UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

DEPARTMENT OF COMPUTER SCIENCE

COS 301 - SOFTWARE ENGINEERING

OnlyRugby Functional Requirements

Author:
Herman Keuris
Johan van Rooyen
Estian Rosslee
Ivan Henning
Muller Potgieter

Student number:
u13037618
u11205131
u12223426
u13008219
u12003672

May 29, 2015

Software Requirements Specification and Technology Neutral Process Design

OnlyRugby Mobile App/Main Project

Version: Version 0.2 Alpha For further references see gitHub. May 29, 2015

Contents

1 Functional requirements

1.1 Introduction

We use this document to give a high level overview of the buzz discussion board. We have identified the various components of our system. The purpose of this document is to create a dynamic and scalable solution. We also want to include an achievement system that rewards users for using the discussion board. This document will inform you on how we will achieve a system that is both scalable and pluggable. We have identified the use cases of the various components of the discussion board and helped expand on them.

1.2 Use case prioritiation

Critical

- Log in/log out
- Load info (from database)
- Game time
- Scoring

Important

- Substitutions
- Discipline

Nice-To-Have

- Line-outs
- Scrums
- Tackles
- Possession
- Turn-overs
- Clean breaks
- Offloads
- Rucks
- Mauls

1.3 Use case/Service contracts

Use Case	Pre Condition	Post Condition	Description
Log in/log out	(Pre conditions)	(Post conditions)	(Description)
Load info	The app has to be connected to the internet in order to load the info from or to the server (and by extension, the database). Specific info can only be loaded when a person is logged in (like profile information). Statistical information being uploaded needs to know and verify where it is being sent to (i.e. to a player's profile or a team's statistics page).	The information should be loaded into the app from the server and statistical match information should be uploaded and stored in the database via the server, in the correct locations where it is meant to go. This is done by verifying the destination and the data being received each time.	This use case provides a method of uploading and downloading data from the database via the server, to and from the app.
Game time	The user must be logged in, the app must be aware that a match is scheduled for play and the game state should be "Not started".	The start and end time of each half of the match along with game pause intervals and reasons should be persisted to the database. The game state should be set to "Finished".	This use case provides an interface for capturing the game time of a rugby match.
Scoring	The app must be aware that a match is currently being played (i.e. scoring can only occur during game time), the app must be aware of which teams are playing and also be aware of which players are currently on the field (i.e. it must know if any player substitutions have taken place).	All scores must be verified by the user and then uploaded to the database where it can be added to team-, player- and league statistics (including points scored, at what time during the match the points were scored, if it was a try, drop kick, etc).	This use case provides a system whereby event-, team- and player statistics can be gathered during a match so that it can be viewed, analysed and compared with at a later stage.

Substitutions	The app must be aware that a match is currently being played (although substitutions can still be allowed at half-time). The app must have a list of players currently on the field as well as players in reserve.	After a substitution is made the on-field team and the list of reserve players must be updated accordingly, the time of the substitution will be logged and any special reasons for the substitution (such as injury) will be noted.	This use case provides a way of logging changes in the on-field team (which is important to know for some other functions like Scoring). This use case also provides additional statistical information about the match such as which players were forced off the field due to injury.
Discipline	(Pre conditions)	(Post conditions)	(Description)
Lineouts	The app must be aware that a match is currently in play (i.e. lineouts can only occur during game time) and the app must be aware of which teams are playing.	This function gathers information on when a lineout occurred during game time, which team was responsible for throwing in the ball, the identity of the player throwing in the ball, whether or not the lineout was successful, if successful which team won the lineout, and a reason if the lineout failed.	This use case provides a way of quickly logging information about lineouts.
Scrums	(Pre conditions)	(Post conditions)	(Description)
Tackles	The app must be aware that a match is currently being played and have a list of all players (both sides) currently on the field to be able to log tackles made between teams.	The tackler's identity should be verified and the statistics added to the relevant player's profile in the database.	This case provides a way to be able to log how well some players can defend by seeing how many successful tackles they have made throughout their career.
Possession	(Pre conditions)	(Post conditions)	(Description)
Turn-overs	(Pre conditions)	(Post conditions)	(Description)
Clean breaks	(Pre conditions)	(Post conditions)	(Description)
Offloads	(Pre conditions)	(Post conditions)	(Description)

Rucks	The app must be aware that a match is currently being played (i.e. rucks can only occur during game time) and the app must be aware of which teams are playing.	This function gathers information on when a ruck occurred during game time, which team was defending in the ruck and which team "won" the ruck (i.e. if possession of the ball changed then the Possession function will also be notified).	This use case provides a way of quickly logging information about rucks.
Mauls	The app must be aware that a match is currently being played which teams are currently playing (individual players are not a necessity).	It should be logged which team won the maul and if the ball was turned over (the other team won the ball) or not.	This use case provides a way to represent how many mauls were present in the match, by logging each time a player tried to defend the ball on the ground.

1.4 Required functionality

- **Log in/log out**

  (Extended description).

- **Load info**

  The Load Info module will be used to transfer information to and from the database, using the server. All destinations are to be verified before attempting to access them and incoming connections to the server need to be verified that they are from a trusted source.

- **Game time**

  This use case will be used to log the start and end time of each half of a match, as well as any intervals during which time was lost (the game was paused) and a reason for this time loss (injury, substitution, referee consultation, replacement of damaged player clothing).

- **Scoring**

  (Extended description).

- **Substitutions**

  (Extended description).

- **Discipline**

  (Extended description).

- **Lineouts**

  This use case will be used to log information on when a lineout occurred, which team was responsible for throwing in the ball, the identity (name or player number) of the player that threw in the ball, whether or not the lineout was successful, if successful which team won the lineout, and if unsuccessful a reason why the lineout failed.

- **Scrums**

  (Extended description).

- **Tackles**

  This use case will be used to log the amount of tackles made, by which team member of which team and when it was made. Knowing who was tackled is not required, since it will not be recorded in their statistics.

- **Possession**

  (Extended description).

- **Turn-overs**

  (Extended description).

- **Clean breaks**

  (Extended description).

- **Offloads**

  (Extended description).

- **Rucks**

  (Extended description).

- **Mauls**

  This use case will be used to log how many mauls occurred throughout the match. It will record how many occurred, when they occurred and who won the outcome of the maul (whether there was a turnover ball or not, or if a penalty was conceded).

- **Achievement/Rewards system**

1.5   Process specification

We want to show various important process specification of our recommendation.

- Log in/ log out

- Load info

- Game time

- Scoring

- Substitutions

- Discipline

1.6   Domain Model