Soccer Game Simulation

Project Report

By

CMPE 195A

Gibran Morales

Parag Pardeshi

Carlos Toledo

[Project Advisor's Electronic Signature]

_____

[Your Supervisor's Name], Project Advisor

mm/yyyy

ABSTRACT
Soccer Simulation

By Gibran Morales, Parag Pardeshi, Carlos Toledo

Electronic arts have long been the preeminent developer of state-of-the-art sports game simulations. This game sometimes serves as an important forecaster tool for those with serious stakes in the world of soccer. This game sometimes serves as an important forecaster tool for those with serious stakes in the world of soccer.

Although strikingly entertaining with their advanced graphics depicting reality with a fantastic level of realism, their soccer simulation is still confined within the parameters of the world of videos games. When generating simulations, EA FIFA's results are dependent on a system that was designed primarily around letting the users manipulate the outcome of the game. For these reasons, their game model is suspect as they commonly fail to accurately predict a winner of a real-life professional soccer game.

Our objective is to reduce the gap between the intangible areas of soccer and the measurable areas. By incorporating in–depth soccer theory free from the intruding structure of user-input, we can generate a more accurate simulation. Ultimately, we want to predict a winner with a higher percentage of than EA FIFA. Additionally, we will employ simple graphics that emulate the soccer game. These graphics serve two purposes. Firstly, we want to want to have an easy to understand visual representation of the complex background calculations. Secondly, we want them to resemble the dynamics of the game as close as possible for an aesthetic quality in our simulation.

Table of Contents

Chapter 1 Introduction

1.1 Project goals and objectives

This project seeks to predict the outcome of a professional soccer match while mimicking the game dynamics, organization using sound soccer theories. Further, the goal is to be able to offer the user a way to see not only what is the most likely outcome of a soccer match, but also what had to happen in the match in order for that outcome to happen. The project will focus on the historical data of previous soccer games, physical attributes and statistics of the players participating in the match. With this information, the software will then predict the outcome of a match taking into consideration specific factors that can be considered game changers. Because specific player statistics will be taken into consideration, the line up of a team will be of significance on every match simulation. Whether the best player starts the game, or a player who usually gets yellow cards is playing will shape the outcome of the match. In addition, factors like weather and the referee's leniency will also be taken into consideration.

1.2 Problem and motivation

Soccer is a professional sport played by large amounts of people all along the world. This and its fan base make soccer the world's most popular sport. Soccer is also a very complex game that involves different factors that make it very difficult for people to predict scores. The game's outcome relies on the performance of 22 different players who are actively playing for 90 minutes. Each of these players can become a game changer by being key goal scorers or by committing a red card deserving penalty or any penalty in the goalkeeper's area. In addition to the 22 players, other factors also make soccer a hard game to predict outcomes. Different strategies used by the coach, the place where the match is taking place, the mood of the audience

and the already mentioned weather and referee could alter the course of a game at any time during the match. Given all these circumstances and that soccer scores are usually small, soccer is a sport where anything can happen and where accurate score prediction is hard to achieve in any regular match. A tool like this will facilitate people seeking score predictions and soccer games simulations.

1.3 Project application and impact

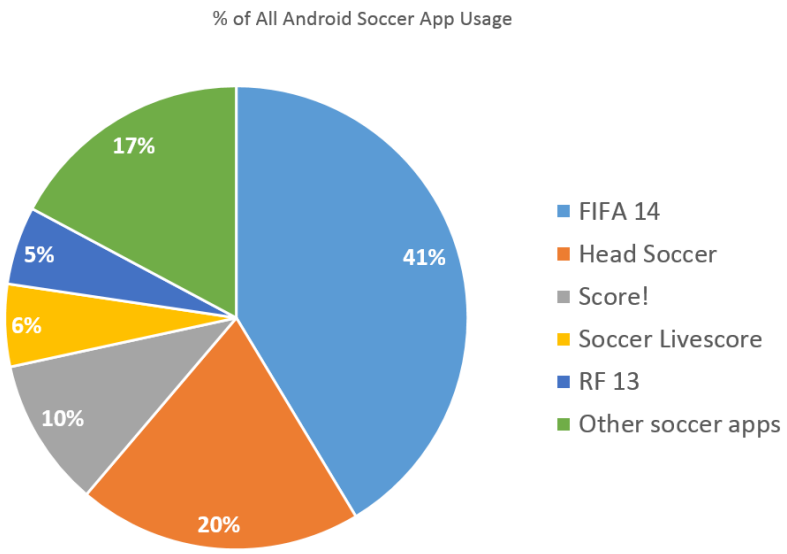
This project could be used as a tool for anyone who will like to see a simulation a soccer match. In academia, this project would take professional sports simulation to the next level. Student will see how to include different factors besides historical data to predict sports outcomes and how important these can be to the final score of a match. In the sports industry, this project will allow for advanced software to get to the hands of team coaches. The coach of a team can benefit from this tool by analyzing different strategies and deciding what might not work and what might work best against an opposing team. In other words, the efficiency of using certain strategies such as what the perfect starting line-up would be against a specific opposing team, or what conditions would allow for certain player to perform their best would be able to be measured by this tool. In society, regular people not necessarily working in the soccer industry could use these tools for their own entertainment. In addition, this tool can be used in people who bet on the outcome and events of soccer games. Since this tool will not only predict a soccer match score, but also show how this score will happen, betting on specific events happening will make a game more interesting to these people.

1.4 Project results and deliverables

The first deliverable of this project will be the report. The report will explain in detail every aspect there is to know about this project. In addition the report can be used as a document to fully implement a soccer simulation system. After the report has been completed, the result of this project will be a stand-alone fully working version of the software that will have the capabilities of predicting the outcomes of a soccer match. Together with the outcome the software will be able to narrate some actions that are of importance in the match. This software will run on a computer independently and will only need user input (e.g. two soccer teams) in order to work.

1.5 Market research

The main companies with similar products are those that develop soccer video games. Across most game enabled platforms, various sources list either Winning Eleven or EA's FIFA as the most widely played soccer games. An article from AppMonitor lists FIFA Soccer 14 as number one in a list of the 5 most played Android soccer games and apps. The other apps and their percentage are show in the following figure:



In addition, there have also been student project that have aimed to predict the outcome of local soccer tournaments. For example, the report 'Forecasting A Winner For Malaysian Cup 2013 Using Soccer Simulation Model' explains the findings of students from Salford University. They were able to deliver a fully functional software that predicted a winner for the Malaysian Cup of 2013.

Chapter 2 Background and Related Work

2.1 Background and used technologies

What makes soccer an exciting sport to watch is that the outcome is really hard to predict. The game has shifted from being an offense vs. defense game towards a more strategic one, therefore creating more competitive game. Even though more powerful, faster, and strong players give a team some advantage, most goals are now achieved due to player errors in the execution of certain strategies. Therefore scores are usually under five goals in total. In conclusion, for a soccer simulation to accurately

predict a soccer game, strategy has to be taken into consideration. A game cannot be simply predicted based solely on a player's strength, the player's team interaction is as important as any other factor.

Besides team strategies, there are other factors that interfere with a game's outcome. One of these is weather. Rain will definitely affect a match's outcome, mainly by altering the ball's trajectory after it is hit. In addition, rain will also slow down the player, and is prone to generate more errors due to players falling. Similar to rain, hot days also influence the game, to the point that the recent world cup had to incorporate hydration break during a match, this something that has never been done before in the FIFA world cup. Besides weather, other factors that might affect the match might be a change of coach, since a coach is the one that influences the games strategies the most. In addition, player injuries, player banning, and player banning are as important since team interaction has to be modified every time that a player is changed within a team.

One of the most important parts of this project are the mathematical calculations that go behind every play. Therefore extensive statistical and mathematical calculations are performed to come up with a score.

2.2 State-of-the-art

In terms of soccer simulations, the most known place to look for is in video games. FIFA by Entertainment Arts and Pro Evolution Soccer by Konami are the most popular soccer video games out there. They both go deep into the soccer games and

calculate even the smallest detail in order to provide a realistic opponent. One factor to take into consideration however is that video games are focused on the user experience. This means that as realistic as the simulation may be, it is still tailored to work around the users input and behavior. Nonetheless, both FIFA and PES take into consideration every single detail when playing against the human gamer. For example, the outcome of the game depends on how strong the opposing team is. This strength is derived by the individual abilities and weaknesses of each of the players that compose the team. In addition, every single move performed by the players takes the game in a specific direction. For example, a kick to the ball might be a pass, long pass, or a shot to the opposing team's goalkeeper, and even in these instances they can either connect, be intercepted, or miss. In order to predict the outcome of the kick, certain factors need to be taken into consideration. These factors include: strength of the kick, initial velocity, position of the player, direction of the ball, and air resistance. An article called FIFA Physics published by Scientific American explains the importance of the detail in the physics used in the game. The article takes as an example air resistance and describes how a more accurate drag coefficient would lead to a better modeling of air resistance; which would ultimately offer the ball the realistic acceleration and spinning effects that causes it to curve. Games like FIFA and PES take soccer simulation to the next level in its use of physics and detail integration.

A project aiming at something similar to the aim of our project was described in a scientific journal published by the AIP Publishing and presented as a journal in the AIP Conference Proceedings webpage. The AIP Conference Proceedings encompasses a

number of scientific journals published by the American Institute of Physics. Students at the University of Salford in the United Kingdom developed the project. The focus of this journal is to use statistical models to predict an outcome to the Malaysian Cup of 2013. That particular project had to pay particular attention in the structure of the Malaysian Cup of 2013, which, similar to other soccer tournaments, started with groups out of which the top two teams of each group would go on to the elimination phase of the game (quarterfinals, semifinals, and finals) until there is a champion. They focused on the outcome of individual matches and used a Poisson model to predict the probability of a team winning a match. In addition, used historical data on the outcome of matches to estimate the attacking strength and defensive weakness of all teams. They programmed the simulation using MathCAD and ran it 5000 times to get a more accurate outcome. Compared to the actual outcome of the cup, their overall results were about 50% accurate in terms of the team positions during the group phase and the overall winner of the cup. They accurately predicted the exact group position of eight out of sixteen teams, however this included the prediction of seven out of the eight qualifying teams. Their work gave this project some insight as to what to aim for. This project however did not take into consideration certain factors that would impact the strength and weaknesses of the teams, such as: player injuries, change in team formation, change of coach, weather condition. Although including all those factors would be too ambitious, our project includes some of them such as team formation and player injuries taking this simulation to the next level.

2.3 Literature survey

Chiaet, J. (2013). FIFA Physics. *Scientific American*, *309*(6), 19.

This article explains the details used in EA's FIFA video game. The article states that the physics is what ultimately gives the video game the realistic feel and that it includes a vast number of factors, from the players kick strength, to the ball's curve as it comes down. It is also mentioned how simple physics errors could lead to wild results and how achieving reality is a hard task to accomplish.

Peterson, D. (2014, January 1). 5 Most Played Android Soccer Games and Apps. Retrieved December 10, 2014.

This article lists the 5 most popular android soccer games and apps. The article published on App Monitor lists FIFA Soccer 14 as the most popular soccer app.

Karlis, D., & Ntzoufras, I. (1998). Statistical modeling for soccer games: The Greek league. *Department of Statistics, Athens University of Economics And Business.*

Karlis and colleague describe statistical modeling for soccer matches. The talk about various aspects of soccer including: soccer data, Poisson distribution, the home effect, sport statistics, and goal independence. Their paper overall explains the effectiveness of using a Poisson distribution to model soccer games. They explain some of the problems with such statistical models, like outside factors, and how to actually do the model.

Yusof, M., Omar Fauzee, M., & Latif, R. (2014). Forecasting A Winner For Malaysian Cup 2013 Using Soccer Simulation Model. *AIP Conference Proceedings*, *1605*1153-1159. doi:10.1063/1.4887753

This document gives a detailed description of a similar project developed by students at the University of Salford. They develop a model to predict the outcome of the 2013 Malaysian Cup. They explain in the document what data, statistics and statistical models they used to perform their predictions. In addition they include detailed assumptions and factors that were not taken into consideration at the time of their project. They also present their results and compare them to the actual outcome of the Malaysian Cup.

Chapter 3 Project Requirements

3.1 Domain and business requirements

Describing it at the highest level, the system will be as easy to use as pressing a "play" button and sitting back and watching the game unfold. There is essentially no user interaction other than starting and ending the application. The design of the system shall be able to easily incorporate new teams by simply loading different data. In other words, the simulation is not tailored to any specific playing style of any team. It uses the same artificial intelligence,

algorithms, and all other game procedures every time, no matter what team is loaded to carry out the simulation.

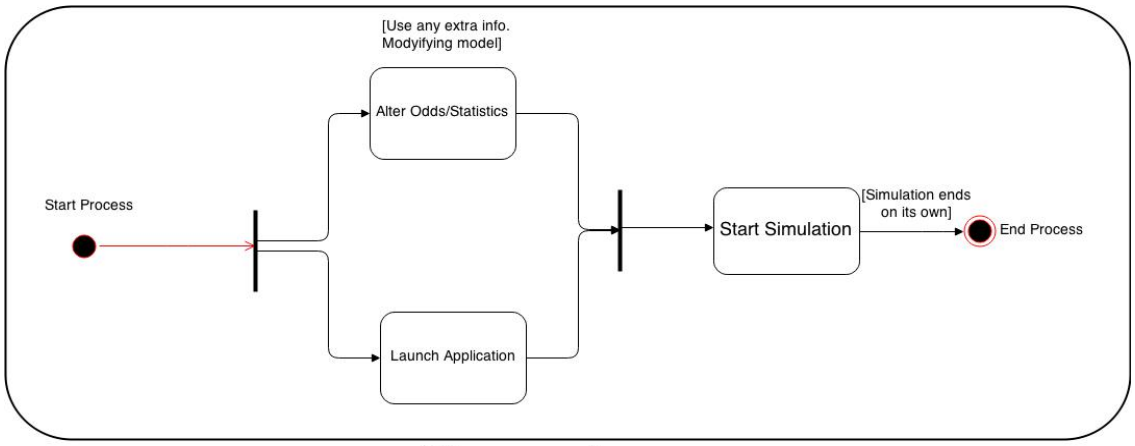


Diagram 1

A process summary of the entire system is depicted in Diagram 1 above.

3.2 System (or component) function requirements

Note: In the requirements below, each reference to 'player' means the virtual soccer player being emulated in the simulation. It does not refer to a user of this system playing or using the system in any way.

3.2.0.1	The Soccer Simulation System shall not have game delays.
3.2.0.2	The Soccer Simulation System shall work on all new versions of OS after 2012.
3.2.0.3	The Soccer Simulation System shall have an easily distinguishable shape in the animation.

3.2.1	The Soccer Simulation System shall display all 22 players at a time.
3.2.2	The Soccer Simulation System differentiates teams by using different color uniforms.
3.2.3	The Soccer Simulation System shall be in 2D format.
3.2.4	The Soccer Simulation System distinguishes which direction a player is facing.
3.2.5	The Soccer Simulation System has 11 players on each side at all times.
3.2.6	The Soccer Simulation System shall be able to change and display the movements of each player.
3.2.7	The Soccer Simulation System shall not take any input from user when the game is in play.
3.2.7	The Soccer Simulation System allows a player to make his own decision when attacking (when possession of the ball)
3.2.8	The Soccer Simulation System allows the player to make his own decision when defending
3.2.9	The Soccer Simulation movements should be slow enough to allow the user/viewer follow the ball and players.

3.2.9	The Soccer Simulation System shall perform as per the official FIFA rules.
3.2.10	The Soccer Simulation allows the player to dribble
3.2.11	The Soccer Simulation allows the player to shoot at the goal when he is at a reasonable distance from it.
3.2.12	The Soccer Simulation System allows the player to pass based on a decision-making algorithm where he analyzes the positional layout of everyone on the field.
3.2.13	The Soccer Simulation System shall display the proper score when the game is in play.
3.2.14	The Soccer Simulation System makes it visually clear when a player shoots, meaning a different ball picture is loaded for the texture.
3.2.15	The Soccer Simulation System allows a defending team to let their players decide when they should go after the ball.
3.2.16	The Soccer Simulation is played on a field that is 1150 X 700 in pixels.
3.2.17	The Soccer Simulation is played on a field with the boundaries and lines specified by the FIFA's official rules.
3.2.18	The Soccer Simulation System does not let the goalie dribble to far away from his original position.

3.2.19	A goal scoring success rate is a function of the position of the goalie and the angle of the shot towards the center of the goal.

3.3 Non-functional requirements

3.3.1	The Soccer Simulation System shall work on all new versions of OS after 2012.
3.3.2	The Soccer Simulation System shall use latest version of SDL.
3.3.3	The Soccer Simulation System shall be compatible with and OSX.
3.3.4	The Soccer Simulation System shall use OpenGL.
3.3.5	The Soccer Simulation System shall be able to expand the number of teams.
3.3.7	The Soccer Simulation System shall be built using Code.
3.3.8	The Soccer Simulation should not crash more than once per simulation.

3.4 Context and interface requirements

This section identifies which subsystems interact with other subsystems. In our project, the independent interfacing modules are sharply defined as our structure closely resembles MVC.

3.4.1	The PC running Soccer Simulation System shall be connected to a monitor to display the game play. For the best viewing experience, you can connect a PC running the simulation to a TV with an HDMI connection.
3.4.2	Graphical system needs data supplied the business logic system. For the sake of clarity, the business logic module consists of soccer mathematical theories and formulas specifically developed for this project.
3.4.4	A simplified narration (text) of the game accompanying the animation will be displayed to the user.

3.5 Technology and resource requirements

3.5.2	Any PC with a reasonably recent microprocessor will suffice.
3.5.4	GCC C compiler
3.5.5	OpenGL release 4.5
3.5.6	Simple Directmedia Layer (SDL) release 2.0.3

Chapter 4 System Design

4.1 Architecture design

Our system does not employ a user interface because all the required data is loaded into memory. In other words, player attributes and other similar statistics are hard coded. Initially, we had intended to use a database to query information but since our project to the shape of a game, we decided to also design it like a game. The main factors influencing this change were the obvious time it would take to query information while the simulation was running. Still, our general system architectural overview resembles a Model-View-Controller structure. However, the typical function of a "Controller" component is replaced in our system since there is no need for a user to alter any persistent data. In fact, we can even omit the controller part and still make sense of the primary interacting elements. Alternatively, we can think of the of the controller part of our architecture as a feeding reminders to our internal formulas about the graphical layout.

As already mentioned, the driver for our system in an ongoing feeding of data into formulas. Therefore, close attention to detailed and good amount of effort will be required for the model component of our design. To reiterate, the flow of persistent data will only be in one direction--towards the view side--and no user input will be going in the opposite direction. This is an important fact to keep in mind because it keeps the development team on track and avoids pitfalls in implementation and solutions. More specifically, the Model will consist of an array of formulas developed to measure and

describe soccer game-play, and these formulas will take the hard coded players' ratings as input in order to emulate the dynamics of soccer game-play.

Figure 1 illustrates an overview of the proposed architecture.

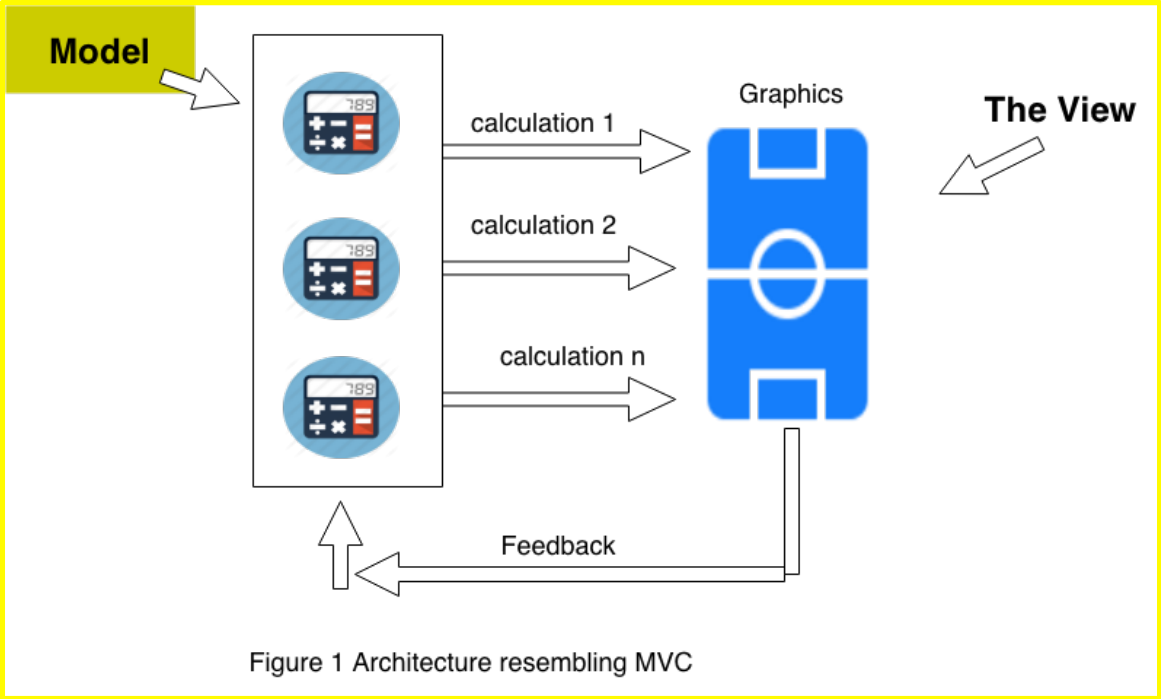


Figure 1 Architecture resembling MVC

4.2 Interface and component design

This project had its genesis in wanting to calculate and measure a sport, which is a task that can often times be considered immeasurable. In order to provide some aesthetic qualities, we will employ an animation of simple 2-dimensional graphics. The closest component resembling a user interface in our system will be the display of a 2D animation mimicking two teams in a field playing soccer. Some non-critical interface features like pausing the game were discarded in the development of our project. User

friendliness will take a back seat in our design as the vital components are behind the user's screen. In addition to managing the simulation, a useful feature will be to have an accompanying text section that describes with short statements specific game plays or situations that might be difficult to realize are happening. Figure 2 illustrates a mockup of what the user will see.

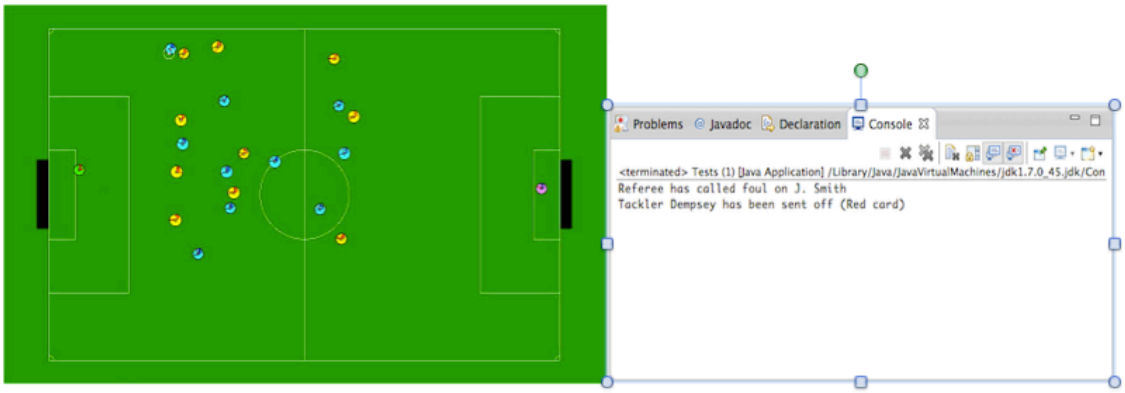


Figure 2. The View

Like already mentioned, we abandoned the use of a database. We figured there is no need to create tables representing control objects like `playerStats`, `fifaRatings`, and skills. Instead, we will speed up our simulation by not using a database and loading up everything to memory as soon as the application starts. SQL Queries will be replaced by setting up all kinds of accessor methods to important control objects, and using these to systematically retrieve relevant data.

Figure 3 shows the general flow of information. In contrast to typical user input oriented systems where information flows in both directions. Here we see that information is

directed upwards towards the view.

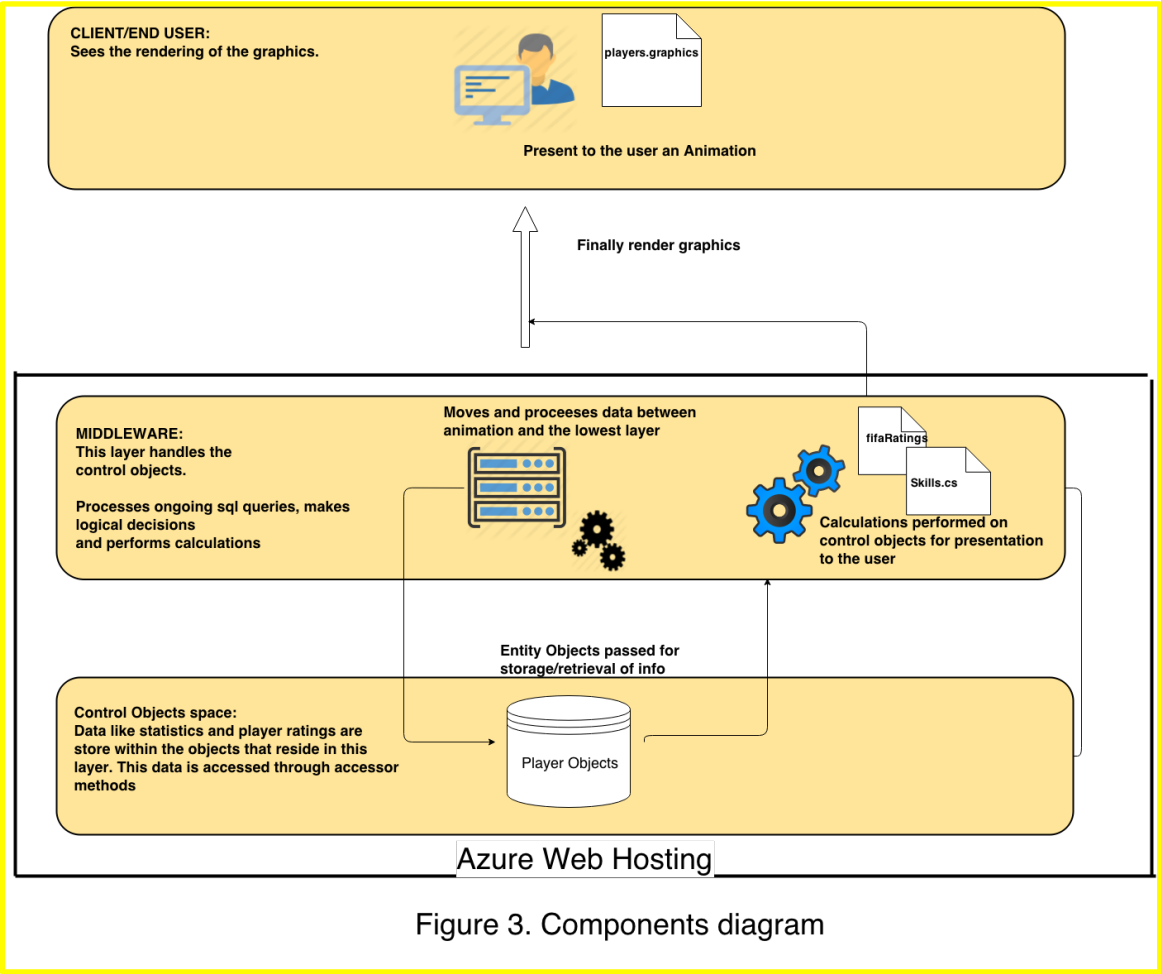


Figure 3. Components diagram

As we can see, the system is composed of three primary parts which can be

summarized as follows:

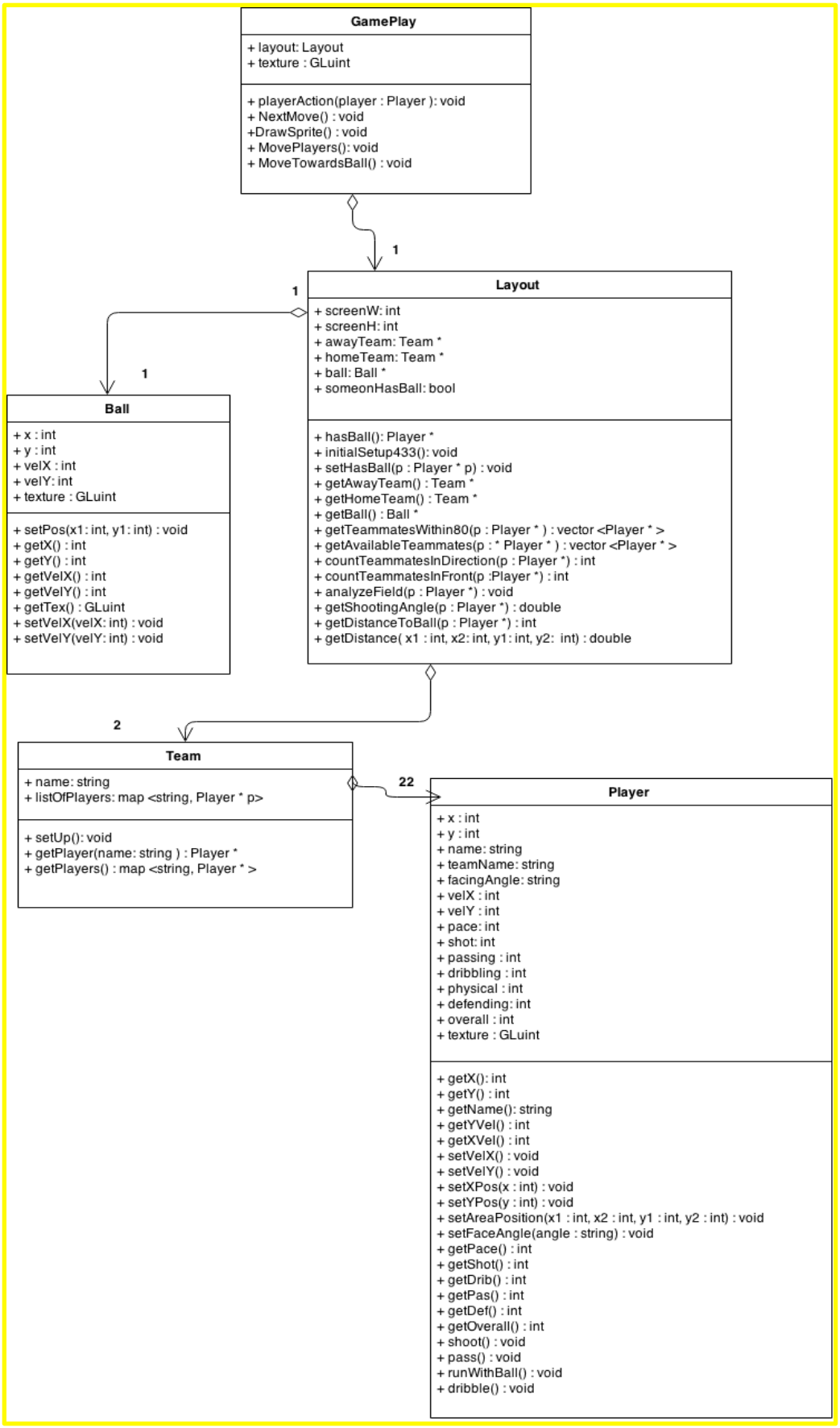
- *Model Layer:* Takes the place the database, or mimics the functions of the initial plans of the database. This is the lowest layer and where everything starts moving. This level can be seen as input to the system.
- *Middleware:* Processes the business data and executes logic. At this layer, the information flows only in one direction, namely up to the view. This is because there is basically no user interface. All the input comes from down below this layer.

- *View Layer:* The culmination of an array of ongoing calculations. The result of each computation is used to influence an animation in 2D graphics. As Figure 3 indicates, the flow of information is one way at this level too.

4.3 Structure and logic design

UML Class Diagram

The following class diagram shows the relationship between the control objects.


GamePlay
+ layout: Layout
+ texture : GLuint
+ playerAction(player : Player): void
+ NextMove() : void
+DrawSprite() : void
+ MovePlayers(): void
+ MoveTowardsBall() : void
1
1
Layout
+ screenW: int
+ screenH: int
+ awayTeam: Team *
+ homeTeam: Team *
+ ball: Ball *
+ someonHasBall: bool
+ hasBall(): Player *
+ initialSetup433(): void
+ setHasBall(p : Player * p) : void
+ getAwayTeam() : Team *
+ getHomeTeam() : Team *
+ getBall() : Ball *
+ getTeammatesWithin80(p : Player *) : vector <Player * >
+ getAvailableTeammates(p : * Player *) : vector <Player * >
+ countTeammatesInDirection(p : Player *) : int
+ countTeammatesInFront(p :Player *) : int
+ analyzeField(p : Player *) : void
+ getShootingAngle(p : Player *) : double
+ getDistanceToBall(p : Player *) : int
+ getDistance(x1 : int, x2: int, y1: int, y2: int) : double
1
Ball
+ x : int
+ y : int
+ velX : int
+ velY: int
+ texture : GLuint
+ setPos(x1: int, y1: int) : void
+ getX() : int
+ getY() : int
+ getVelX() : int
+ getVelY() : int
+ getTex() : GLuint
+ setVelX(velX: int) : void
+ setVelY(velY: int) : void
2
Team
+ name: string
+ listOfPlayers: map <string, Player * p>
+ setUp(): void
+ getPlayer(name: string) : Player *
+ getPlayers() : map <string, Player * >
22
Player
+ x : int
+ y : int
+ name: string
+ teamName: string
+ facingAngle: string
+ velX : int
+ velY : int
+ pace: int
+ shot: int
+ passing : int
+ dribbling : int
+ physical : int
+ defending: int
+ overall : int
+ texture : GLuint
+ getX(): int
+ getY() : int
+ getName(): string
+ getYVel() : int
+ getXVel() : int
+ setVelX() : void
+ setVelY() : void
+ setXPos(x : int) : void
+ setYPos(y : int) : void
+ setAreaPosition(x1 : int, x2 : int, y1 : int, y2 : int) : void
+ setFaceAngle(angle : string) : void
+ getPace() : int
+ getShot() : int
+ getDrib() : int
+ getPas() : int
+ getDef() : int
+ getOverall() : int
+ shoot() : void
+ pass() : void
+ runWithBall() : void
+ dribble() : void

Client objects that query data from persistent data objects will be able to retrieve information fast and easily, just like in a game structure. Data from other classes such as `Layout`, will be generated after triggering the beginning of the game. Also, the lone class that will be closely tied to the graphics is the `GamePlay` class. This is where most of the calls to the OpenGL drawing functions take place.

4.4 Design Constraints, Problems, and Trade-Offs and Solutions

4.4.1 Design Constraints

The most concerning constraint is the time to develop this system. The team believes that some features might need to be dropped to save time and complete the project by the due date. Also, we are limited to a development team of only three people. Other types of constraints might include intellectual resources such as expertise in animating graphics.

4.4.2 Design Problems and Challenges

The sheer amount of computations and moving around of data might render the animation slow at times, especially if the implementation of C++ pointer logic is not optimal, or flawed. Coordinating all the moving parts will be challenging. It appears that all the parts are intertwined, and so when one thing breaks, one will have to inspect several different parts. Also, since we incorporate random numbers, debugging will be much more difficult.

4.4.3 Design Solutions and Trade-Offs

One of the biggest trade offs will be migrating trading familiarity for optimality. Most of of the team is not familiar with the C++ programming language. However, we will employ it to make the animation as smooth as possible.

As far as economic constraints, we are not spending any amount of money for hardware or software, so it is a pretty economical efficient project. SDL and OpenGL are free.

For resource constraints, in terms of the actual syntax needed to implement this, we will be relying solely on free information available online.

Chapter 5 Project Plan and Schedule

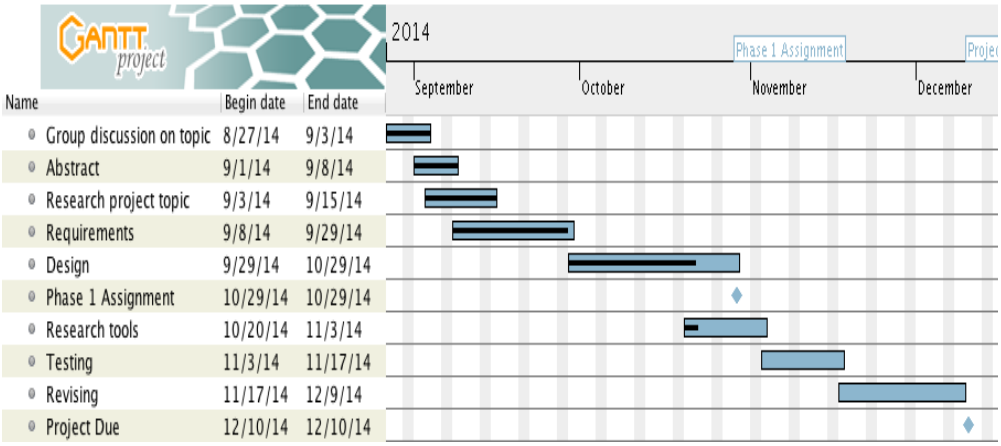
5.1 Project Team

The team members developing this project are Gibran Morales, Carlos Toledo, and Parag Pardeshi. All of us are students of Software Engineering at San Jose State University. The developing on this project has been based on group discussion and advising. We all have knowledge about the soccer industry so we were able to collaborate in all sections of the project. After a group discussion we would put together a draft of the documentation and one of us would be in charge of changing the draft into a document to later have a final revision by all the group members together with the advisor.

Even though the whole project has been done in a collaborative manner through extensive discussions, each one of us had specific focus within each section through the development of the documentation. Carlos focused on chapter 2, the background

and state-of-the-art, and Chapter 5, the scheduling for the project; Parag focused on chapter 3, the requirements for the project; and Gibran focused on Chapter 4, the system design.

5.2 Project Tasks and Schedule



Chapter 6 Tools and Standards

6.1 Tools

We decided to employ Simple Directmedia Layer (SDL) because it has a good reputation for being a game development library. It was designed and written by a group of highly skilled game programmers that really understood what game development was all about. The advantage of SDL is that it provides us with what we need while hiding implementation details. Additionally, it has a proven track record for being robust and free of significant bugs.

SDL will assist us in interpreting data in a more meaningful and aesthetically pleasing way. This cross platform library will provide us with an Application Programmer's

Interface (API) and allow us to use the software/hardware features from our platform (which will be Mac OSX). In this case, the features will be graphics more than anything else. We might need the API for reading keyboard and mouse input as well. For instance, we will need to decide whether we want to start our simulation by clicking a button on a GUI or by simply executing the program. But this is a minor cosmetic detail. Mainly, SDL will be used for its rendering of different colored geometric shapes on a GUI. We will need fast manipulation and deployment of these graphical constructs to mimic the organization and movement of a soccer game. The actions of our graphics will solely depend on the output of our formulas, which in turn depend on the values retrieved from the persistent objects.

On the hardware side of our project, any personal computer with any reasonable amount of RAM will suffice for this simulation.

6.2 Standards

Not much standards are needed for this project, but one that comes to mind is the C programming language-coding standard. Since different members have different coding styles, coming to an agreement in set of rules in programming will be of great assistance in reading each other's code.

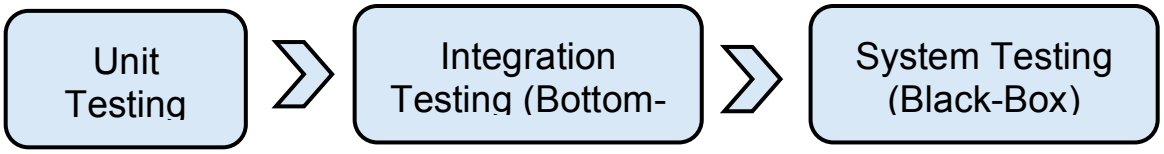
Also, as a supplement to the coding standard and for our convenience, we will follow the documentation standard for C sources (GNU Coding, 2014).

Chapter 7 Testing and Experiment Plan

7.1 Testing and Experiment Scope

The testing of the software will be mainly by units. A unit will be measured as a procedure. In our case, user stories are replaced with what can be most accurately and briefly be described as set of mathematical and logical-decision models. Test cases will be created for each unit (units will be the mathematical models) and testing will occur as code is being generated. Once we can start to put units together into components, we will employ a bottom-up approach to integration testing of the software. In this approach, the lowest level components will be tested first and once integration continues to greater components, the testing will continue with these until the whole system is developed. At the end of the testing process, the whole system will be tested as a whole.

Additionally, the complexity stemming from the need to synchronize various moving parts forced us to perform manual testing on various modules that are tightly coupled with other modules. Examples of these difficult modules include the functions that make the player dribble, functions make a decision while passing, and defending.



For the system testing, different test cases will be created. Black box testing is done while a game is in process. Each of these test cases will be tested against the system through black box testing. Each of these test cases will test general functionality of the system that is expected from the user. No knowledge of the system code will be needed in order to test these features.

ID	Test Objectives	Pre-conditions	Priority	Expected Result	Result
TC_01	Verify that the ball begins rolling as the game starts	The players data is loaded from the database and encapsulated in corresponding objects	High	The ball is passed between players from the same team	Passed
TC_05	Verify that players follow the general direction of the ball	The game has been initiated	High	All the players move in the general direction of the ball	Passed
TC_10	Verify that the ball a player can pass the ball to a teammate successfully	The ball has to be in control of the player	High	The player who receives the ball needs a reasonable amount of time to have full control of the ball	Passed
TC_15	Verify that a player can move with the ball at various paces	The player has to have full control of the ball	High	The player is able to run with the ball at a reasonable pace	Passed
TC_17	Verify players can dribble the ball	Player needs to have full control of the ball	High	A successful dribble consists of a dribbling player running pas an opposing player in close proximity	Passed
TC_20	Verify that a player can obtain ball opposing player	The tackling player is in close proximity of the ball	High	The tackling player gains control of the ball	Passed
TC**_25**	Verify that a player can shoot the ball	The player needs to have full control of the ball and enough time to gather and load a shot	High	The ball is shot with a speed of 25 % higher than that of an average pass	Passed
TC**_30**	Verify a goal can be scored	The ball is in control of attacking player and in close proximity of the goal	High	The ball crosses the goal line	Passed
TC**_35**	Verify a reasonable amount of goals are scored during the game	The game is played in normal time regulation	High	The average should be 3.5 goals per game	Not Passed
TC**_40**	Verify players play to their capacities	A player needs to be engaged in a game play	High	If a superior player confronts an inferior player, the superior player wins the outcome 60 % of	Not Passed

				the time	
TC**_43**	Verify that a long pass (25 yards) can be connected	The passing player must have enough time to locate and load up power to pass the ball for a long distance. Also, the receiving player must be relatively unmarked	Medium	The receiving player receives that ball from the air (instead from the ground as in short passes)	Passed
TC**_47**	Verify that a wall pass can be performed	There must be 2 players from one side and 1 player from the other in close proximity (within a 10 radius)	Medium	The player who starts the play with the ball receives it back.	Passed
TC**_50**	Verify that a cross pass can be performed (this refers to a pass sent to the penalty area from the sides, not form the front side)	A player has to have time to locate a teammate in the penalty area. This requires more time than the standard 'long pass'	Medium	A successful cross pass is one directed to the penalty area from one of the sides.	Not passed

7.2 Testing and Experiment Plan

Our test plan consists of extensive unit testing of discrete and relatively small modules. Since the output of our system is basically one thing, namely one big continuous animation, we need to be certain that the movements of every individual shape are the correct ones. To accomplish desired behavior, we will need to verify that the calculated values are indeed the desired ones so that the graphics move in the correct manner as well.

The testing of our system as a whole is encapsulated by the following unit test cases, which test the behavior of relatively small modules of the simulation. Designing tests required having precise knowledge of the state of some data sets. Therefore, some of

these data sets were narrowed and simplified with the intention of accurately inspecting some results. This testing procedure can be identified as being of the white-box type as we know the internal details of the data sets. A specific example of this could be where we know the exact coordinates of a set of players along with their state and angles with respect to the player who has possession of the ball.

Summary:

Number of Tests: 13

Tests passed: 11

Tests Failed: 1

Not Run: 2

Percentage of Passed: 84 %

Chapter 8 Project Progress and status

Activities:

No.	Activities	Completion (%)
1.	Choose a project topic	100
2.	Improvise on the project topic; shape it and decide	100
3.	Make and discuss a design plan for the project	100
4.	Decide the purpose of the project and the State-of-Art	100
5.	Find a perfect advisor for the project who can help us get more information	100
6.	Define project goals and objective	100
7.	Find the applications of the project and the impact of the project in the real world	100
8.	Make a complete abstract for the project for demonstration and submission	100
9.	Write project requirements for the projects - Domain requirements - System requirements - Non-functional requirements - Context and interface requirements - Technology and interface requirements	100
10.	Decide and define the architecture of the system	100
11.	Define how one part will connect to other, in context of interface and different computer languages	100
12.	Collect and merge all the information collected for submission of project milestone report	100
13.	Work on project plan and development schedule	100
14.	Improve and add the project requirements	100

15.	Decide and define the tools used by the project while implementation	100
16.	Define the standards of the project	100
17.	Explain and decide the level of testing to be done to the final form of implemented project	100
18.	Define the scope for experimentation and testing of the project	100
19.	Write the test cases for the project to test the requirements of the project	100
20.	Complete the Project Documentation phase and submit the document	100

Risk Factors and their mitigation plans:

No.	Risk Factors	Solution
1.	Proper connection of front-end design with back end data.	We will use a proper database connectivity and test it before using
2.	Designing and displaying graphics is tough part in our project	We are learning graphics to implement it in the project
3.	Proper usage of the database to store the data	We will use proper data storage software to store related data
4.	Fast and responsive database	We will create database such that the responsive time is very low
5.	Losing the connection with the database	We will do proper error test on the project
6.	Runtime errors that may mislead the game	We will do proper use case and error testing
7.	Finishing the project implementation on time	We will start working on the project as soon as possible, possibly before the beginning of next semester

References

GNU coding standards. (2014, May 13). *The GNU Coding Standards.* Retrieved November 25, 2014, from http://www.gnu.org/prep/standards/standards.html#index-README-file