### **TEAM MANAGEMENT SYSTEM IN THE CHAMPIONS LEAGUE**

**PHASE 1: IDENTIFICATION OF THE PROBLEM**

**Identification of Needs and Symptoms**

* **Real needs:** The UEFA organizers require an automated solution to manage the teams, matches, and rankings of the Champions League efficiently.
* **Conditions:** The system must allow for team and match registration, manage rankings efficiently, and have the capability to undo errors in actions quickly and easily.
* **Symptoms:** The lack of an automated and efficient system to handle the vast amount of data generated by each match, team rankings, and real-time updates.

**Problem Definition**

UEFA needs a software system to manage the teams, matches played, and rankings of teams in the Champions League, ensuring that the information is processed efficiently, accurately, and is easily accessible.

**SPECIFICATION OF REQUIREMENTS**

**FUNCTIONAL REQUIREMENTS**

1. **Team Registration:**

* The system must allow the registration of teams with attributes such as: name, country, number of titles won, and UEFA coefficient.
* Teams must be stored in a hash table for efficient searching.

1. **Match Registration:**

* The system must allow the registration of matches between two teams, including goals, home and away teams, and match date.
* Matches must be stored in a queue to guarantee processing in the order they were registered (FIFO).

1. **Ranking Management:**

* The system must update the team rankings in real-time based on match results and UEFA coefficient.
* Rankings should be managed using a priority queue, where teams are sorted by points and their UEFA coefficient.

1. **Undo Action:**

* The system must allow undoing the last performed action, whether it’s adding a team, updating rankings, or registering a match.
* Actions must be stored in a stack to enable efficient undo operations.

1. **User Interface:**

* The system must provide a simple menu that allows users to:
  + View registered teams and matches.
  + Check the updated rankings.
  + Perform undo operations.

**NON-FUNCTIONAL REQUIREMENTS**

1. **Efficiency:**

* The system must guarantee fast response times, especially in querying and updating the data for teams and matches.
* Search, insert, and update operations should be performed optimally using data structures like hash tables and priority queues.

1. **Scalability:**

* The system must be capable of managing a growing number of teams, matches, and rankings without significant performance degradation.

1. **Reliability:**

* The system must be reliable with minimal failures. In case of errors, it should allow users to undo the last action performed.

1. **Security:**

* The system must ensure that only authorized users can access and modify the team and match records.
* User authentication should be implemented for data management operations.

1. **Maintainability:**

* The system should be designed to facilitate long-term updates and maintenance, with a modular architecture and proper documentation.

**PROCESS REQUIREMENTS**

1. **Team and Match Registration:** The registration process should be intuitive, and data for each team and match should be validated before being stored.
2. **Ranking Update:** The ranking update must happen automatically after each match is registered. Points and coefficients for teams should be recalculated and the ranking reordered in real-time.
3. **Undo Actions:** The system must allow the last user action to be undone. When an action is undone, the data should be immediately updated without compromising the integrity of the system.
4. **User Interface (UI):** Users should have access to all main functionalities through an easy-to-use interface, allowing them to manage teams, register matches, view rankings, and undo actions.

**COLLECTION OF NECESSARY INFORMATION**

**SOURCES OF INFORMATION**

1. UEFA Regulations and Rules:
   * Data on teams: Requirements for team registration (such as titles won, country, UEFA coefficient).
   * Data on matches: Match results, teams involved, match dates, and goals scored.
   * Ranking criteria: How the ranking system works (points, coefficients, etc.).
2. Data Structures:
   * Hash Tables: For storing and quickly accessing information about teams and matches.
   * Stacks: To track actions that can be undone (team registration, match results).
   * Queues: To manage the order of matches (FIFO processing).
   * Priority Queues: To maintain the ranking of teams based on points and UEFA coefficient.
3. Software Requirements:
   * Programming Language: Java (based on the task requirements).
   * Database or File System: If applicable, to store persistent data.
   * Libraries: Use of standard libraries for implementing hash tables, queues, and stacks.

**Key Elements for the System**

1. Teams:
   * Attributes: Team name, country, titles, and UEFA coefficient.
   * Data Storage: Hash table for fast lookup by team name.
   * Functionality: Ability to add, update, and remove teams efficiently.
2. Matches:
   * Attributes: Home team, away team, goals scored by each team, and match date.
   * Data Storage: Queue for FIFO processing of match registration.
   * Functionality: Ability to record match results and process matches in the order they were added.
3. Ranking:
   * Attributes: Points and UEFA coefficient for each team.
   * Data Storage: Priority queue to keep teams ordered by performance.
   * Functionality: Update rankings after each match and sort teams by points and UEFA coefficient.
4. Undo Actions:
   * Attributes: A record of all actions taken (team addition, match registration).
   * Data Storage: Stack to manage the undo operation.
   * Functionality: Ability to revert the last action performed by the user.

**Additional Information**

* External Research:
  + Looking at similar systems (e.g., other sports management systems) to understand the best practices.
  + Reviewing how rankings are typically calculated in other football leagues to ensure accuracy.
* Stakeholder Input:
  + Communication with stakeholders (like the UEFA organizers) to gather any additional data they might need for the system.

**SEARCH FOR CREATIVE SOLUTIONS**

**Brainstorming**

In this phase, we will generate potential solutions to solve the problem of managing teams, matches, and rankings in the UEFA Champions League. Here are some possible ideas:

1. Modular System Architecture:
   * Modules: Create independent modules for each function: Team Registration, Match Registration, Ranking Management, and Undo Function.
   * Each module would handle a specific part of the system (team data, match data, ranking data, etc.), making the system easier to maintain and extend.
2. User Interface:
   * Create a CLI (Command Line Interface) for easy interaction.
   * Alternatively, we could design a Graphical User Interface (GUI) using Java Swing for a more user-friendly experience.
3. Data Structures:
   * Hash Table for storing team information efficiently (quick access to teams).
   * Queue for storing matches in the order they were registered (FIFO).
   * Priority Queue (Max-Heap) for maintaining the ranking of teams based on points and UEFA coefficients.
   * Stack for undoing the last action performed, whether it’s adding a team or recording a match result.
4. Automated Ranking Update:
   * Each time a match is registered, the ranking should be automatically updated based on the results (goals, points, and UEFA coefficient).
   * The ranking will be dynamically adjusted to reflect the latest match results.
5. Undo System:
   * Allow users to revert the last action (team registration, match result entry, etc.) by using the stack data structure.
   * This can be useful in case of mistakes or incorrect data entry.
6. Data Validation and Error Handling:
   * Implement checks to ensure that data entered (team information, match scores, etc.) is valid.
   * If there’s any invalid data, the system should give an error message and prevent the action from being finalized until corrected.

**Review Lists**

* Team Data: Include the team’s name, country, number of titles, and UEFA coefficient.
* Match Data: Home and away teams, goals scored, and match date.
* Ranking: Points, UEFA coefficient, and team position.
* Undo History: Actions that can be undone, such as adding/removing teams, updating match results, or recalculating rankings.

**List of Attributes**

We can list the attributes of each module:

* Team Module:
  + Name
  + Country
  + Titles won
  + UEFA coefficient
  + Points
* Match Module:
  + Home team
  + Away team
  + Goals scored by both teams
  + Match date
* Ranking Module:
  + Points
  + UEFA coefficient
  + Current rank position

**Forced Relationship (Integration of Modules):**

To push creativity further, we can force a relationship between two seemingly unrelated components to spark innovative solutions. For example:

* **Team Registration + Match Registration**: Each match involves two teams, so every time a match is registered, the teams’ details must be checked or updated (e.g., updating the number of games played).
* **Undo System + Ranking Update**: When a team or match is undone, the rankings should be recalculated automatically based on the previous state.

SOLUTION ALTERNATIVES