insert data retried from a model dynamically into the HTML. In the example below the input is a an array of objects where each element in the array contains a key value pair 'name' : value. The html output of this will be a list of names.

```
{{#players}}
        <li>{{name}}</li>
{{/players}}
```

4.2.3.3 Router

A component not mentioned before that lies on the client is the router. The router is the glue that binds all the other moduels together. It handles the navigation between pages in the application. When users navigate on the page by clicking on links or buttons in a traditonal web page the back-end serves a new html page. In this page this click event is handled by a

[5]http://underscorejs.org/

router module. The router model examins the URL request and routes the request to the mapped up view. The view is then responsible for calling fetch functions on the model and render the HTML.

ADD A SECOND ILLUSTRATION ADDNING THE ROUTER.

4.2.4 Getting players and teams into the database

Getting squads in a useful format like XML or JSON was harder than expected. On the norwegian football alliance's website you could download the squads for the current season only in PDF. Current squads is fetched from altomfotball.no, a website by the norwegian TV channel TV2. This is own python script meant to run only once to set up the database. For each team the script basically reads the HTML document with all players listed, parse out players name, and sends in to the web server.

MORE ON DEVELOPMENT PROCESS? How zone map changed etc. A change from Mongodb to elasticsearch

4.2.5 Security

Secturity is not taken into concern. This means anyone getting into the page can post new match data and add attacks. This could have been fixed by requiring a login before getting access to the site.

Chapter 5

Demo

5.1 Interfaces

5.1.1 Mockup

5.1.2

The first page you are prompted with is the listing of all matches registered in the database. A click on match gives you details about that match and prompts you a interface for capturing new attacks if requested. Every field has to be submitted with an correct input value.

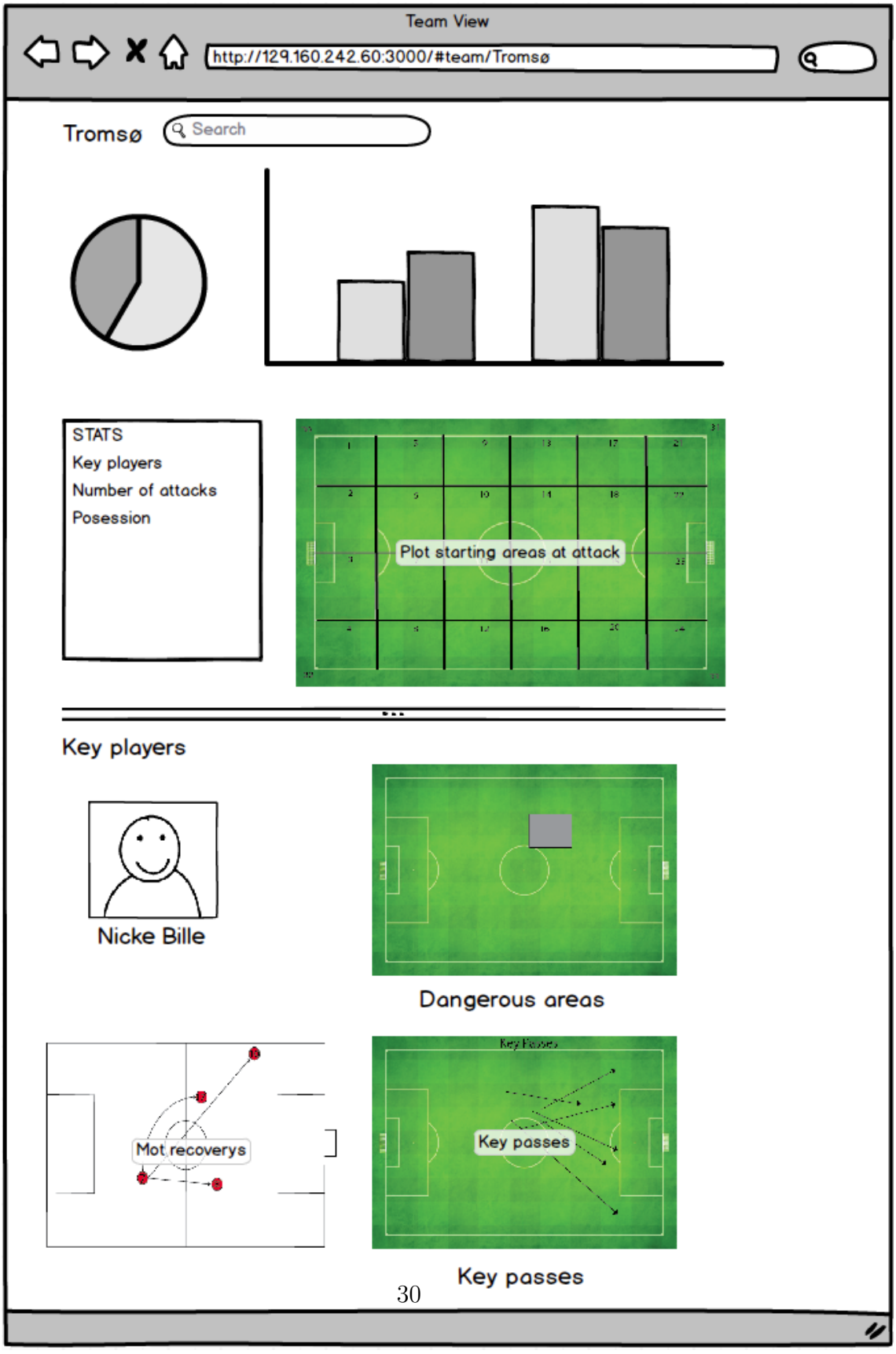


Figure 5.1: All matches registrated in the database are listed on this page

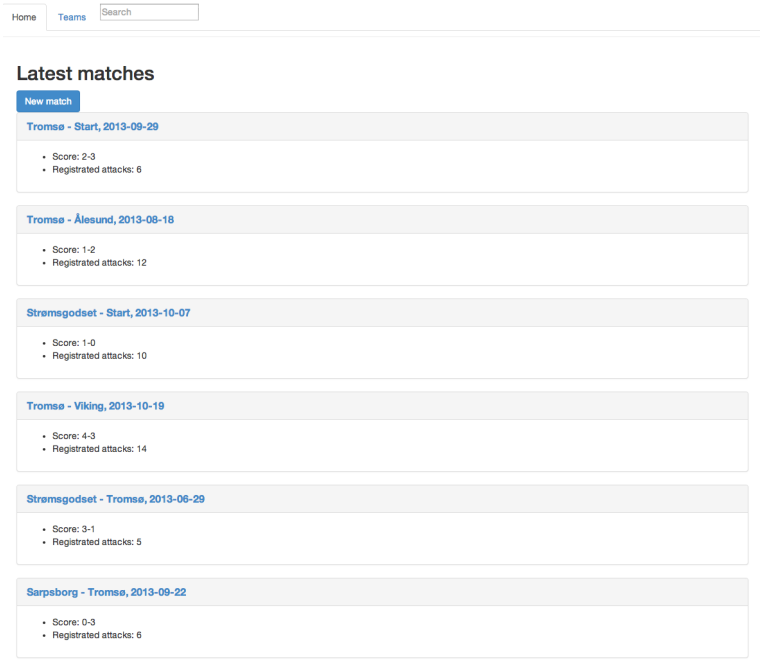


Figure 5.2: All matches registrated in the database are listed on this page

Chapter 6

Evaluation and Results

6.1 Limitations

As the input is manual the current biggest limitations is humans.

The input is to some degree subjective for some data like identifying breakthroughs.

Chapter 7

Conclusion

This chapter presents our achievements, gives some concluding remarks and outlines possible future work.

7.1 Achievements

7.1.1 Concluding Remarks

7.2 Future Work

References

[1] D. E. Comer, David Gries, Michael C. Mulder, Allen Tucker, A. Joe Turner, and Paul R. Young. Computing as a discipline. *Commun. ACM*, 32(1):9–23, January 1989.

[2] MARTIN HARDY.

[3] Collins; Barry McNeill; Marco Cardinale Valter, Di Salvo; Adam. Validation of prozone : A new video-based performance analysis system.

[4] Mark Venables. Sportstech: Football, 2013. [Online; accessed 12-November-2013].