System Choice

FACTOR

A FACTOR criterion consists of six elements; functionality, application domain, conditions, technology, objects, and responsibility:

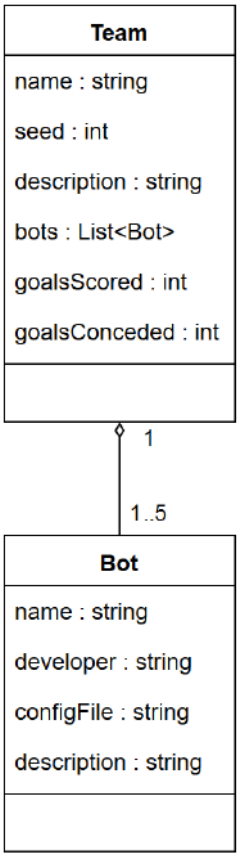
- **Functionality**: The system is able to create and manage tournaments for the RLBot-community. It can automatically start matches. Teams, match results, and stages can be edited at any given time. The system is able to save and load tournaments.
- **Application Domain**: The system is used by members of the RLBot-community to run tournaments that are live-streamed.
- **Conditions**: The system can be used during a live-streamed tournament and while Rocket League is running. The system is used by experienced and novice hosts of RLBot tournaments.
- **Technology**: The system is written in Java and runs on a Windows computer, which is able to run an instance of Rocket League. Furthermore, the machine has sufficient resource-overhead for live-streaming purposes.
- **Objects**: The most important objects in the problem domain are bot, team, match, and stage.
- **Responsibility**: The system is responsible for generating and scheduling matches. The user is responsible for inputting the correct match results and restore the tournament state after a critical edit.

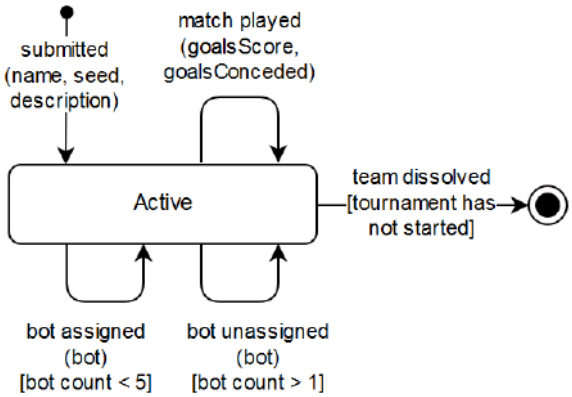
Problem Domain Analysis

Classes

This problem domain contains the following classes: **Tournament**, **Stage**, **Format**, **Match**, **Team**, and **Bot**. Following is the description of each class and its behaviour.

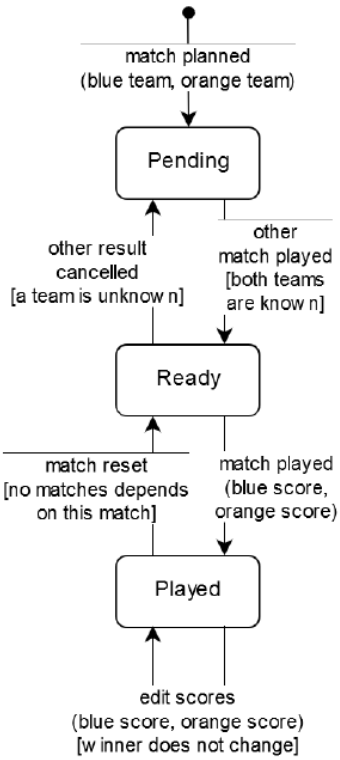
Bot and team



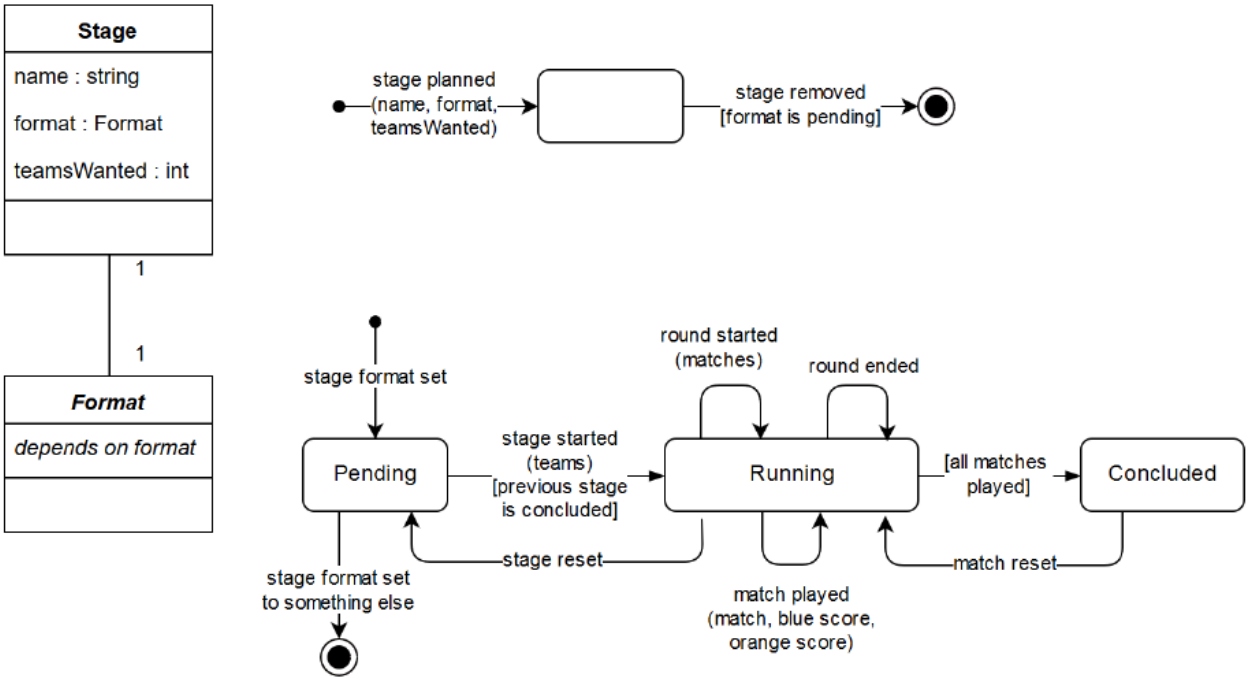


Match

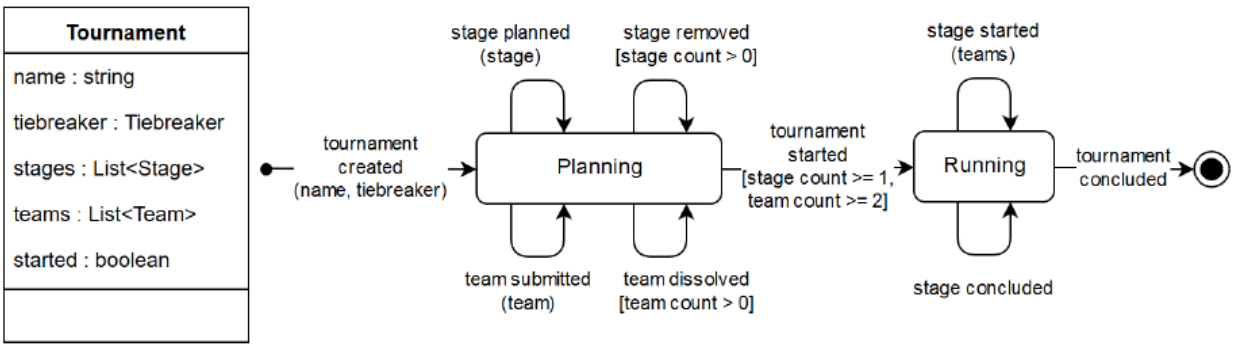
Match
blueTeam : Team
orange team : Team
blueScore : int
orangeScore : int
blueFromMatch : Match
orangeFromMatch : Match
blueWasWinnerInPreviousMatch : boolean
orangeWasWinnerInPreviousMatch : boolean
winnerDestination : Match
loserDestination : Match
winnerGoesToBlue : boolean
loserGoesToBlue : boolean



Stage and Format



Tournament



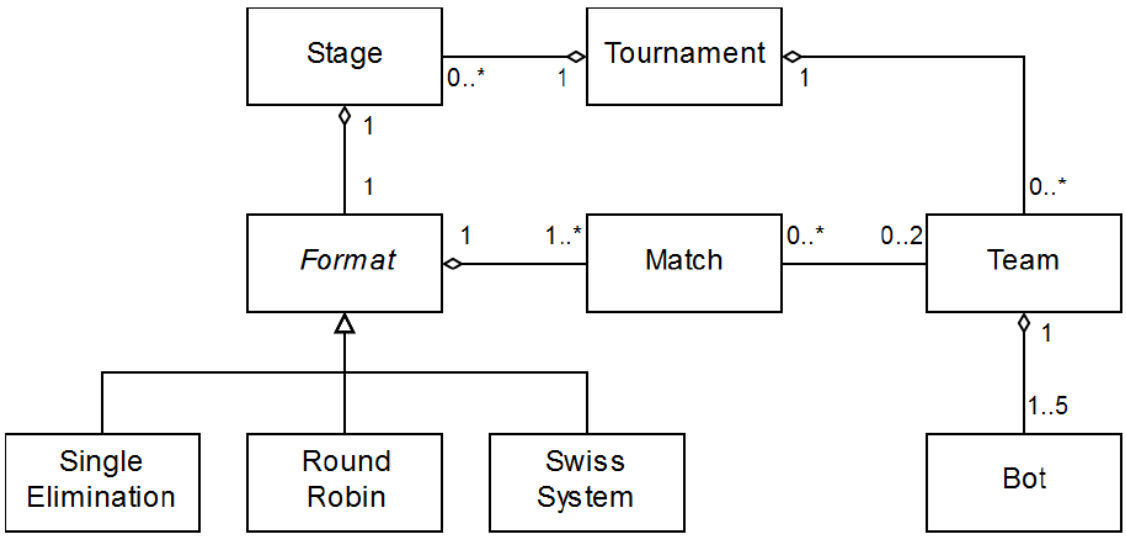
Events

All of the classes presented have behaviour, these behaviours are described with events which can be shown with an event-table. An event is more specifically an atomic, singular event that can occur in the problem-domain. The events are written in past-tense.

Events	Tournament	Stage	Match	Team
match played		*	*	*
match reset		*	*	*
stage planned	*	+		
stage removed	*	+		
stage started	*	*		
stage concluded	*	*		
stage reset	*	*		
team submitted	*			+
team dissolved	*			+
round started		*	+	
round ended		*		
tournament created	+			
tournament started	+			
tournament concluded	+			

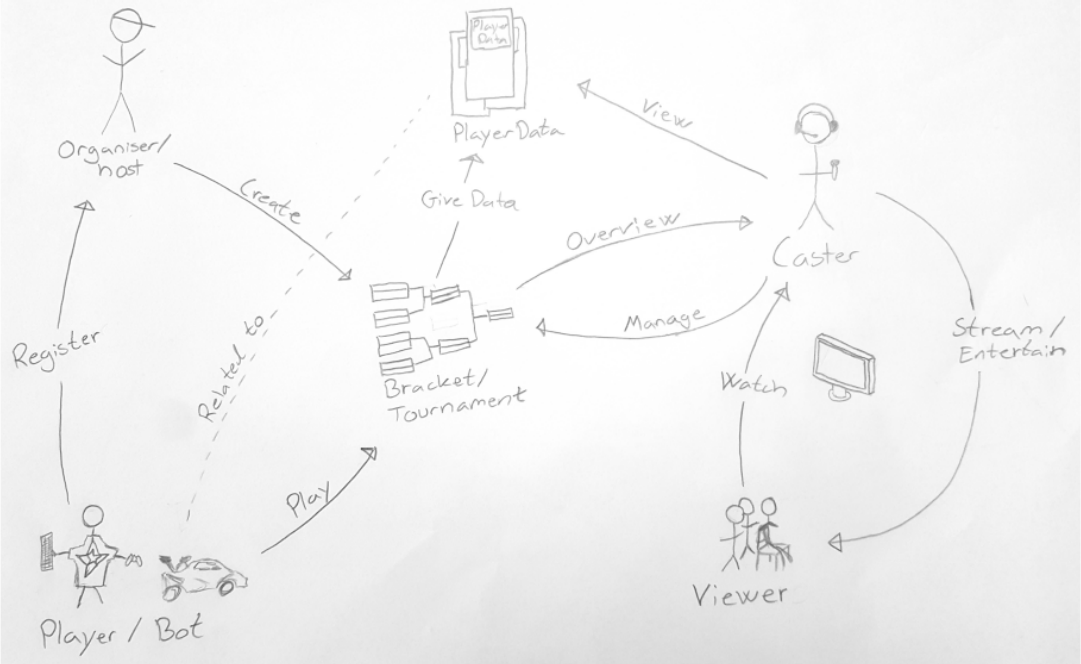
The table shows a multitude of events that happen in the problem-domain, along with their relation to the classes that are noted with either a "*", for multiple occurrences, or "+", for a single occurrence.

Structure



Application-domain Analysis

Rich Picture



Design

Design criteria

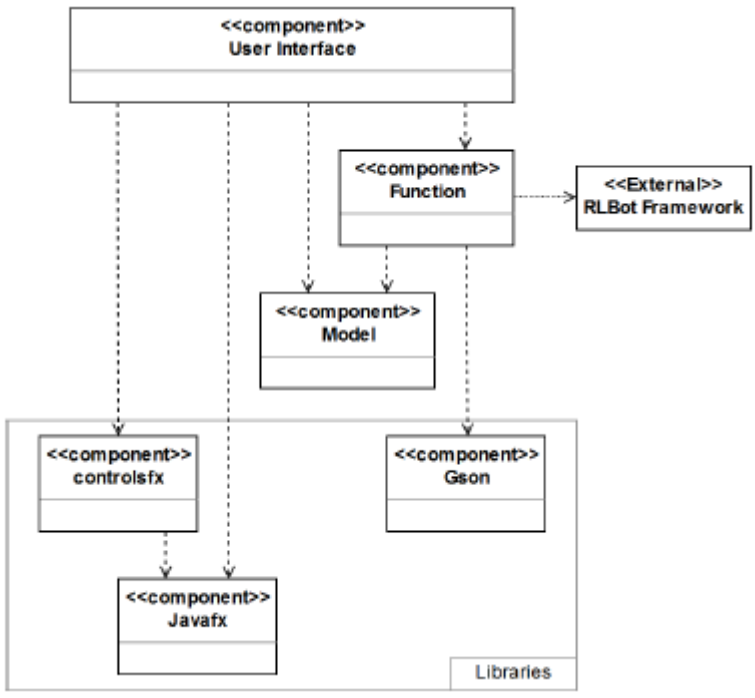
Criterion	Very Important	Important	Less Important	Irrelevant	Easily Fulfilled
Usable	X				
Secure				X	
Efficient			X		
Correct			X		
Reliable		X			
Maintainable		X			
Testable			X		
Flexible	X				
Comprehensible		X			
Reusable				X	
Portable				X	
Interoperable		X			

TEMP

Functions

	Name	Type	Complexity
F25	Add Stage	Update	Medium
F26	Edit Stage	Update	Simple
F27	Create Tournament	Update	Complex
F28	Set Seeding	Update	Medium
F29	Generate Single Elimination bracket	Update	Complex
F30	Set Game Overlay	Update	Simple
F31	Generate Double Elimination bracket	Update	Complex
F32	Generate Swiss round	Update	Complex
F33	Generate Round Robin matches	Update	Complex
F34	Edit Match Result	Update	Simple
F35	Create Match	Update	Simple
F36	Start a match manually	Update	Complex
F37	End Match	Update	Simple
F38	Close RLBot	Update	Simple
F39	Start Match Automatically	Update	Simple
F40	Rank bot based on result	Update	Medium
F41	Load Tournament	Update	Very Complex
F42	Generate configuration file	Update	Medium
F43	Disqualify Team	Update	Simple
F44	Run RLBot framework	Compute	Simple
F45	Update configuration file	Compute	Medium
F46	Save Tournament	Compute	Very Complex

Components



Usage

Work Tasks:

- Register tournament/configure tournament stages: The host decides to register and set-up a tournament, in accordance to the wanted tournament-structure.

- Register bot: The host decides to register a bot for the tournament. The host must enter the bots name, its path to its config, its elo, and optionally a description. Values can be entered in any order, and when done, the task i complete. The task can also be aborted instead of entering a value.
- Edit a played match: The streamer decides to go back a edit the results of a played match. The streamer will select a match to edit, and enter the desired results. The streamer ends the task at any time, either saving the values or abouting without save.
- Start match: The streamer decides to start the next match. First they selects the match they want to start, then starts it. This runs the RLBot framework with the bots in the selected match and creates an overlay with the bots' name.
- Conclude match: The host should be able to end a specific match and fetch the results if wished. First they select a match they want to conclude, then the system will ask if the fetched result should be assigned to this match, if no is chosen the host should be prompted to enter a result manually.