CScore – A Real Time Application for E-sports

PROVIDING LIVE UPDATES OF PROFESSIONAL MATCHES

BJÖRN ANNERGREN

DANIEL ULUG
Abstract

The e-sport market is a growing one and as it grows monetary interests, willingness to exploit it grows with it. A part of the e-sport market that is not satisfied yet is the market of mobile applications catering to e-sport audiences. To tackle the problem this thesis explores the problems one might encounter whilst developing a soft real-time mobile application for an e-sport game. The applications name is CScore, an application that will present the score of ongoing e-sports matches. Methods used for data analysis of necessary information includes analytical induction and coding. As system development model a prototyping model was used with Scrum as its framework.

This thesis results in the presentation of a prototype of CScore, with enough information about the application to develop the complete system. An overview of the planned CScore system is presented including, how to retrieve information about matches and current information of ongoing matches. The relationship model of the local database and the layout of the mobile application is also presented.

Keywords

E-sports, Android, Real-time, Mobile application
Abstract


Projektet resulterar i att en prototyp av CScore presenteras, med tillräcklig information om hur denna applikation borde implementeras för att utveckla det fullvärda systemet. En översikt över CScore systemet är presenterat inklusive hur information om matcher skall hämtas samt hur information om pågående matcher skall hämtas. Relationsmodellen över databasen och layouten för mobilapplikationen presenteras också.

Nyckelord

E-sports, Android, Real-time, Mobilapplikation
Table of Contents

1. Introduction ................................................................................................................. 1
   1.1 Background .............................................................................................................. 1
   1.2 Problem statement .................................................................................................. 2
   1.3 Purpose ................................................................................................................... 3
   1.4 Goal ......................................................................................................................... 3
   1.5 Benefits for society, ethics and sustainability ............................................................ 4
   1.6 Method ..................................................................................................................... 4
   1.7 Delimitations .......................................................................................................... 5
   1.8 Disposition .............................................................................................................. 5

2. Real-Time Mobile Applications ...................................................................................... 7
   2.1 Real-time systems .................................................................................................... 7
   2.2 Real-time applications ............................................................................................ 9
   2.3 Mobile applications ............................................................................................... 9
   2.4 Mobile platforms Android and IOS .......................................................................... 10
   2.5 Server ..................................................................................................................... 11

3. E-sports ......................................................................................................................... 13
   3.1 E-sports in general .................................................................................................. 13
   3.2 Counter-strike: Global Offensive ............................................................................ 14
   3.3 Professional teams and players .............................................................................. 15
   3.4 Competitive matches and betting .......................................................................... 15
   3.5 Live score applications ......................................................................................... 16
   3.5.1 Live score applications for traditional sports ..................................................... 16
   3.5.2 Live score applications for e-sports ................................................................. 17

4. Methods ......................................................................................................................... 18
   4.1 Data collection ........................................................................................................ 18
   4.2 Data analysis methods ............................................................................................ 19
   4.3 Methods used in this thesis ................................................................................... 19
   4.4 Texts ....................................................................................................................... 20
   4.5 Interviews ............................................................................................................... 20

5. System Development Models and Development Environment ....................................... 22
   5.1 Waterfall model ..................................................................................................... 22
   5.2 Prototyping ............................................................................................................. 22
1. Introduction

E-sports (electronic sports) is a growing concept and will only get bigger in the future [68]. Many betting sites, sponsors and professional teams are involved in the scene [68]. Because of the many sponsors involved there are professional matches scheduled by tournament organizers almost every day of the week [37].

The e-sports scene is very similar to traditional sports such as football, basketball or tennis. These traditional sports are attracting billions [5] of viewers and have many applications, programs and websites catering to their audiences. Since e-sports have a much smaller scene, comparing it to traditional sports is an easy way to see what aspects are lacking and what can be done to make the e-sports scene more complete.

One of the less fleshed out parts of the e-sports scene are mobile applications. Some services are non-existent and some are underdeveloped. There exists different applications but in general, existing applications for e-sports are about information retrieval. The information a viewer demands can be a schedule of upcoming matches, latest news of teams and players, statistics of old matches and so on. One of the underdeveloped services is real-time information retrieval applications.

Real-time systems deliver a response before the end of a deadline. The deadlines are time constraints in which the system must create a response to the outside event that triggered the deadline [46 page 11].

This thesis will be focused on real-time live score applications for e-sports. This translates to an application that shows the current score of live matches, and results of matches. The name of the application will be CScore.

1.1 Background

The term e-sports includes every competitive multiplayer video game with a big enough fanbase to support a competitive scene [52] and these video games can be played on any platform. Currently the biggest gaming platform in e-sports is PC¹ and is mainly represented by the games Dota 2, League of Legends and Counter-Strike: Global Offensive [1]. The big tournaments in these games usually have more than millions of viewers and they attract more people every year [1].

Since it is a growing market there are services that are common for traditional sports, but not fully explored for e-sports. Such a feature is an application that shows score in real-time for live matches. This kind of application does already exist for traditional sports and there are some in e-sports too [32]. The main goal of the application is to give information about ongoing professional matches to the user. The information contains the score of the match but it can contain other features such as player statistics or information about teams.

¹ Personal Computer
A common term for this kind of applications is “live score application”. Their purpose is to make it easier for fans and viewers to check up the score of matches, to follow up on their teams, and see how their bets are faring.

The reason there are so few of this type of application is because it is hard to implement correct real-time. The constraints of the deadlines needs to be considered during the development. Real-time systems deliver a response within the confines of a deadline [46 page 11]. Depending on what happens with a response if it misses its deadline, a classification has been agreed upon for real-time systems. A real-time system can be either hard, firm or soft depending on the management of the response missing the deadline [46 page 19]. A real-time system of the type hard is a system where deadlines are system critical and missing one deadline means critical system failure [46 page 19]. If the deadline is not met in a firm real-time system, it does not lead to a critical system failure, although the response that misses the deadline loses its usefulness [46 page 20]. In a real-time system with soft deadlines the responses usefulness degrades after missing the deadline, it does not instantly lose it [46 page 21].

It is expected that CScore will receive constantly changing data from the external source and rapidly process and utilize the incoming information [19]. This is not an easy task and there are many ways to solve this problem. However, working with soft real-time [20] might be sufficient enough in this kind of application since the main goal is to make the user have the feeling of real-time.

Many companies are noticing the many possibilities for application development in the growing e-sports scene. One of these companies and the customer of this project is Iajans [39]. It is an international company founded in Turkey. The company has its main focus on ecommerce, web/application development, web hosting and online marketing [76] and has eight ongoing major projects in total. Small projects are scattered throughout Europe, mainly done by inexperienced students while bigger projects takes place in Istanbul. Their biggest ongoing project is an online shopping site for jewellery and accessories [40].

This project is one of Iajans smaller projects. The project is to develop a real-time Android2 application that can retrieve live scores of professional e-sport matches. Iajans will have a project manager giving out requirements for the application. The project manager will also have to approve every design, function and detail.

1.2 Problem statement

As mentioned earlier, e-sports is a new market that can turn out to be a good investment. Companies, small or big, wants to enter the market to establish a good spot for future benefits. Adding the interest of companies with e-sports’ lack of mobile application services may cause an explosion of new applications. The quality of these mobile applications can become a problem because companies might try to take advantage of e-sports growing as of now non-competitive market. Another problem might be the oversaturation of a niche of the market, because many companies may develop the same sort of mobile application and release it, competing for the same users.

2 Biggest operating system for mobile platforms [41].
The games included in e-sports are developed in many different ways, creating an unstandardized environment. For example the servers used for one game might have a public application programming interface whilst another game’s servers might require sign up for access. The problem is that interested parties wants to encompass as many of these games as possible when developing a system or an application for e-sports. In this case, locating resources to examine how each game is implemented, and adapt the system or application accordingly, will be a time consuming necessity. This also makes it impossible to create a completely general real-time system for e-sports information retrieval, instead it must be adapted for each case, or game.

Making a real-time system creates many constraints on what can be done in it, both from a hardware and software perspective. The hardware must be fast enough to handle the responses in a viable time. The servers in a distributed system must also be highly secured, minimizing the risk of down time, else they risk creating many missed deadlines. From the software perspective the functions relevant must execute fast enough, and each instance should execute for approximately the same duration, to be viable as real-time functions. If these considerations, hardware or software, are not taken into account the application might be crippled and users might be dissatisfied.

Iajans is one of the many companies that wants to be a part of the e-sports scene by developing a mobile application. The outcome of the project should be useful for e-sports and at the same time fulfil Iajans’ requirements. This might prove to be difficult because of the constraints of real-time and e-sports. The project will try to solve the problem:

How to develop a mobile application that displays, in real-time, scores of professional e-sports matches that meets the requirements of the customer?

1.3 Purpose
The purpose of the thesis is to present the development of a real-time application, following requirements from a customer. More importantly, how to design and develop a mobile application for Android and to present the procedure and the end result.

The purpose of the application is to give the possibility for e-sport fans, viewers and bettors to easily see live scores, scheduled matches and results of played matches for the day on a mobile device.

The purpose for Iajans is to get an understanding for the e-sports application market and get involved in the e-sports community.

1.4 Goal
The goal of the thesis is to provide a development solution for real-time applications for a new market. In this case the market is e-sports.

The goal for the application is to provide simple and fast information about currently ongoing or upcoming e-sports matches.
The goal for Iajans is to create a mobile real-time live score application for e-sports.

1.5 Benefits for society, ethics and sustainability

E-sport fans and bettors that are not able to watch the game via a stream, or other methods, will benefit from this project the most because they will be able to follow up on the matches they are interested in. The application can inspire other developers to do similar projects or give them ideas to new better ways to approach some of the problems that exist in e-sports.

A downside is that the live matches and streams might get less viewers because of the applications easy way to follow scores. The drop in viewers can be problematic since the amount of viewers a stream of an event has is one of the ways to measure the popularity of an event. The lower viewer count may in turn discourage new sponsors and companies to invest in e-sports or similar events.

From an ethical point of view this application could make betting more attractive and in consequence cause gambling addictions. Some people only bet on the matches they can follow. Giving the opportunity to follow the matches anytime, anywhere might influence some to bet more and heighten the risk of becoming addicted to betting. However, the application will not create the gambling problems directly and hopefully the effects will be minimal.

Other ethical issues were considered during the interviews with the project manager. It was important to refrain from using leading questions during the interviews. Responses in disagreements needed to be neutral and free of emotions in order to create a healthy and agreeable solution. If not for these precautions, the direction could easily be altered during the interviews and one of the parties could end up with a decision they did not agree with.

Economical sustainability can become a difficulty when there are too many daily matches or too many users, which would push the server to its limits. This would in turn force the application to use a more expensive server. The application will be available for free but advertisement in the application could be a source of income in case these expenses threatens the applications sustainability.

There is not any directly inferred environmental impact.

1.6 Method

Research methods can be divided into two broad categories, quantitative research methods and qualitative research methods [38]. Quantitative research relies on hypotheses or theories that can be proven false or true via experiments [38]. For this to be true the hypothesis or problem of a thesis must be able to be reduced to variables that can be determined experimentally. Statistical analysis is a requirement for a quantitative research method. Qualitative research methods relies on hypotheses or problems that may not be experimentally or statistically proven true or false for all instances [38].

In this thesis a qualitative research method will be used because of the nature of the problem statement: “How to develop a mobile application that displays, in real-time, scores of professional e-sports matches that meets the requirements of the customer?” This statement cannot be answered by doing statistical evaluation, or by using other general quantitative research instruments. The problem can thus not be adapted to fulfil a quantitative research method’s requirements for a problem.
Specifically an applied research method will be used. This means that a practical solution will be created by applying accepted research. Accepted research will be applied to resolve problems that arise, research that have been proven to work as intended, and no completely new unproven theories will be used.

The data collection will contain texts and interviews. The source types of the texts will be academic books, scientific articles and some journalistic articles. The texts will be used for a theoretic understanding of necessary components of the project and interviews will be conducted with the customer to find out their requirements for the application and their expectations.

1.7 Delimitations
The product’s requirements will be set and confirmed by the project manager assigned by Iajans. The manager has final say on everything and this includes design and functions of the application.

It is not possible to develop an application for all mobile operating systems at the same time, without some overhead. Each operating system have its own application development considerations that must be taken into account. Therefore, a specific operating system must be chosen.

E-sports are highly unstandardized in how to retrieve information from matches. Therefore, CScore will be focused on one e-sport game. Adapting it to multiple e-sports will require resources allocated to adaptation and specialization beyond the limits of this project.

Even with the focus on only one game, there must be other limitations to the project. The updating of scores from live matches will not be in hard real-time, since the way scores are achieved are not time-sensitive enough to warrant the extra development time, especially in e-sports. The updating must only appear to be in real-time for the user, in order to create a satisfactory experience.

1.8 Disposition
Chapter two explains real-time system, mobile applications and servers theoretically. Chapter three explores the current state of the e-sports scene, live score applications and what the term e-sports entails with focus on the game Counter-Strike: Global Offensive. Chapter four discusses the data collection and data analysis methods that exist and what methods were chosen for this thesis. The results of the data analysis is also discussed. It also explains how the interviews with the customer were conducted. Chapter five presents an overview of different system development models, development environments and an introduction to design science. The chapter also mentions the system development model chosen and why, and which development environment was chosen and why. Chapter six describes the development of the first CScore prototype. Chapter seven presents the first CScore prototype. Chapter eight contains a discussion of the thesis and if it fulfilled everything it set out to fulfil.
2. Real-Time Mobile Applications

Everyone that owns a smartphone or tablet uses at least a couple of mobile applications. Most of have even used a mobile application with a real-time system without even noticing. In order to begin developing a real-time mobile application, it is necessary to fully understand what it really means.

2.1 Real-time systems

Real-time systems are systems that deliver a response within the confines of a deadline. There exists two ways of measuring time, real-time and logical time. Real-time demands the use of a physical clock to measure time, [46 page 2] thus earning its name “real-time” since it requires measuring the flow of time in the real, physical world. Logical time is instead the ordering of different events in chronological order without the use of a physical clock, for example if this button is pushed this happens afterwards. A real-time system cannot be described fully without the aid of a physical clock. If it could it would not be a real-time system [46 page 3].

Real-time systems defining characteristics are deadlines and the real-time systems slightly different correctness criterions than regular systems. The deadlines are strict time constraints in which the system must guarantee a response to the event that triggered the deadline [46 page 11]. Correctness in real-time systems are not only based on logical correctness of responses but also on the time the response is produced. A response produced after a deadline will be considered incorrect no matter the logical correctness in most real-time systems [46 page 11], soft real-time systems being an exception where the response’s usefulness is merely seen as degraded.

Depending on the importance of the deadlines for a real-time system, a classification has been agreed upon. A real-time system’s deadlines can be either hard, firm or soft [46 page 19]. A real-time system can of course have a mix of different kinds of deadlines.

Hard is a system where deadlines are system critical and missing one deadline means critical system failure [46 page 19]. Hard deadlines are often deployed in embedded systems that have time-sensitive activities that might have fatal consequences if a deadline is missed. An example is the engine control systems of vehicles, where a late response may cause complete engine failure or worse. Another is pacemakers where a missed deadline might mean a dead user. A hard real-time system must thus strive to meet all deadlines.

A firm real-time system’s deadlines are important but if the deadline is not met it does not lead to a critical system failure, although the response that misses the deadline loses its usefulness [46 page 20]. This system must also strive to meet all of its deadlines since otherwise the response is completely useless and can be disregarded.

In a real-time system with soft deadlines the response’s usefulness merely degrades after missing the deadline, it does not instantly lose it [46 page 21]. The goal of a soft real-time is thus to meet as many deadlines as possible, and if a deadline is missed to minimize the time taken for the response to arrive to lessen the degradation of the response.
There are two different kinds of events that can trigger a response, or task, and a deadline in real-time systems: Stimulus events and Response events [46 page 23]. Stimulus events are triggered by the outside stimulus or the environment and creates an action on the system itself. Response events are triggered by a Stimulus event and creates an action on the environment. For example a stimulus event might be the periodic measurement of temperature on a car engine, and a response event might be if the system notices via this measurement that the temperature is too high thus making the system try to cool down the car engine.

Figure 1. Shows how a real-time system and environment interacts via two kinds of events.

Real-time tasks, or responses, can be divided into three types depending on their occurrences over time: Periodic tasks, sporadic tasks and aperiodic tasks [46 page 38-40]. Periodic tasks are triggered in intervals of the same length of time; Sporadic tasks can be triggered at random times and aperiodic tasks are triggered randomly, like sporadic tasks, but multiple instances of the same task can be triggered at the same time.

An important part of real-time systems are how to schedule these tasks, optimally so all of them meet their deadlines. Depending on the type of deadlines these task have, the scheduler might be given some leeway in how to schedule them. The task schedulers can be divided into three broad categories: Clock Driven, Event Driven or Hybrid [46 page 42].

Clock driven schedulers creates the schedule before the system runs, or goes online. An example of a Clock Driven scheduler is the Table Driven one. This type of scheduler is bad at handling sporadic and aperiodic tasks because of their random nature.

Event Driven schedulers schedules depending on a task’s deadline when the event that triggered the task happens. Two well-known Event driven schedulers are Earliest Deadline First (EDF) and Rate Monotonic Analysis (RMA) [46]. EDF is when the task with the shortest deadline gets to run first avoiding an inherent priority queue in the scheduler. RMA gives each task a static priority. The priority is based on the duration of the task. This makes it possible to guarantee a certain response time.
2.2 Real-time applications

Many different applications exist that uses real-time systems and they are called real-time applications. These applications usually requires input from an external environment, perhaps a sports game or the temperature of an engine. These inputs are mostly time-sensitive requiring a response from the application before a deadline has ended.

An example of a real-time application is the streaming of a live feed from another person’s computer [46 page 7]. In this case the external input is the streamers own monitor screen. Everything that is shown on this screen are converted to a video file, compressed, and continuously broadcast over the internet to viewers [46 page 7]. This is a time-sensitive operation meaning that all frames that misses their deadline of reaching the viewer’s screen will be dropped. This will manifest for the viewer as an unstable framerate and increases the viewer’s dissatisfaction level.

Another example is multiplayer competitive games. A real-time sensitive aspect of these are the inputs of users to their player avatars on the game server. The external input in this case is the user’s button presses. The inputs must propagate as fast as possible to the game server or else the user’s instance of the game world will be desynchronised from the server’s instance of the game world which will manifest as lag once the server tries to synchronize them.

2.3 Mobile applications

Mobile applications are applications that are suited to run on mobile devices, mainly on smartphones and tablets. There are millions of applications [62] and more gets developed every day. They are usually focused on fulfilling a specific task.

Many applications are adaptations of websites. Their purpose is to improve their usability on mobile platforms. Their equivalents exist as websites too, but users prefer the applications. Facebook [27] and Twitter [35] are some examples that initially started as websites. On the other hand, there are applications that do not have website adaptations. Some popular examples are the application Snapchat [30] where users can send pictures and text messages that disappears after a short period of time, the dating application Tinder [34] that allows the user to find and match others in a certain area with help of GPS\(^3\) and Tango [31] that gives mobile users the possibility to video chat and voice chat over the internet.

More than one billion people has downloaded the Facebook application on their Android devices [27]. The amount of mobile application usage is rising exponentially every year and the reason is mainly because of accessibility. Both smartphones and their network usage are getting cheaper and faster [70]. People do not need to sit by a computer to chat, read news or watch videos anymore. Radio, newspaper, television are getting their places in the smartphone in form of applications.

Other appealing benefits exist for mobile device. Sharing location is one of them. If the users allow, mobile applications can get the information about the devices’ location. This gives the

---

\(^3\) Global Positioning System
possibility to find places and events near you, socialize with others in the neighborhood or simply share location for people to see.

Another beneficial function for mobile devices is alerts and notifications. Smartphones are becoming almost inseparable with their users nowadays and mobile applications use this to interact with the user.

2.4 Mobile platforms Android and iOS

There are two operating systems dominating the current market for mobile devices, Android and iOS. Together they account for 96.7% of the market [41], although Android accounts for the biggest part at 82.8%. These two operating systems share many similarities and the differences are more in slight variation and policy. iOS policy can be summed up as ease of use, even if it inhibits users’ control of the device whilst Android allows more user control, even if it increases the device’s complexity.

Android and iOS is primarily used on mobile devices, such as smartphones and tablets, although they see usage in all kinds of devices [61] [3]. Everything from cars to tv-screens uses these operating systems with slight adaptations to the specific device. IOS is mostly restricted to Apple devices [3], Apple being IOS’s maintainer. Android is based on the open-source Linux kernel [61], is currently maintained by the company Google [25] and is licensed to many brands of devices.4

Android users download applications to their phones via Google Play and there are 1.6 million applications available [62]. On the other hand, 1.5 million applications are available in the Apple App Store for IOS users [62]. This shows how evenly invested both operating systems are.

However, this was not always the case. Figure 2 shows that Apple was dominating the market in 2010 but Android managed to pass Apple in 2015. The growth of Apple is steady while Android is growing exponentially. It is unknown if Android will keep on growing faster than Apple or if they will both grow steadily in the future.

4 Some phone companies using Android is Acer, ASUS, HTC, Huawei, LG, Kyocera, Motorola, Samsung, Sony, and Zte [2].
There are different amounts of applications because it is complicated to develop an application for both iOS and Android at the same time. Cross platform tools such as PhoneGap [54] and Xamarin [74] makes it possible to develop a mobile application for multiple platforms. But these tools are difficult to use, especially for amateur developers. Most applications in the market does exist for both platforms but small companies and projects does not have enough resources to do so. The same problem exists for this project too. In order to reach out to as many viewers and fans as possible, the choice was to work with Android as the main platform. Its biggest competitor IOS was not chosen because of its smaller market share.

2.5 Server

An application can be divided in two separate parts; front end and back end. Front end is the part a user can access and see. It is the presentation of the back end, where all the data and functions such as real-time systems are stored [58].

Usually, back ends of websites or applications get handled by a machine called server. It handles it by taking requests from clients and respond to them following the client-server model [50].
Usually the clients are users that connect to the application via a network, although the clients may be on the same computer too.

Every website and application has a collection of data that it uses. This data can be account numbers, passwords, pictures and comments in form of numbers, text or other formats. It is necessary to store and retrieve this data with an efficient system. This system is called database and it is used by most websites and applications [59]. It is up to the developers to choose how their database is built. The database gets more complex when there are different sorts of data with different attributes.

This project will need a database server that contains the information about the professional matches being played. The database will contain information such as teams, players, start time and so on. It will be this server that retrieves the scores of the matches in soft real-time and sends them to the mobile application for the clients to see.

The server will also interface with all Application Programming Interfaces (API) used in this project. API is the method for different systems to communicate with each other over a network [43 page 5]. When allowed by the API provider, it gives the possibility to access the other systems back end information. Some back end information is public and some are private only seen by selected systems. It is up to the API providers to choose functionalities offered on their own API [43 page 7].
3. E-sports

Knowledge about e-sports, Counter-Strike: Global Offensive and live score applications is a necessity for developing an application pertaining these areas. The knowledge will give a better understanding of the needs e-sports fans have and this will help to create suitable designs and functions for the application.

3.1 E-sports in general

E-sports encompasses all videogames that has a multiplayer competitive element to it and has gathered enough of a fanbase to support professional participants. The term stands for electronic sports and was coined to differentiate it from traditional sports. Many will argue e-sports is not a real sport, since it does not demand physical exertion of the same level. The defending argument for video games is that it does require tremendous amounts of exercise, strategy, teamwork and raw skill [16].

E-sports is a relatively new concept, since it uses turing complete machines as platforms which only reached the mainstream consumers in the seventies [73]. This in stark contrast to the rich and long history of many of the traditional sports that are popular today, which might influence the opinion that e-sports does not belong as a traditional sport. However in 2013, United States Government officially accepted the popular video game League of Legends as a fully professional sport [7], but this did not end the debates.

Over ten million people attended live e-sport events in 2014 [68]. Even though this number might pale in comparison to traditional sports, the trend is an upward one. For the last decade the popularity of e-sports have been growing steadily. Just over the last few years the viewership growth of e-sports have been indicating an exponential curve [68]. The viewership for 2015 has been estimated at 134 million total [68]. This is still less than traditional sports.

The current surge of popularity have a few possible explanations. One cause that is widely accepted yet speculative, at least as a necessity for it, is the massive investment in high broadband internet by multiple states during the nineties and two thousands. This opened up for a global competitive internet-based environment from the comfort of interested parties homes.

Another cause that is more speculative is the relative recent refinement and sudden rise of real-time streaming services focused on videogames, of which the current market leader by revenue is Twitch.tv [69]. This platform created a simple, cost-effective way for professional matches to be casted live, minimizing cost whilst maximizing reach.

Out of these streaming services a new kind of e-sports celebrities have emerged, drawing in interests of corporate sponsorships to individual players and in extensions their teams. Many e-sport teams popularity can be attributed to having an e-sport celebrity on the roster, much like traditional sports.

Another big reason for the video game industry growing is the development of digital game store softwares such as Steam [65] and Origin [51]. The game stores makes it easier to buy games and
download them without a compact disc. It is possible to play single player games offline and see friends playing games in real-time. This makes it easier to join up with friends to play multiplayer games.

The biggest game store software is Steam owned by Valve. Valve develop their own games but they also distribute other games from different companies. As the Figure 3 shows, around five to ten million users are online on Steam every hour of the day [67], with overall fifty million active users in total [75]. These numbers indicate how popular the video game industry have become.

![Figure 3. Graph showing active steam users between 29th and 31st October.](image)

### 3.2 Counter-strike: Global Offensive

Counter-Strike: Global Offensive is a first person shooter game made by Valve. It is the fourth iteration of the game, with its three predecessors, in order of release, being: Counter-Strike, Counter-Strike: Condition Zero and Counter-Strike: Source [66]. All competitive matches of the four iterations have used the same format with the only difference being slight variations in gameplay. It is widely accepted as an e-sport and its popularity is rising.

There are different play modes in this game, yet only competitive mode is used in professional matches. The competitive mode has two teams of five players each. One side is called Counter-Terrorists, the other is Terrorists. The goal of the Counter-Terrorists are to stop the Terrorists from planting and detonating a bomb at one of two bomb sites on each map and the Terrorists objective is the reverse. There are three win conditions for Counter-Terrorists of which only one need to be fulfilled:

- Kill all the Terrorists
- Run the round timer out without the bomb being planted
- Defuse the bomb when it has been planted
The terrorists only have two win conditions:

- Kill all the Counter-Terrorists
- Plant the bomb and defend it until detonation

The competitive matches are best of thirty rounds with each round having a time limit of one minute and forty-five seconds. Halftime occurs after the fifteenth round and the teams switch sides. The team that gets 16 rounds first wins the match. If the score ends up even at fifteen-fifteen after thirty rounds, the match goes to overtime. Overtime is best of six rounds, and if the overtime ends up with the score three-three even, the overtime repeats itself until one team gets four round wins. A competitive match finishes in less than one hour most of the time.

### 3.3 Professional teams and players

Since the game is played five versus five, players who want to play competitively must form teams. Teams with good players are usually sponsored by companies that is looking to make their brand heard and want to get involved in the community. However, the sponsors do not have much control over the team and players, as players’ roles or roster goes. The players have the authority to kick or recruit a player from the team. This, however, causes roster changes to happen quite often.

In most cases, players from the same country or similar countries that speaks the same language is on the same team. And just like most sports, there are a huge amount of teams and players yet only the top teams in the world gets any attention and good sponsors. Most of the sponsors are related to gaming, such as monitor and computer brands or energy drinks [21].

### 3.4 Competitive matches and betting

Like traditional sports, Counter-Strike: Global Offensive requires a lot of time invested. Players play the game eight to twelve hours every day just to keep up their performances. A professional player is either playing alone to improve individual skills or with the team practicing strategies. Professional teams play against one another a couple times a week to try out their new tactics and find out flaws in their gameplay. But these games are only friendly matches that does not get recorded or taken seriously by any means.

On the other hand, competitive matches are scheduled by tournaments and they are what really matters most for the teams. Contrary to the one or two matches a week in traditional sports, a good team in Counter-Strike: Global Offensive plays three or more competitive matches every week. There are many tournaments going on simultaneously and they are similar to traditional sport leagues that have finals for the top eight or sixteen teams. The amount of teams playing and the prize pools are different for each tournament. The prize pool can go up to $500,000 [18]. A prize pools of this size motivates teams and having good results at tournaments attract better sponsors and organizations for the team.

In addition to the normal tournaments, there are four big tournaments called majors every year. These tournaments are officially sponsored by Valve which makes them the most important
tournaments in the game. The top teams of majors get their brand everywhere in the game for months and millions of players see their logos and brands every time they play.

The game has an active betting community. There is of course the traditional betting community, from traditional sports, but there is also an internal betting community based around the game’s virtual items: skins. These skins can be traded on the game’s virtual market thus creating a way to exchange these skins for real currency. There are many sites dedicated to betting with these virtual items.

### 3.5 Live score applications

Live score applications shows the current score in real-time for on-going matches. These programs can be adapted to all score based sports. An important aspect of these applications are to minimize the time it takes an user to find out relevant information about matches that interests that user. To accomplish this the successful live score applications designs are simple, straightforward and intuitive.

These applications are important for highly invested bettors. What this type of customer care about is the results of their investments (bets), thus they are interested more in the score of matches than seeing the matches live. Most applications are catered to this kind of user, since they have shown a willingness to invest their money in the sport, by betting, thus showing a stronger incentive to pay for different aspects of a live score application, or be more interested in advertisements that fulfils the user’s needs. These applications are also important for fans that are unable to watch their match of interest, but still want to know how the match is going.

![Simple blueprint of a multiple sports live score application](image)

*Figure 4: Simple blueprint of a multiple sports live score application*

As interest for a sport grows, monetary interests will start investing in different ways to monetize different aspects of that sport, one of them being the development of live score applications.

### 3.5.1 Live score applications for traditional sports

There are many live score applications for traditional sports. Their main goal is essentially the same but some of them have different functions and perks. Traditional sports have more and better applications than e-sports. This is because traditional sports have been around much longer and larger amounts of money exchanges hands every year. The money involved attracts
more companies and viewers, and this causes expansion of market and competition in every aspect of the sport, including live score applications.

The three most downloaded Android live score applications for traditional sports are “Live Scores” [29], “LivesScore” [28], “theScore” [33]. A fast peek and comparison of the applications shows that the focus of these applications are usability and simplicity. There are no fancy pictures, colors or functions. The users downloads these applications to get the score and information, and nothing more is needed to satisfy them. Therefore, reliability and correct information is a must. Users will not tolerate mistakes and slow updates, especially in such a competitive market. If they have a bad experience, they will just change to one of the alternatives.

It is also usual for betting sites to have their own live score function, especially if the site offers “In-Running-Bets”. In-Running-Bets or Live-Bets means that it is possible to bet on an ongoing match [42]. The odds shift and change depending on the statistics, information and the score of the game. Majority of the betting sites for traditional sports have this function, but unfortunately it does not exist for e-sports.

The reason for live bets not existing for e-sports is mainly because of lack of information retrieval from ongoing matches. This problem can eventually be solved with new information retrieval applications, such as the one that is discussed in this project.

### 3.5.2 Live score applications for e-sports

A big amount of live score applications for e-sports are currently being developed with no application dominating the market completely yet. The viewership is bigger than at any point before and continues to grow [68], this indicates a niche that the product of this paper can fill.

The biggest mobile live score application for e-sports is called “theScore eSports” [71]. It is available for both IOS and Android, and even has a website with the same basic functions. It is developed by a big company called theScore that has its roots in live score application for traditional sports. The application has other functions other than live scores. Some of these functions are latest news in e-sports, statistics of teams and players and match notifications as alerts.

Another competitor for the e-sport game Counter-Strike: Global Offensive is the website HLTV.org [36]. It is a highly cluttered website, with many features, one of them being live score of matches. This feature does not work well on mobile devices even though HLTV does offer a mobile version of their website. HLTV does not have a mobile application, thus missing out on the advantages of a mobile application, such as push notifications and stronger cross device support. The most important features that would separate the mobile application constructed in this project from HLTV’s mobile version of the site is the ability to generate push notifications for interesting matches and simplicity of layout, and user interface.
4. Methods

It is important to choose correct methods to fulfil one's information needs when embarking on a project. A summary of the different data collection sources and data analysis methods will help with choosing.

4.1 Data collection

Data collection methods are the methods for collecting data for a subject. This thesis is applying qualitative research, therefore the data need should be able to be fulfilled by using one or a combination of these data collection types: questionnaire, case study, observations, interviews and language and texts [38].

Questionnaires collect data by asking respondents predetermined questions. This data collection source resides in the more “structured” territory since all the questions are predetermined [23 p.2]. A questionnaire can contain either closed and open questions, or a combination of both [23 p.4-5]. A question is deemed open if the answer is completely decided by the respondent. It is closed if the answers are supplied by the researcher as a multiple choice. Many questionnaires contain only closed questions.

There are many advantages of using questionnaires [23 p.6]. They can be developed and deployed quickly to a big number of respondents. There is no immediate response pressure from respondents. If the questions are closed the analysis of the data is very straightforward. There is also no interviewer that can influence the answers from the respondents.

There are also many disadvantages of using questionnaires [23 p.8]. There are some problems with the data’s quality such as its completeness and accuracy. The questions must also be short and simple to avoid misunderstandings and keep the attention of the respondent. The wording of the questions can have a major impact on the answers, thus one must be aware of the emotional connotation to the words used. Another disadvantage is that the only way for questionnaires to retrieve information is by asking questions and since these are predetermined the researcher must anticipate what questions should be asked to extract relevant information from each respondent.

Case studies focuses on a single case or a very small sample size set in its real world environment [38]. One tries to examine and extrapolate information that can be used in other similar circumstances. They are usually applied as a smaller stepping stone to more broad studies about a subject if the case study shows promising results. This data collection source is hard to apply correctly since the data gained is hard to extrapolate to other instances.

Observations means describing events, behaviours and objects in a specific setting [38]. Observations are the primary data source used by anthropologists. It is a very good data collection method for observing human behaviour.

Interviews are sessions between one or more interviewers and one or more subjects [38]. The questions asked can be open or closed, and the whole interview can be more or less structured. An interview has a smaller risk of misunderstandings than questionnaires, since the interviewer can easily correct the subject. This also allows for more complex questions to be asked without
fear of the subject not completely understanding the question. Depending on how the interviewer conducts the interview and interprets the answers, depending on his own biases, the answers meaning can easily be interpreted in a way not intended by the subject. This means an interviewer must be careful when interviewing. Another disadvantage is the time cost and manpower required for each interview. Interviewing takes time and requires a lot more manpower than a simple questionnaire. This means interviews should only be used sparingly. Text and languages are simply the collection of texts and conversations relevant to the subject matter [38]. Every thesis contains some usage of this collection source with some rare exceptions. Thanks to the popularity and growth of the internet there is an abundance of texts easily available for most subjects. The problem is rather how to analyse the data and narrow it down to a humane amount containing only the most relevant data that are trustworthy.

4.2 Data analysis methods

It is necessary to analyse the data in an organised way to understand the data correctly and to do it time-efficiently. One or a combination of these data analysis methods should be enough to analyse the data collected for this project: coding, analytic induction, grounded theory, narrative analysis, hermeneutic, and semiotic [38]. The methods are narrowed down to these 6 because of the qualitative nature of this thesis.

Coding is the categorization of data chunks, retrieved mainly from interviews or texts and languages, by naming the category a name that interprets the data chunk [47]. This creates an easy way to sort the data making it accessible for deeper analysis.

Analytic induction is the examination of a big collection of data pertaining to the same subject [55]. By this exhaustive examination the hope is to find a hypothesis about the subject, not contested by the data. The method is iterative in nature. The analyser looks at one chunk of data, formulates a hypothesis, looks at next chunk of data and if need be adjusts the hypothesis, until the hypothesis is non-contested.

Grounded theory is iterative like analytical induction but there is instead a theory or process being modified until it is not contested by the data [38]. The first step of the grounded theory method is the coding of the data [6]. This should start immediately when data collecting starts. Theoretical memos, which is commentary about the coded data, should be created too. By sorting the memos, a process or theory that explain most of the memos should be found. Repeat this until data saturation is reached and hopefully the result is a validated theory.

Narrative analysis is used when the collected data is in the form of stories or retellings of accounts and is analysed.

Hermeneutic and semiotic are the analysis of collected data in the form of text. Hermeneutic is the interpretation of the text and semiotic the interpretation of signs and symbols.

4.3 Methods used in this thesis

To collect the data required to create the prototype defined in the problem statement a combination of interviews and texts will be used. Since the prototype is commissioned by a
customer interviews with the customer is necessary. The interviews will mostly be used to extract the requirements for the product and the expectations for the product of the customer. The reason for using interviews is the ability to be unstructured, which means being able to ask questions on the fly and ask follow up questions, and to use open questions with a lessened risk for misunderstandings. The texts are used to create the theoretic understanding necessary for starting the development of CScore.

To analyse the data collected by the interviews and the text, coding and analytical induction will be used. The coding will be used to extract the requirements for the prototype of the product from the interviews. Analytical induction is used on the texts to form a theoretic understanding.

4.4 Texts

The texts were collected from a wide array of sources and in many different forms. Most of the theoretic texts were gathered with the help of Kungliga tekniska högskolans (KTH) search tool primo, which gives access to a big range of articles and books from different databases, and all of KTH’s own publications [45]. Some texts were retrieved via Google Books [26]. The type of Texts collected were in the form of academic books, academic papers and some journalistic pieces. To analyse the texts analytical induction were used. For example, to achieve a good understanding of real-time systems the first causal relationship hypothesized was: “To create a live score application, the application will need to be a real-time system”. To prove this many texts were found about real-time system which verified this until it were sufficiently proven to be true. This procedure was done for most of the topics handled in this thesis until either all causal relationships were proven true or the relationship sufficiently modified until it could be proven true.

The analytical induction was helpful and created a good theoretical understanding of real-time systems and e-sports.

4.5 Interviews

The interviews will mostly be used to extract the requirements for the product and the expectations for the product from the customer. The amount of interviews should not be limited to a certain number. The interviews should continue until all requirements are known and when all major questions are answered. Small questions that does not require a discussion or more than one sentence long answer does not require an interview. In those cases, a simple exchange of emails may be sufficient. After getting the requirements for the project, any additional questions that might come up can be asked through email. However, it is better to plan a new interview session if a major question needs to be discussed or if something important comes up during the later phases of the development.

The interviews with the project manager were conducted verbally via Skype [60], a free chat application with voice chat functions. The voice chat functions were the ones mainly used. The interviews themselves were semi-structured, with open questions. This allowed for much discussion and unplanned follow up questions. It was not a conventional interview where one party asks all the questions and occasionally, the project manager asked questions too.
The purpose of the interview was to get requirements needed for the project. The project manager was interested in the capabilities of the project group. Since the project manager was new to the subject, there were few straightforward requirements and many requirements were discussed into shape.

To extract all the requirements there were three major interviews conducted (See Appendix A for detailed notes about the interviews). All three interviews had similar layout. The interviews was about one to two hours each. If there were questions lingering from the last interview, they got answered and discussed at the next. New questions about the requirements were asked at each interview. Many questions asked during the interview was planned before the meeting and many questions came up as the interview continued. The meetings ended with a summary of discussed topics and a date was set for the next interview. The whole interview, with every question asked and answered, was written down with pen and paper. The record of the interview felt unnecessary at the time but the value of the records presented themselves with time.

The work and requirements felt clear after the third interview. No more major topics was left to be discussed and the third interview was decided to be the last one at the end of it. The possibility of planning a new interview in the future remained, in case the project manager came up with new requirements or some major questions needed to be answered and discussed. After the third meeting, emails were exchanged in order to get answers for small unclear problems. The records written helped with remembering the small details of the interviews even after weeks of time had passed.

With help of the data analysing method Coding, four major points of requirements was gathered during the interviews:

- Real-time information about ongoing matches
- Detailed information about all matches during a calendar day
- Push notifications
- User-friendly interface
5. System Development Models and Development Environment

There are different ways to work on a project. It is important to know the different work methods and models in order to find the most suitable one.

5.1 Waterfall model

The waterfall model is a sequential approach to software engineering [56]. The unmodified sequence contains these steps in order: requirements, design, implementation, verification, maintenance. Requirements is the collection of system and software requirements that results in a product requirements document. Design is how the finalization of a system architecture that fulfils all of the requirements of the previously mentioned product requirements document. Implementation is the building of the product itself. Verification is the testing of the product to make sure all of the requirements are fulfilled. The last step, maintenance, is the installation, migrations and so on of the completed product. Each step must be completed before proceeding to the next thus creating an easy plan to follow.

The model is not made for sudden changes in the requirements section mid-project making it unsuitable, when unmodified, for small, highly volatile software development [56]. Especially if important requirements only become apparent during the implementation phase.

5.2 Prototyping

The prototyping model is an approach to software engineering that differs from the Waterfall model in that it utilizes prototypes, which are incomplete versions of products that can be reviewed by customers or end-users [72]. The basic outline of these steps are: identify requirements, develop prototype, review, and revise and enhance prototype. In the first step you identify the most basic requirements of the prototype. Develop prototype is when you develop the first prototype. Review is when the customers and end-users evaluate the prototype and provide feedback. The revision and enhancement of the prototype is when the feedback from the previous step is implemented, if the feedback is within the scope of the project. If necessary one returns to step three again.

This model allows for fast iterations of products that can be evaluated by customers and if the requirements for the product are unclear in the beginning, and becomes uncovered when the project is underway, these newly discovered requirements can be added during the cycle [72]. The framework chosen for this project is Scrum which is a type of prototyping model.

5.3 Prototyping: Scrum

Scrum is a framework that has seen use since the early 1990s. Its process control is based on the three pillars: transparency, inspection and adaption [57].

Transparency implies that all important aspects of the process must be understood, or “visible”, for the ones that are responsible for the outcome. To achieve this a common standard must be
created which all important participants understand. Inspection is important in the framework, but one should not inspect Scrum artifacts too frequently to avoid interrupting work. Adaptation implies that if the process or the object of the process deviates outside of the acceptable range it must be susceptible to quick change, or adaption.

The participants of the process are called the Scrum team [57]. This team must be as independent as possible. For this to be true all aspects of the object being processed or the process should be handled by the members of the team as far as feasibly possible. The Scrum team can be divided into three subcategories: the product owner, the development team and the scrum master [57].

The product owner are the one responsible for the viability of the Scrum team’s product or efforts. This person is the one accepting and setting the requirements for the product, mainly through the help of the product backlog. The management of the product backlog includes, quoted from the official Scrum guide [57]:

“Clearly expressing Product Backlog items;
Ordering the items in the Product Backlog to best achieve goals and missions;
Optimizing the value of the work the Development Team performs;
Ensuring that the Product Backlog is visible, transparent, and clear to all, and shows what the Scrum Team will work on next; and,
Ensuring the Development Team understands items in the Product Backlog to the level needed. “

The product backlog can thus be simplified as the tasks that need to be achieved in order for a viable product or product [57]. The product owner is the sole person accountable for this being handled correctly [57].

The development team are the ones solving backlog items during, and creating prototypes, (named in Scrum-terminology as an increment or a “Done”), at the end of each Sprint [57]. The sprint is what the Scrum framework is based around and where the actual work on the product is done. The timeframe for each sprint is a month or less if a feasible increment can be created within the shorter time period. Each sprint is started continuously after the conclusion of the previous one until the product is deemed finished. For something to be qualified as a sprint it must contain: the sprint planning, daily scrums, the development work, the sprint review, and the sprint retrospective [57].

The work that should be done during the sprint is planned at the beginning of it during the sprint planning [57]. The sprint planning is a session attended by the whole Scrum team that should never exceed eight hours. The work is chosen from the product backlog and the items chosen for that instance of the sprint is called the sprint backlog. The daily scrums are no longer than 15 minute sessions where it is decided what work should be done for the next 24 hours. The sessions are repeated at the start of each workday. The development work is what the development team does to achieve its sprint goals. At the end of the sprint a sprint review session is held that should not exceed four hours. Here the increment is evaluated by the scrum
team and adaptations of the product backlog might be applied. During the sprint retrospective the scrum team inspects itself.

The scrum master is the one responsible for making sure all the mandatory Scrum events are done during each sprint [57].

5.4 Development Model: Scrum

Scrum was chosen as the development framework for the thesis. Scrum is a prototyping model that is highly adaptable, which fits the project discussed in this thesis. It is a well-documented framework, making it easy for beginners to approach it and use. There are clearly defined roles. The purpose and responsibilities of these roles are also well-defined, thus minimizing the cost of applying the framework. The development team’s size in this project is very small though which will create some inefficient overhead but this overhead should be negligible.

In this thesis the product will be the CScore application. The requirements for CScore are set by the project manager from Iajans in collaboration with the chosen product owner picked from the scrum team. The requirements are set during several interviews with the project manager. For the first sprint where a schematic of the application should be developed a week of development time should be enough. Later sprints might become longer but this should be discussed during a later sprint planning session.

5.5 Design science

A couple of crucial points must be considered before an application can be designed. These points are called design theory or design science and there are different ways to approach them. It is a matter of preference to choose which method to use while solving these problems and a bit of research is necessary to have different options.

There are functioning design theories for many different branches and areas, therefore it is important to find the right approach for the project. It is possible to find many design theories to apply for mobile applications. Even if they might be good in different ways, it is better to pick only one of them to use as guidelines while designing. It is however beneficial to read and understand what other theories have to offer.

There are six different design principles that should be taken into consideration according to an article on vectips. These major topics are: hierarchy, repetition, contrast, shape, color, texture [53]. However, even though these principles are mentioned to be crucial, they are just guidelines to create a user friendly design. Therefore, they are not set in stone and working around them or ignoring them is an option too. To explain these points briefly:

Hierarchy is something to think about in every layer of the design. Whether it be the layout, the color or the shape, hierarchy is the point where the designer intentionally makes the user focus more on some parts of the design [53]. This is done to make the user find specific parts of the application more important. However, a majority of people tend to focus more on the top left part of an application layout and become less and less interested in parts further down the
bottom right as Figure 6 shows. The cause of this may be because of the way people from western culture read books.

![Figure 6](image)

*Figure 6. In western culture people tend to follow a top to bottom, left to right eye path as they take in information*[53].

Repetition is about making the user more comfortable and familiar with the design. This can be implemented in the different layers of the design, but mostly in color, size, shape and layout [53]. However, there is a warning about repetition being overused, since user may find the design becomes mundane.

Contrast is the way you can manipulate hierarchy, simply by making differences between two or three elements [53]. These differences can be change of size, shape or color and is also used in this thesis. Titles have bigger font size and are bold. However, this is not only used in texts, but also in websites and applications to direct the attention of the user.

Shape is classed as one of the most ambiguous design principles. The reason behind this is because it applies to all other design principles. To put it simply, everything created has a shape. However, it is important to think about the shape of icons and layouts [53]. It is crucial that they tell and represent exactly the things they are intended to. And all other design principles must be taken into consideration while deciding shapes.

Color can be one of the most important factors for a user to like the design. In addition to that, it is good to keep in mind that different colors can attract different audience depending on their culture, age, gender and so on. Different colors can have different meanings, they can invoke different feelings. For example blue is the most used color, there are many brands and sites such as Skype, Facebook that uses blue [53]. Supposedly, it is describing power and tranquillity.

Texture is the last principle that is mentioned. It is the surface or feeling of an object [53]. A popular texture that is used right now is flat design, giving the layout a simplistic look. It gives a modern and comfortable feeling.

These are guidelines that can help when designing an application but these principles alone are not enough. Similar applications, websites, programs must be observed to find some good inspiration and to see some working designs that do not only exist in theory.
5.6 Development environment

Integrated development environment (IDE) is a bundle of tools with the purpose of helping the developer to write code with less effort and with more efficiency. One of the perks is that the developer will not need to write every part of the code from scratch thanks to already written libraries. IDEs gives a better way to organize and save projects and a better way to debug the program [44]. However, it means that a developer must learn to use the IDE first which might take some time.

Different IDEs exist for different purposes and programming languages. The Android operating system has many different IDEs to choose from. Choosing the right IDE can create a big difference in the development. Some IDEs are simply better or more suitable than others. The newest and most used one for Android is Android Studio which is maintained by JetBrains and Google. It is based upon JetBrains’ IntelliJ and maintains most of the same abilities as IntelliJ [22 p. 25]. It got released in 2013, at the Google I/O event and got favored by many developers. Android Developer Tools (ADT) which was built upon Eclipse was the main development environment for Android before Android Studio.

Mobile application developers for Android prefers these two developing environments. However, Android Studio is the newer and more advanced one out of these two. Even though Android Studio is still new and is unstable compared to ADT, Google has announced that they will stop developing ADT by the end of 2015 [17]. This shows how much Google is investing in Android Studio and that it will be the primary choice of development environment for most in the future.

Software development kits (SDK) are part of IDEs. They are the tools with functions such as debugging, libraries, sample codes, autocorrect and so on. There are different SDKs for different operating systems. It is up to the developer to decide what SDKs to use. However, some of these SDKs are not optional. For example Android’s main language is Java. Java is a programming language that is object-oriented and class-based [49]. If the plan is to develop an Android program, Java Development Kit (JDK) must be used.

Optional SDKs can be useful and necessary too. Some of the popular SDKs for mobile application development are DDMS [11], Hierarchy Viewer [12] and more [13]. The most relevant SDK for this project is Android Studio Development Kit because its main purpose is to help developing a mobile application. It is embedded into the Android Studio IDE but even possible to download it separately.

When more than one developer works together on a project, a system called version control is needed. The developers must access the latest source files and see the changes that has occurred in them. It is also necessary when only one developer is working on the project, if the developer works on more than one computer. It is a system that helps to record different versions of one or more files and share them with a group of people [8 p. 1]. People with access can see all the different versions and changes and save their own version to the system for other members to see [8 p. 1]. This makes it possible to work on an application simultaneously.

---

5 Dalvik Debug Monitor Server
5.7 IDE: Android studio

The main platform of the product is Android and an IDE for Android must be chosen to work with. Android Studio is chosen for this project. Android studio is the newest and the most user friendly IDE available for Android. It is also Googles official IDE for Android. Android Studio has functions that helps mobile application development greatly. It has a starting guide [10] on their official site that helps greatly for developers that are new to the IDE. Testing Android Studio for a short while is enough to convince one that it is an IDE that has its focus on mobile application development. The downside is that it can be confusing in the beginning and it is easy to get lost but following the guides on Android Studios official site gets you familiar relatively fast. As seen in Figure 5, it is possible to see changes done on the code directly on a prototype mobile screen. It is even possible to change something on the prototype mobile picture without manually writing code, and automated code will be written according to the changes. However, this can cause some problems and unwanted code, but it is useful for small changes. This IDE is the best choice especially for developers who are new to developing mobile applications for Android.

Figure 5. Screenshot of Android Studio with code and mobile prototype window open.

Since the programming language for Android is Java, Java Development Kit must be downloaded for the IDE to function. Another SDK that will be useful is Android Studio Development Kit which is already embedded in Android Studio and offers great help for mobile application programming. Other SDKs are not useful for this project. For version control, a good choice is GitHub [24] because it is a stable and smoothly working system that is compatible with Android Studio.
6. Development of the CScore application

Before starting the first sprint some important basic requirements for the product is needed to be added to the product backlog. The requirements were extracted during the interviews with the customer and are compiled as easy to read bullet points below.

This is the requirements provided by Iajans for the mobile application:

- Real-time distribution of match information from current matches
  - Retrieve match information
  - Propagate match information to all instances of mobile application
  - Fast enough for user satisfaction
- Results of matches played that day
  - Store matches finished in the last 24 hours
- Upcoming matches being played that day and their information
  - Store matches being played in the upcoming 24 hours
- Push notifications
  - Choose matches
  - Create push notifications once match information has been updated
- Intuitive user interface
  - Two screens
  - Color combination of dark and light colors
  - Simple and straight forward design

6.1 Requirements in detail

A database will be needed in order to achieve real-time distribution of match information. This database will make it possible to store finished and upcoming matches for the day as well. Retrieving match information will be done by constantly working programs, “bots”. The way they will behave will be programmed by the project group and the mobile application server will be able to always run them. The most difficult requirement to implement will be the design since it has no definite solutions.

6.2 The first sprint

During the Sprint Planning session it was decided that a prototype of CScore should be developed, where these parts of the application should be sufficiently explained:

- The real system or the real-time distribution of match information
  - How to retrieve match information
  - How to propagate the match information
  - How to store the match information
    - Create conceptual and relationship model of database
- The layout of the application
  - Create designs
The points above were put in the product backlog as items and put at the top of the list implying top priority. The rest of the requirements extracted during the interviews that are not related to the items above were also put in the product backlog but as lower prioritized items.

Each morning, during the development work phase of the sprint, a short daily scrum were attended by the scrum team. Here the progress made on the items were discussed, what items should be worked on the next 24 hours were determined, any problems that might have arisen were discussed and plans made to deal with the problems were proposed.

6.2.1 The real-time system used in the CScore application

The distribution of current match information to mobile devices will require a custom made soft real-time solution. The match information is time sensitive since the user satisfaction decreases the longer it takes for it to reach the end user. The real-time solution will be a soft one since if the deadline is missed the instance of the mobile application will not toss the response. For user satisfaction to be on an acceptable level the deadline should be thirty seconds since the rounds of a Counter-Strike: Global Offensive match can almost never be shorter. The rounds are the most important match information, thus the deadline is based on them.

The external input for the solution will be a Counter-Strike: Global Offensive match. It is crucial that the information is retrieved and sent to the real-time system. Use of an API will be needed in order to achieve this functionality. Fortunately, ongoing professional matches have public servers called GOTV. The GOTV servers mirror the server that a match is played on, avoiding the need to release the match servers IP. If the match servers IP were public it could lead to nefarious attacks on the server halting the match. GOTV servers are available for everyone without restrictions. It will be easy to update information about matches in soft real-time if it is possible to retrieve data directly from GOTV.

A database is needed for the application in order to store all retrieved information about matches such as results, players and so on. For each change in any of the information on the match server, an aperiodic task should be triggered by the external input. The external input should trigger a response on the local database server which should update the correct match entry and attribute in the match database and a timestamp of when the match entry was updated. The instances of the mobile application will continuously check with the database server what timestamp relevant match entries has and compare them with the timestamp of the current match information displayed on the user’s screen. If the one displayed is older than the one in the match database retrieve the new one.

6.2.2 Retrieving match information

The application will be using two different bots in order to retrieve data from ongoing matches. The first bot, Bot1, is going to collect information about upcoming matches. The bot will go through different tournaments’ official sites in order to find every upcoming match. It will do it every thirty minutes. Bot1 will collect information such as team names, team logos, the date and time the match is going to start and the GOTV IP address of the match. This information will be saved in the local database.
The second bot, Bot2, will use the data collected by Bot1. It will connect to the GOTV ip address when the match starts and retrieve information about the match. This information will be retrieved manually or through the GOTV API if it exists. The collected information will be score, team names, team logos, player information, player stats, the map, alive players and time of collection. All this information will be saved in the local database and the mobile application will present the information properly. Bot2 and the application will work in soft real-time.

6.2.3 Local database construction
The database will contain all match information necessary for the mobile application and the real-time system described above. The database will need to contain information about all matches:

- The round score
- Information about teams
- The players and their statistics
- How many players are alive for each team
- When each instance was last updated

It is generally a good idea to create a conceptual model and thereafter a relationship model before beginning to build the database. A conceptual model of a database is an abstract visualization of a database. To create the conceptual model, Astah by the company Change Vision, Inc [4] will be used. The relationship model shows each table’s and attribute’s relation to each other. The relationship model will be created with the Dia Diagram Editor [15]. The relationship model will use primary keys and foreign keys. Primary keys are attributes in tables.
that make a row unique. Foreign keys are attributes in a table that links to a row in another table. The Database manager will be MySQL [48] since it is widely used and well documented.

6.2.4 Push notifications

An important function that will be available in CScore is Push Notifications. Push Notifications are a delivery of information from an application to the client without any specific requests. It can be in form of badges, sounds or custom text alerts. Many applications uses this to remind users about different things, sometimes for commercial purposes.

An application can send Push Notifications to all users or just a subset of users. It is a useful function yet many users who do not wish to get Push Notifications will be dissatisfied when they see one. Therefore, it is important to not spam the users and annoy them.

Push Notifications will be optional on this project. It will only be available when the user chooses to. The Push Notification choices will be “match start”, “half time” and “result”. The users will get alerts on their mobile phone when any of these specific event alerts are chosen. The notification will occur even if the application is closed or the phone is on standby. The phone will get a small alert once and the notification will pop up on the screen.

6.2.5 User interface

Design Science must be taken into consideration. The layout, colors and functionality must be as simple as possible to not overwhelm the user as the design science texture suggests. This way, the user can access the information sought in a fast and easy way.

The information that will be visible on the main screen is team names, round score for both teams if the game is alive, time of game start for upcoming matches and score result for finished matches. The information on the second screen will be player information such as name, kills, assist, deaths, MVP points, scores and even round scores, map, players alive, team names, team logos.

The designing process will start with a rough sketch of the mobile application. The layout was partially inspired by existing live score mobile applications. Most of the live score applications discussed in this thesis have a similar layout where the first screen shows the team names and the status of the match and the second screen shows more detailed information.
Some of the design science principles were taken into consideration while working on the sketch. Left part of Figure 8 is the Homescreen, the first screen that will be seen by a user that starts the application. The idea was to present all matches for the day, whether they are live, upcoming or finished. In Homescreen, the design science principles repetition, shape and contrast were taken into consideration. Repetition is used because all different matches presented on the screen should have the same repetitive layout in order to achieve a comfortable design where users can explore all different matches and find what they are looking for. However, most users will probably want to find one or more specific matches. To make it easier for users to find specific matches in Homescreen, the design science principles contrast and shape was used to differentiate every specific team. To make every match unique, team names will be presented in bold text in order to achieve contrast in the design. For the design science principle shape, every team will have their team logo next to their name.

Users will be able to see the second screen called Details when they click on a specific match on the Homescreen. The second screen will present information such as round score, player statistics and so on as seen on the right part of Figure 8. The main information that the user will seek in this screen is the round score and player statistics. To present this information, design science principle hierarchy and shape was used. For hierarchy, the most important information will be placed further up on the screen. In order to catch the user's attention further down, the options bar on the bottom will have a different shape. Having the layout like this will make sure that the user can see the shape difference and easily focus his attention on the bottom of the screen as well as at the top of the screen. The rough sketch was made in paint without any complicated details as Figure 8 shows. The purpose of this sketch is to see what should be achieved in the end, yet it is not set in stone and changes will probably occur in the future.
7. CScore prototype

There are two different parts of the application that must be designed and developed. The first part is the database, with its design and layout. It will be used constantly by the application and the server, and it will mainly be used by the back end of the application. The second part is the user interface, the front end of the application. The design must remain as simple as possible but offer all needed information at the same time.

7.1 Conceptual model

First a conceptual model of the database must be created. If this model is sound, minimal changes will be needed when implementing the database. The completed model can be viewed in Figure 9.

![Figure 9. The conceptual model of the database.](image-url)

All match information must be stored in the database so each instance of the mobile application can find all the relevant information. The Match-class contains all information about the match such as starttime, what league is hosting it, when some of the related classes were last updated, if the match has ended and the GOTV server IP. The TeamScore class contains the current score of each team involved in a match and how many players are currently alive, whilst the
PlayerStats-class contains the current statistics for each player. The Team-class contains the name of the team and file address to the team's logo. The Player-class only contains the Player name.

There are some rules for the model that are not immediately apparent. The updated attribute in the Match-class should get a new time every time an attribute have been changed in either the TeamScore class or the PlayerStats class.

### 7.2 Relationship model

Once a sound conceptual model of the database has been created it is time to translate it into a relationship model. In this translation one can easily find errors created by the conceptual model. The relationship model can be viewed in Figure 10.

![Figure 10. The relationship model of the database. Underlined attributes are primary keys and attributes starting with a '*' are foreign keys.](image)

The big change to the model when translating the conceptual model to a relationship one is the insertion of an auto generated primary key called matchId in the Match-table. This was inserted to make sure each match entry is truly unique. The same rule that applied to the conceptual model applies to the relationship model. Now that a sound relationship model has been created it should be trivial to create the database with MySQL.
7.3 Skeleton Frame

Now once the backend server’s database is outlined it is time to design the front end of the application. The sketch shown in Figure 8 will be remade one step at a time with boxes in order to separate the presented data and functions. The new remake will become the skeleton frame and it will make it easier to work with each individual part of the application.

Homescreen is divided into identical boxes called Box1 with the max width and twenty percent of the total height. Details is divided into three different boxes; MatchInfo, RoundInfo and Options. They all have max width. MatchInfo has fifty, RoundInfo has thirty and Options have twenty percent of total height. The different boxes and their layout are demonstrated in Figure 11 and they will be the main containers of information.

However, the containers must be divided furthermore in order to separate presentation of different information. Box1 will have the information about team names and logos as well as the information about the match. Therefore, three more boxes must be implanted into every Box1. As the left part of Figure 12 shows, the new implanted boxes will have equal width and height and their lines will be transparent to give the application a smooth look. They will be called Team1, Team2 and MatchBox.

MatchInfo will be divided into three boxes; Scoreinfo, Teaminfo and Playerinfo. RoundInfo will have three boxes to represent alive players for each team and round time. They will be called Alive1, Alive2 and RoundTime. Options will be divided into different boxes too, three boxes for push notification alerts and one box to go back to Homescreen. The new look of Details can be seen on the right side of Figure 12 with the boxes size and their names. As shown, the new boxes will have transparent lines except the boxes in Options.
It is possible to see the missing boxes when comparing the Figure 12 with the end sketch shown in Figure 8. ScoreInfo, TeamInfo, PlayerInfo needs to be divided into smaller boxes. ScoreInfo and TeamInfo will be divided in two each and PlayerInfo will be divided in twelve boxes. The result of the skeleton frame is shown on Figure 13.

**7.4 Implementing functions**

Now that the skeleton frame is done, functions can be installed on each box that exist. When the application is programmed, the data will be fetched automatically from the applications own database. The boxes begin value are null. If there are no matches for the day, no match will be
fetched and an empty screen with the text “No match found.” will be shown in Homescree. A test can be made by implementing made up data into the boxes. This will give the opportunity to see if everything works as they should be.

Looking at Figure 14 shows that there are some problems with the current design. One of them is the RoundInfo box being unnecessarily big and MatchInfo being too small. There is no space for player stats PlayerInfo and an alternative solution must be found. Homescree feels cramped and it does not have a smooth look.

Figure 15 is the attempted solution to problems that was found in Figure 14. Changing the skeleton frame was necessary in order to fit everything better. Minimized the size of RoundInfo and added it to MatchInfo. PlayerInfo is still cramped but it was possible to fit in player
statistics. Homescreen layout was changed to give a smoother look and RoundTime was erased from the prototype.

Figure 15 will be the latest iteration of the prototype. This version will be presented to the customer. Homescreen will present all matches for the day. The only information that will be presented in the Homescreen is the team name, team logo and round score of the match and the round score will update in real-time even in this screen. Details will open when a user clicks on a match in Homescreen and it will present all other information about a match that is available in this application. On top of the screen, round score team name and team logo will be presented as it was in Homescreen. Player statistics will be presented under team logos. Information about the same team will be presented vertically. Player statistics will show name, kill, assist and death score in this order for each player. The team coach, if there is one, will be presented under player statistics. If the match is live, alive players for the round will be available under the information about the coach. However, if the match is not live, the Alive tab will disappear and it will give more space for player statistics. Four different options will be available on the bottom. Users will be able to mark the first three options if they want to get a push notification. These options will be Match Start, Half Time and Match End in this order. When marked, the application will send a notification at the desired time of the match. The fourth option that is on the bottom right corner will direct the user to go back to Homescreen.

This version of the prototype does not have the right colors or font. It is focused on the layout and design. If it gets approved by the project leader at the sprint review session, the background color and texts fonts will be changed as well as any other changes project manager might ask for.

7.5 Evaluation

The latest version of the prototype was shown to the project leader for feedback. The project leader liked the new design on Homescreen but asked if it was possible to fit in information about the match, if the game is ongoing or finished and so on. It was decided that Homescreen would present maximum three matches on the screen, and the user would have to scroll down for more matches.

The design of Details was evaluated too. The RoundTime was discussed since it was a part of the original sketch and it was decided that removal of the RoundTime was okay. Furthermore, the project leader was not satisfied with the PlayerInfo because it was cramped and an alternative solution was discussed. The other ideas was to have one player on each row, having ten rows instead of five. This would however cripple the design a lot and tangle up the whole layout. It was decided to keep it as it was.
As for the colors, black or grey background and white text color was requested by the project leader. The color orange was chosen by the project group to make the application similar to counter strike colors, in order to get users familiar with the colors as the design science principle color suggests. The fonts was asked to be bold and big but no specific font was requested. The final prototype after the evaluation is shown in Figure 16.

The evaluation discussed above corresponds to the phases: sprint review and sprint retrospective. The phases are mandatory, and important, for a sprint to be considered complete, together with the phases: sprint planning, daily scrums and development work.

### 7.6 Prototype discussion

The application was originally planned to show round time in the form of minutes and seconds, since a round is one minute forty five seconds long, thus forcing the application to update every second. But the real-time that will be used in this mobile application will not be fast enough to update the time every second. This means that the round time must be updated in longer intervals and that might not give a satisfactory experience to the end-user. However, the round time is one of the less important information variables that a user seeks. Therefore, it was okay to remove the RoundTime completely from the mobile application.

Similar to the round time problem, the amount of alive players might cause difficulties too. A player can die any second and more than one player can die simultaneously. In some instances,
a minute can go by without any casualties in the match. Therefore, working with near real-time can be sufficient enough. Information about alive players is decided to be kept until it can be tested in the application. In eventual failure, information about the alive players might be removed.

Player statistics in the PlayerInfo box is cramped and no alternative solutions was found. It is a future problem that cannot be solved at this moment because solving it means to redesign the whole second screen. The design can be redone in future iterations if the current design is decided to not be satisfactory.
8. Conclusions and future work

This thesis presents the procedure to develop a real-time mobile application for e-sports, called CScore. This thesis was done for a company called Iajans. A representative for Iajans was assigned to this project as a project manager and was giving out requirements for the application. The main focus of this project is on the e-sport game Counter-Strike: Global Offensive. The term e-sports applies to too many videogames otherwise and this project does not have the necessary resources for development of more than one game. The application discussed in this thesis is a “live score” mobile application for e-sports. Live score applications displays the score of a professional match, in at least soft real-time.

In order to develop a proper live score mobile application, the work started with a research. The research done was on real-time systems, real-time applications, mobile applications, mobile platforms, design science, servers, development environments and e-sports, especially Counter Strike: Global Offensive. The result of these researches helped in creating a user friendly design and functions, and made the development procedure clear. Android was chosen as the mobile platform to work on and development environment was chosen to be Android Studio with Github as version control.

Some methods had to be chosen before starting the development; a research method, data analysis methods and a system development method. What the data collection should mainly consist of also had to be decided. The research method was chosen be an applied research method. The data collection gathered was a mix of interviews and texts, whilst the data analysis methods was chosen to be a mix of coding and analytical induction. Scrum was decided to be the framework of the system development method for the project.

During the first sprint a schematic for the CScore real-time system, a design of the mobile application with an intuitive user interface and a relationship model for the required database was completed. No programming was done and no real functionality was implemented for CScore.

8.1 Discussion

The purpose of the thesis have been accomplished although the development did not come very far. In chapter seven, the steps taken to create the first prototype are clearly outlined and explained. The purpose of the application have not been fully achieved since the first prototype of the application does not provide any functionality. What the first prototype do provide is an easy to understand schematic of how to build an application that would fulfil the outlined purpose. This was achieved by the extensive research done on real-time systems and mobile application development. However, the purpose for the company Iajans was to get a feel for the e-sports application market and to get somewhat involved in the e-sports community. This has been achieved by this thesis research on the current state of the e-sport market, which provided much useful information.

The goal of the thesis was to create an understanding of how to create a real-time mobile application for a new market, e-sports, which has been accomplished by this thesis research on
the subject of real-time and e-sports. The goal of the application itself was to provide simple and fast information about currently ongoing or upcoming e-sport matches. It is not possible to say that the goal was fulfilled because only a prototype exists. The prototype does not provide any end user functionality as mentioned before. However, the goal for the company iajans was to get a feel for the e-sports application market and to get involved in the e-sports community. This has been achieved by this thesis research on the current state of the e-sport market, which provided much useful information.

This thesis does provide a simple and clear procedure on how to develop a real-time mobile application with functionality decided by an external customer which is what the problem statement in this thesis is seeking. Although the first prototype only outlines how to implement the functionality, the thesis communicates how to proceed. Take for example the database. In the first prototype only a relationship model have been created of it. But since the structure, and all the relations of the database is known it would be trivial to implement it with whatever database manager one deems the best. The database manager chosen for the continuation of this thesis is MySQL though. So even though the application has not been fully developed, it has been explained how to do it, and thus the main problem has been answered.

The delimitations on this project can be divided into four distinct delimitations. The first delimitation, that CScore’s requirements will be set by the project manager assigned by Iajans, had a big impact on our prototype since most requirements were provided and conceived during our discussions, or interviews, with the product manager. The product manager influenced most of the requirements. The second delimitation only influenced what IDE were decided to be used for the development of the mobile application. The third delimitation highly affected our product though, since the game chosen, Counter-Strike: Global Offensive, makes our implementation of how to retrieve match information and display them specific for just that game. The fourth delimitation, that the real-time system were to be a soft real-time system, created a big amount of freedom in how to implement the real-time system since the requirements for a soft real-time system are so lax, which made the development of it easier.

There are three methods that were chosen in this thesis, a research method, a data analysis method and a system development method. For the research method an applied research method were chosen. It was a perfect fit because of the applied computer science nature of creating the CScore system. The data collection were a mix of interviews and texts which provided enough data to accomplish the main problem stated in this thesis. The data analysis method used to analyse the collection were a mix of coding, for the interviews, and analytical induction, used on the texts. The data analysis methods created a good understanding of the data collected and thus were good methods for this thesis. For the system development method Scrum were chosen. It was a well-documented framework, but because of the small development team created some inefficiencies since multiple roles needed to be assigned to one person.

The work done on this thesis is a good example of development for a real-time mobile application. It shows the working procedure and train of thought used when working with an application in a new market. The prototype is promising but competition is bound to become fierce. When bigger companies with more resources takes upon themselves to explore these areas, this application might pale in comparison. However, it contains lots of functions and possibilities that can be useful for continued development of this application.
8.2 Future work

The application can easily be programmed since most details are explained. For continuation of this application, approximately 130 hours of programming will be needed to fully develop the database and the application. We estimated twenty hours work on database, fifty hours work on the bots, twenty hours of work on the layout and thirty hours of work on functions in the application. These number are just educated guesses and the numbers might be less when experienced developers work with the same material. The project group worked on this paper will continue with the development and hopefully get the application out in the market before summer 2016.

Adding additional functionality is an option when the application is fully developed. There are some functions that can be very helpful but is not necessarily important for a live score application. One of these functions is the ability for users to follow one or more teams. The idea behind following teams is to notify the user with push notifications every time their team has a match that is about to start. This can be a great function for fans and it can be a nice way to remind users about the ongoing matches for the day.

Another thing that can be developed with this application is a big database where every score and information about the matches can be store. This database can later be presented as statistics or history for the users.
References

   [Accessed 27 October 15].
   [Accessed 27 October 15].
   [Accessed 27 October 15].
   [Accessed 27 October 15].
32. Google Play. theScore eSports.
   [Accessed 17 December 15].
   [Accessed 17 December 15].
   betting-explained.aspx. [Accessed 27 October 15].
   Integrated Development Environment. Distributed Frameworks for Multimedia Applications,
   2006. The 2nd International Conference.
45. KTH Primo. http://kth-
   primo.hosted.exlibrisgroup.com/primo_library/libweb/action/search.do?vid=46KTH_VU1&pr
   efLang=sv_SE. [Accessed 29 January 15].
47. Miles, M. B., Huberman, A. M., & Saldana, J. (2014). Qualitative data analysis: A methodsˇ
   [Accessed 29 January 15].
   15].
   11 October 15].
   Routledge
   December 15].
   Dorling Kindersley (India).
   15].


Appendix A

Interviews

First meeting with company representative:
Location: Skype conference call.
Date: 2015-9-25
First meeting with the project leader was about what the company was expecting from the project and how they wanted the project to work. We asked questions about deadlines and if the company had any preferences on basic functionality. The project leader was not familiar with the subject. The requirements for the application after the first meeting was: Show scheduled matches, show current score of live matches, and enable push notifications for matches of interests. The project leader asked us to choose a main platform for the application too. We were mostly interested in Android but we would make sure it was the right choice before next meeting and give our final decision. The next meeting was planned for 2015-10-14.
Requirements for the next meeting:

- Project proposal with proper design, layout, possible functions and a rough blueprint.
- Main platform choice

Second meeting with company representative:
Location: Skype conference call.
Date: 2015-10-14
A design and color choice was presented to the project leader. The colors got a pass, some changes was wanted in the design of Homescreen. Details screen was also presented and it was accepted. We said that we wanted to work on Android.
It was asked if it was possible to save old results and information in a database that can be used to present different statistics. If that was possible, we would create some new functions such as searching for teams or tournaments to see their history, played matches for the past week and so on. We had to think about it, while it is possible it would mean that the applications purpose was more than what it started out with. The project leader wanted a meeting in one week but we had to focus on data collection and research for a while. Next meeting was planned for 2015-1025.
Sub-requirements for the next meeting:

- Newer version of Homescreen with small changes the project leader wanted
- Decision about possibility of a database to store old data

Third meeting with company representative:
Location: Skype conference call.

Date: 2015-11-10

The meeting was delayed two weeks. However, it continued where it ended last meeting. The new designed was presented and it was accepted. Database to store old information and presenting statistics was discussed. The database was possible and we could save the information from the matches but we did not want this application to present the statistics. We could however store the information for future use either implementing in this application or creating a new application just for that sole purpose.

We asked some direct questions to get information about our some concrete functions and requirements. The main requirement was to retrieve the match information and present it to the users in a real-time. This would be able for all professional matches for both Europe and North America, which means day and night. Additionally, upcoming daily matches and their results must be visible for the users the whole day. And push notification should be optional for all users.