POLITECNICO
MILANO 1863

Department of Electronics, Information and Bioengineering
Doctoral Programme In Information Technology

Software Engineering 2 Requirements Analysis and Specification Document

Author(s): **Filippo Balzarini -**
Christian Biffi -
Michele Cavicchioli -

Academic Year: 2023-2024

Download Page: `https://github.com/filomba01/BalzariniBiffiCavicchioli`

Contents

1 | Introduction

1.1. Purpose

The CodeKata is a learning method that takes inspiration from the Kata techniques and is based on continuous practice which became very popular in those years.

CodeKataBattle delineates an innovative platform geared towards enhancing students' software development skills through collaborative learning using CodeKata's fundamentals. Facilitated by educators, CKB provides a dynamic environment where students engage in code kata battles, refining their programming proficiency and embracing best practices such as the test-driven development approach.

Similar to recent initiatives addressing global challenges, CKB empowers educators to orchestrate challenges within competition, fostering healthy competition and cultivating an environment for skill enhancement. The platform enables educators to define battle parameters, set deadlines, and configure scoring criteria, fostering a tailored and effective learning experience.

At its core, a code kata battle presents students with programming challenges within specific language frameworks, coupled with exhaustive test cases. Teams collaboratively tackle these exercises, adhering to a test-first methodology and submitting solutions to the platform upon battle completion.

CKB's automated evaluation system ensures an impartial assessment of student submissions. Automated scrutiny covers mandatory factors, including functional aspects, timeliness, and source code quality, offering an unbiased representation of team performance. Educators can further enhance evaluations with optional manual assessments, providing nuanced insights into student work.

1.1.1. Goals

The CKB platform aims to provide a collaborative environment for students to practice and refine their software development skills. The platform enables educators to orches-

trate challenges within competitions, fostering healthy competition and cultivating an environment for skill enhancement. The platform enables educators to define battle parameters, set deadlines, and configure scoring criteria, fostering a tailored and effective learning experience.

The platform will be used by two types of users: Educators (ED) and Students (ST). The ED will be able to create competitions and battles within competitions. The ST will be able to create teams and join battles as a team or individually. The platform will provide a dashboard for code submission and automated evaluation of the code submitted. The platform will also provide a ranking of the competition and battles.

Below there is the table of goals that the platform will achieve:

#	**Goal**
G1	Enable ED to MANAGE Competitions
G2	Enable ED to manage Code Battles within Competitions
G3	Enable ST to participate in a Competition
G4	Enable ST to be part of a team within a battle
G5	Send Notifications to STs
G6	Automatically Create GitHub Repositories for Every Battle in a Competition
G7	Synchronize the Submission of Each Candidate with Their GitHub Repository
G8	CKB Provides an evaluation of the code submitted
G9	Allow users to View Rankings in both battles and competitions
G10	Allow to assign badges to the students

Table 1.1: List of goals

1.2. Scope

1.2.1. World phenomena

ID	Definitions
WP1	ED wants to create a competitions
WP2	ED wants to create a battle
WP3	ST wants to participate in a competition
WP4	ST wants to participate in a battle
WP5	ST set up GitHub actions

Table 1.2: List of the world phenomena

1.2.2. Shared phenomena

ID	Definitions
SP1	ST creates an account on the platform
SP2	ED creates an account in the platform
SP3	ST logs in to the platform
SP4	ED logs in to the platform
SP5	ST registers for the competitions before the deadline
SP6	ED creates a badge with certain rules
SP7	ED manually evaluates the code submitted by students
SP8	ED creates a competition
SP9	ED creates a battle within a competition
SP10	ED closes a competition
SP11	ST pushes new commit(s) into their GitHub repository before the deadline
SP12	ST invites other STs to participate in a battle as a team
SP13	ST subscribes as a single/team for an incoming battle before the deadline
SP14	CKB sends a notification that a competition is available to ST
SP15	CKB sends a notification that a battle is created inside a competition to ST
SP16	CKB sends a notification that a competition has ended to ST
SP17	CKB sends a notification that a battle has ended to ST
SP18	CKB sends links to the GitHub repository to all the ST subscribed
SP19	CKB updates scores for each ST
SP20	CKB gives badge to ST
SP21	CKB updates the ranking of the competition
SP22	CKB updates the ranking of the battle

Table 1.3: List of the shared phenomena

1.3. Definitions, Acronyms, Abbreviations

1.3.1. Definitions

User	anyone interacting with the system, it can be both a Student or an Educator
Manage	create, supervise and edit a certain element of the application.

Table 1.4: List of

1.3.2. Acronyms

ST	Student
ED	Educator
CKB	CodaKataBattle
RASD	Requirements Analysis and Specification Document
SAT	Static Analyzer Tool
T	Team

Table 1.5: List of Acronyms

1.3.3. Abbreviations

WPX	World Phenomena X
SPX	Shared Phenomena X
GX	Goal Number X
DX	Domain Assumption X
UCX	Use Case X

Table 1.6: List of abbreviations

1.4. Revision history

1.5. Reference Documents

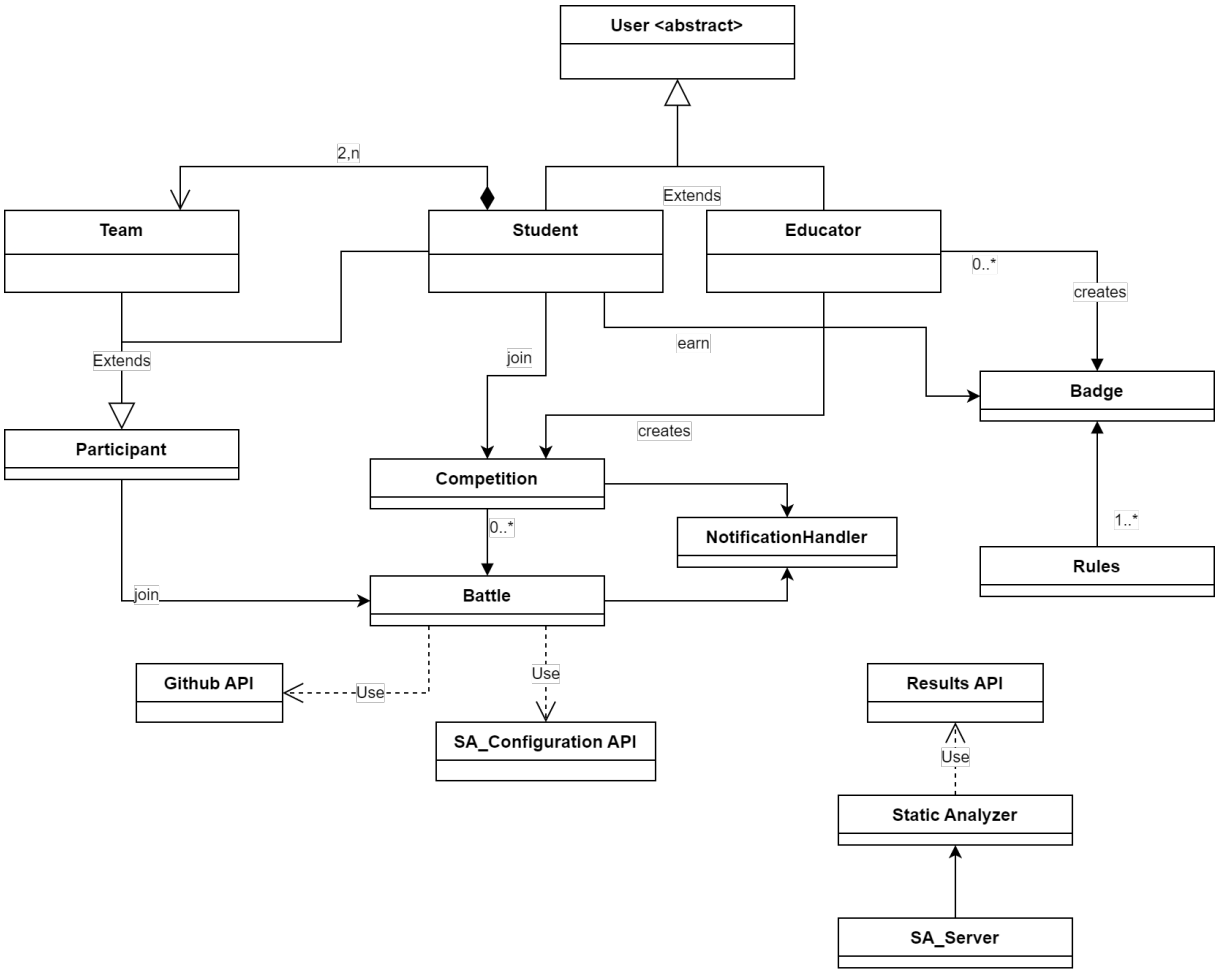
1.6. Document Structure

2 | Overall Description

2.1. Product perspective

2.1.1. Class Diagram

Here we provide the class diagram of our system and a brief description on some of the key aspects. The very first thing to explain is its logical split of the *Static Analyzer* from the rest of the system, the reason for this is that we wanted to have a system that could be more *flexible* than usual. The two logical parts can be implemented in the same tier or can also be split physically in terms of devices. What does not change is that when the STs set up github actions on their repository, the server that will be called when a commit is pushed is the ***SA_Server***, which will do the necessary checks and then sends the results to the main server (***Results API*** has this purpose), where the score will be computed. Note that the Static Analyzer can be configured from the main server by using the provided APIs in ***SA_Configuration API*** (e.g., upload the test cases...). As asked by the assignment we distinguished two types of *Users*, ***Students*** and ***Educators***, in order to differentiate the features the platform offers to the two. The same concept is also applied to the participants of a battle, which does not distinguish a team from a single student; this is very useful when the battle finishes and the results have to be translated into the competition. The ***NotificationHandler*** has the purpose of handling the notifications of the users and it provides the procedures to do just so.



2.1.2. State Diagrams

2.1.3. Scenarios

Scenario 1: Professor Harry is a professor teaching at Politecnico di Milano together with professor Donald. Harry would like to encourage his students to study during the course, instead of having to study everything a few days before the exam. To do so he came up with the idea to create a challenge where the students can test their preparation and earn some extra points in the exam. While talking with other colleagues, Prof. Harry discovered CKB and he thought it was the perfect fit to implement his idea. The first thing he does is to go to the webpage of CKB and create an account by clicking the ***Sign up*** button and providing some information about himself. Afterwards he is redirected to the home page of the platform where he can click the button ***Create Competition***, and finally he inserts the name of the competition and the subscription deadline. At this point he wants to invite his colleague Donald to manage the competition with him; since he is an ED in the competition he can click on ***Invite Educator*** in the competition page,

then provides the email of Donald's account, who will be part of the competition once he accepts the invite.

Scenario 2: Professor Harry is an ED of a competition, within which he wants to create a battle. To do so he enters the dashboard of the competition, clicks on the button ***Create Battle*** and provides everything the platform needs: description, test cases, build automation scripts, deadlines, accepted sizes of groups. Marco, a ST who subscribed to the competition, received the notitication about the newly created battle via email. Outside of the platform Marco agreed with a couple of friends to participate in the battle together. Marco then goes on the competition page and finds the newly created battle, here he finds two buttons, ***Participate as: 1. Loner 2. Team***; he clicks the second button to participate as a team and invites his friends by providing the platform his friends' account email. Once Marco's friends accepted the invite the subscription to the battle will be automatically finalized by the platform.

Scenario 3: Professor Harry wants to give credit to the hardest working student, so while creating the competition he decided to create a new badge. The hardest working ST is the one that has written the highest amount of code lines among all the battles in the same competition. To implement this badge Prof. Harry must create a new variable ***hardest_worker*** and provide the code that defines how to compute the value of such variable. Some time after the specification of this new badge, a ST, participating in the competition and in the current battle, called Marco, pushes a commit to his repository. Since all students are supposed to setup *GitHub Actions*, CKB is notified about Marco's commit, so it proceeds to run the required processes to calculate the new score, but also checks if Marco acquired new badges by checking their rules. Assuming that with the last commit Marco has now the most written lines of code, CKB assigns to him the *hardest_worker* badge.

Scenario 4: Marco and his team participated in a battle provided by Prof. Harry, one of the ED of the competition. Since the battle ends the next day, Marco wants to look at the partial rankings of the battle, so he goes on the page related to the battle and clicks on the *Results* section, and sees his team at the bottom of the chart. Understandably, Marco's team resumes to work on the problem and they are able to commit a new version of their solution, which increased their placement in the partial rankings of the battle. The submission deadline now expired and the EDs now want to manually check the work of their STs to assign manually a score to each team; to do this Prof. Harry goes on the battle page and clicks on ***Perform Manual check***, which will redirect the ED to another page where he can inspect the source code of each team and give a score to each final work. Once this consolidation phase has been declared finished by an ED, CKB

sends to all the STs subscribed to the competition a notification that the battle's results are available and the global scores of the competition have been updated.

2.2. Product functions

The ED can create a new competition
The CKB allows the ED to create a new competition by clicking on the ***Create Competition*** button on the home page, then providing the name of the competition and the subscription deadline.

The ED inside of the competition have the possibility to create a new battle by clicking on the ***Create Battle*** button, where he can then insert the description, test cases and the solution. He can also set which aspects of the code he wants that CKB evaluate, such as the quality of the code, the number of lines of code, the security, the readability and the maintainability. Moreover he can select the minimum and maximum number of team components and the deadline for the submission.

Inside of each battle the ED can add badges by clicking on the ***Add Badge*** button, where he can then insert the name of the badge, the description and the rules to assign the badge.

The ED can also invite other EDs to manage the competition with him by clicking on the ***Invite Educator*** button and providing the email of the account of the ED he wants to invite. This will allow the colleague to change the settings of the competition and create new battles.

The ST can participate in a competition
The ST after logging in the platform can see the list of the competitions available, and can subscribe to them by clicking on the ***Subscribe*** button. When the student clicks on the button he can choose to participate as a loner or as a team, in the latter case he has two possibilities:

- Create a new team by clicking on the ***Create Team*** button, where he can then insert the name of the team and the email of the other STs he wants to invite. Once the other STs accepted the invite the subscription to the competition will be automatically finalized by the platform.
- Join an existing team by clicking on the ***Join Team*** button. Once the team leader accepted the request,the ST will be added to the team.

After subscribing to a competition, the ST can see the list of the battles inside the competition and can subscribe to them by clicking on the ***Subscribe*** button. When the student clicks on the button he can choose to participate as a loner or as a team, in the latter case he can invite other STs to participate in the battle with him by sending them an invite via email.

Inside the competition ST can also see the general ranking of the competition, which is updated after each battle. It is also possible to see the partial ranking of each battle he have partecipated in, which is updated after each submission.

Another feature available to the ST is the possibility to see the list of the badges he earned, and the list of the badges he can earn with the corresponding rules.

The CKB can evaluate the submissions

The CKB is able to automatically evaluate the submissions of the STs by running the test cases provided by the EDs. Each new code submission made on the Github repository of the STs is notified to the CKB, which then runs the test cases on the code.

The test cases are run in a sandbox environment to prevent malicious code from damaging the system. The CKB is also able to run the build automation scripts provided by the EDs to check if the code compiles and if it satisfies the requirements.

The code is evaluated also considering the quality of the sources by running static analysis tools on the code that considers the complexity of the code, the readability and the maintainability.

The CKB after performing all the checks assigns a score to the submission and updates the ranking of the battle and the competition. It also checks if the STs earned new badges by checking their rules and, in case, assigns them to the STs.

It is also possible for the ED managing the battle to manually evaluate the submissions by clicking on the ***Perform Manual Check*** button on the battle page. This will redirect the ED to another page where he can inspect the source code of each team and give a score to each final work. Once this consolidation phase has been declared finished by an ED, CKB sends to all the STs subscribed to the competition a notification that the battle's results are available and the global scores of the competition have been updated.

2.3. User characteristics

The actors that are going to use the CKB system are:

- **Educator (ED)**: an educator is a user that can create competitions and battles within competitions. He can set the parameters of the battles and the deadlines. He can also invite other educators to manage the competition with him.
- **Student (ST)**: a student is a user that can create teams and join battles as a team or individually. He can earn points and badges by partecipating in the competitions and battles.

2.4. Assumptions, dependencies and constraints

ID	Description
DA1	ST owns a device able to connect to the internet
DA2	ST owns a GitHub account
DA3	ST has installed Git on his computer
DA4	ST knows how to use Git
DA5	ED knows how to use Git
DA6	ED owns a device able to connect to the internet
DA7	ED writes correct tests
DA8	ED correctly evaluates the final source code of a T
DA9	GitHub permits automatic push to a repository
DA10	GitHub permits automatically pull from a repository
DA11	ST knows the usernames of other STs they want to invite to a T
DA12	STs has an internet connection
DA13	ED has an internet connection
DA14	ST writes code only with languages that are treatable by the platform
DA15	ED knows the email of the other EDs he wants to invite to manage a competition
DA16	ED writes the correct badge' rules

Table 2.1: List of the domain assumption

3 | Specific Requirements

3.1. External Interface Requirements

3.1.1. User Interfaces

The CKB user interface will be a web page that will be accessed through a web browser. The web page will be designed to be simple and easy to use with the support for multiple screen sizes and devices.

3.1.2. Hardware Interfaces

The platform requires a computer with a web browser and an internet connection to access the CKB web page.

3.1.3. Software Interfaces

CKB will be using some external interfaces in order to provide the service. The external interfaces are listed below:

- **Github API:** CKB will use Github as source control system for the projects. The Github API will be used to create the repositories for each team in the battle and share it with the team members.
- **Static Analyzer API:** CKB will use a static analysis tool to analyze the code of each team after every new commit and will use the result of the analysis to assign points to each team. The purpose of this interface is to give the possibility to the system to configure the analyzer as the educator needs, in terms of test cases, languages supported and other parameters.
- **Results API:** The *SA_Server* will use this API to send the results of the analysis to the main server.
- **Github Actions:** Github Actions will notify the system when a participant has

pushed a new commit to its repository. The *SA_Server* will be listening to this notifications and will trigger the analysis of the source code.

3.1.4. Communication Interfaces

All the communication between CKB, the external interfaces and the user will be done using HTTPS protocol.

3.2. Functional Requirements

3.3. Performance Requirements

3.4. Design Constraints

4 | Formal Analysis Using Alloy

Organize this section according to the rules defined in the project description.

5 | Effort Spent

Members of group	**Effort spent (hours)**	
Filippo Balzarini	Introduction	1*h*
	Overall description	0*h*
	Specific requirements	4*h*
	Formal analysis	0*h*
	Reasoning	3*h*
Christian Biffi	Introduction	3*h*
	Overall description	3*h*
	Specific requirements	0*h*
	Formal analysis	0*h*
	Reasoning	4*h*
Michele Cavicchioli	Introduction	1*h*
	Overall description	2*h*
	Specific requirements	2*h*
	Formal analysis	0*h*
	Reasoning	2*h*

Table 5.1: Effort spent by each member of the group

Bibliography

List of Figures

List of Tables

List of Symbols

Variable	Description	SI unit
$\boldsymbol{u}$	solid displacement	m
$\boldsymbol{u}_f$	fluid displacement	m