SCO Project

MyChamp

18th January

Thomas Rune Sørensen Casper Hjorth Jensen

CS2011 Int - EASV

Contents

[Introduction 1](#_Toc314561446)

[State of Delivery 1](#_Toc314561447)

[Application Structure 1](#_Toc314561448)

[Implementation Details 2](#_Toc314561449)

[Appendix 2](#_Toc314561450)

# Introduction

In this project, the goal is to create an application that can manage and keep track of the soccer tournament among local schools in the Esbjerg district. It is going to be programmed in the Java language with the program NetBeans, and diagrams are illustrated with Visual Paradigm Community Edition.

# State of Delivery

The database has received a lot of edits over the course of the project. We have added another relation called Tournament in order to store tournament data for more than just one tournament. This relation has two data entities: TournamentName and TournamentID.

We have also made three new data entities in the Team relation. ScoredGoals (int), MatchPoints (int) and TournamentID (int).

However, these new data entities are currently not in use.

The program itself is not complete. It is lacking some fundamental functions, such as being able to create quarter finals, semi finals and finals. It can therefore also not run the Final Ranking option. It also contains a lot of loose ends that we had to discard continuing, either because we couldn’t find out, or because of time constraints.

The parts that *do* work are:

* Menu system
* Group creation
* The ability to add a team
* The ability to update team information
* Group match creation. (It is however not possible to view the matches, as the printer seems to miss some information)

# Application Structure

The program is built up on the standard 3-layer architecture, with Business Logic, Data Access and User Interface layers. Each UI layer class controls its own bit of the menu system, so one class doesn’t get all the submenus to keep track on. There is only one Data Access layer class, as we do not see the necessity to have more than one class to do this task.

We have three Business Entity classes, Match, Team and Group. Each BE class has its own information that we can call upon in our Business Logic Layer classes.

On the diagrams located in the appendix of this report, you can see the hierarchical interconnections between all classes.

# Implementation Details

This time around, the database has been very uncooperative, as it has given us more problems than ever before. Maybe it’s the way we access it, or maybe it’s the way we ask it to receive or give information, but the limited access to the database has hindered our progress in any further programming testing.

We were given the allowance to borrow the coding techniques from another group, as we had no plan how to arrange teams into groups and create a match schedule for them. This process also wasn’t without road bumps, as the way they had structured it was a little different from the way we had set up our structure.

We have had many problems during this project, and some of these problems have cost us many man hours that could have been used to write more code. We especially had problems asking the database for specific options, and not just entire array lists or business entities.

# Appendix

## The sixteen schools consist of the following:

1. Bakkeskolen
2. Bakkevejens skole
3. Boldesager Skole
4. Bryndum Skole
5. Danmarksgades Skole
6. Darum Skole
7. Egekratskolen
8. Fourfeldtskolen
9. Gredstedbro Skole
10. Hjerting Skole
11. Kvaglundskolen
12. Nordre skole
13. Norremarkskolen
14. Praestegaardeskolen
15. Roerkjaer Skole
16. Skads Skole

## Menu Hierarchy:

* 1. **Start new Tournament**

“Tournament Name”

“Team Names”

“Team Captain”

“Contact Email”

“When done listing teams, type x. Warning, after you do this, you can no longer add teams.”

* 1. **View Match Schedule**

Choose Tournament

2.1. Total Match Schedule

2.2. Group Match Schedule

2.3. Team Match Schedule

2.4. Finals Match Schedule

1. **Update/Change Tournament information**
   1. Choose Tournament
      1. Choose which team to update information on

“Team Name: ”

“Team Captain: ”

“Team Email: ”

“Remove team?”

* + 1. Choose which Match to update information on:

(if group matches are still being played)

Which group would you like to update information on:

“group 1”

“group 2”

Which match would you like to change information on:

“team1 vs. team10 - round 1”

“team3 vs. team14 – round 1”

“team1 vs. team 14 – round 1”

“team1 vs. team 3 – round 1”

“team3 vs. team 10 – round 1”

“team10 vs. team 14 – round 1”

“team1 vs. team10 - round 2”

“team3 vs. team14 – round 2”

“team1 vs. team 14 – round 2”

“team1 vs. team 3 – round 2”

“team3 vs. team 10 – round 2”

“team10 vs. team 14 – round 2”

Which team is home team?

1 – “team1”

2 – “team10”

“Home Team <teamName> Goals: ”

“Guest team <teamName> Goals: ”

(if groups have played all their matches)

1 - “team1 vs. team3”

2 - “team4 vs. team2”

3 - “team6 vs. team7”

4 - “team5 vs. team8”

5 - etc…

“Which team is home team?”

1 – “team1”

2 – “team3”

“Home Team <teamName> Goals: ”

“Guest team <teamName> Goals: ”

1. **View Ranking Lists**
   1. View all teams (in random order)
   2. Group rankings

“Group ID”

Example:

|  |  |  |
| --- | --- | --- |
| “Team Name” | Team1 | Team2 |
| “Matches played” | 4 | 3 |
| “Goal Difference” | 4scored-3against=1 | 2scored-5against=-3 |
| “No. of Points” | 4 | 1 |

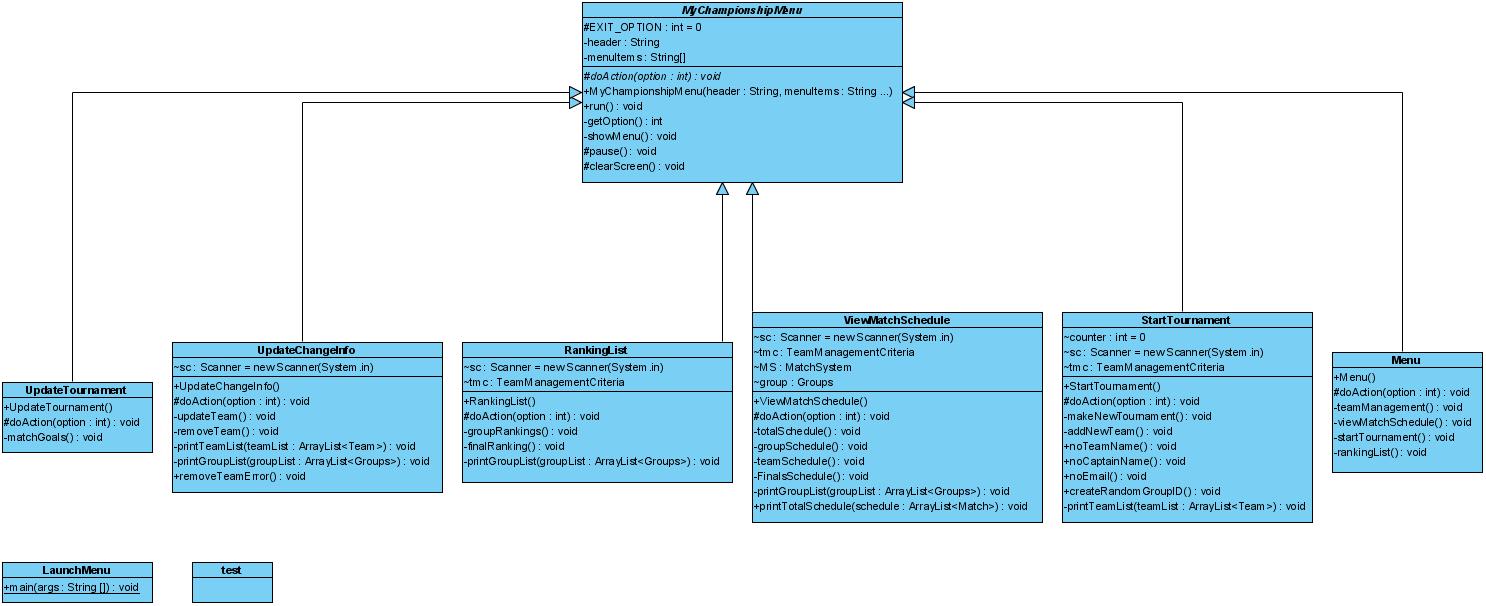
* 1. Final Ranking

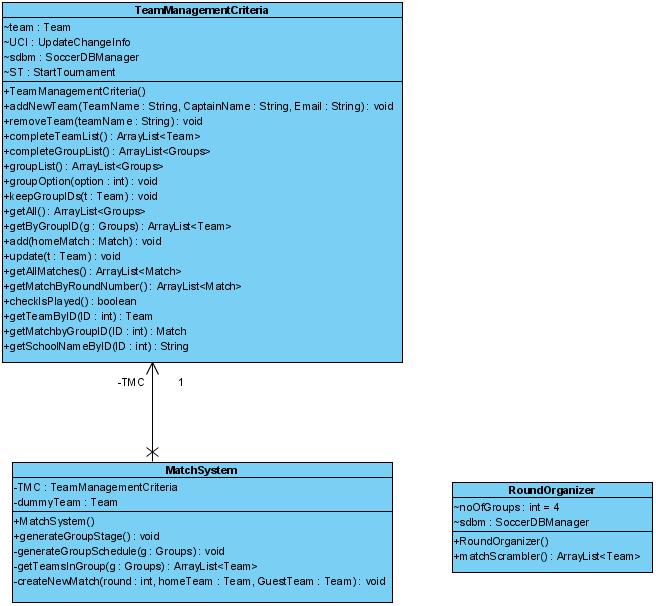
“This option is only available when the tournament is finished and the grand final has been played.”

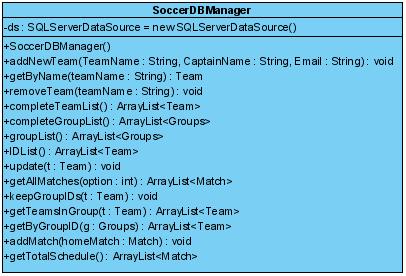
Example:

1. Team5
2. Team3
3. Team8
4. Team9
5. Team14
6. Team12
7. Team13
8. … you get the idea

User Interface Layer:

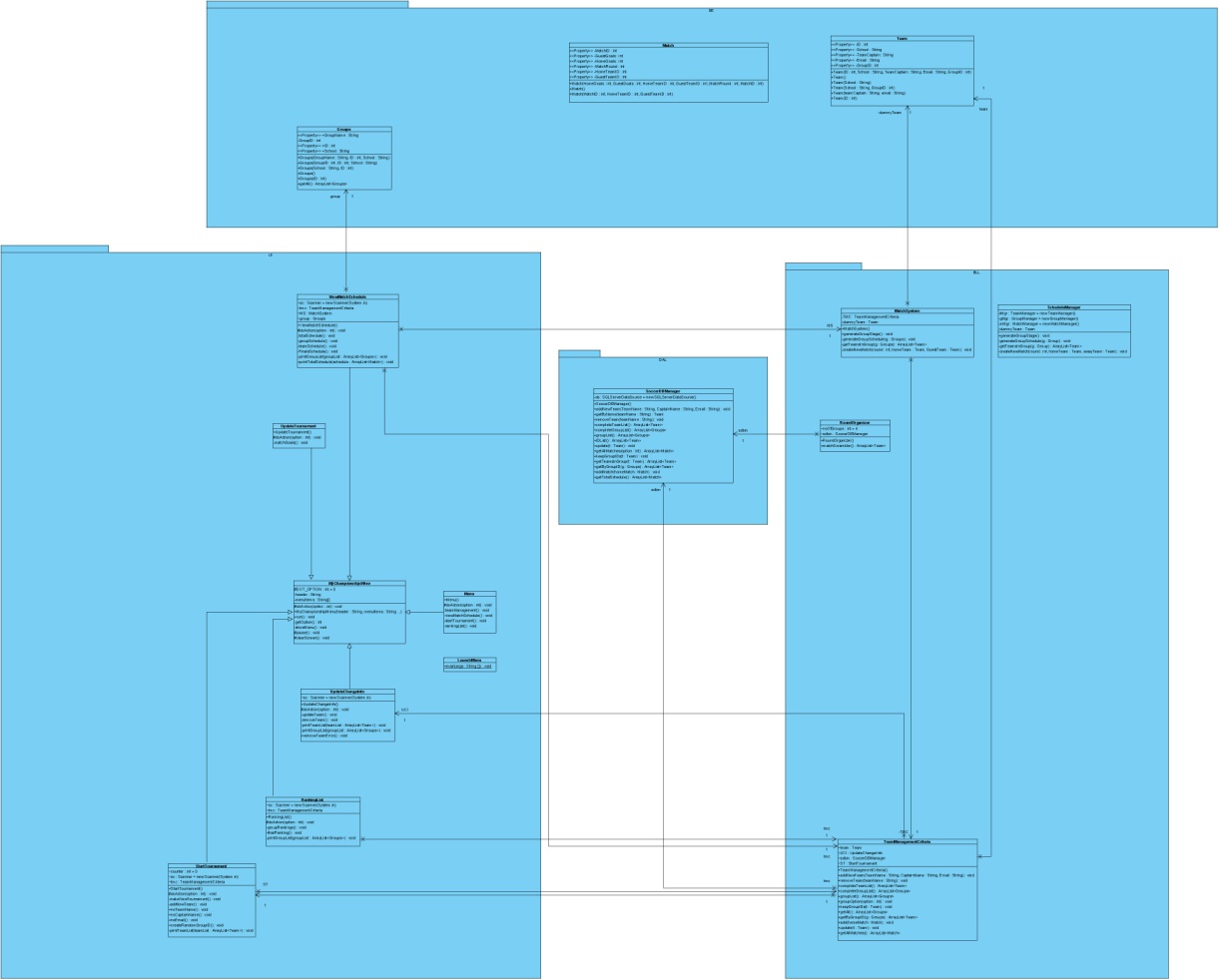


Business Logic Layer:

Data Access Layer:

Business Entities:



Total Diagram:

These images will also be available in a bigger format on the CD with the project.