System definition 2

Functionality

The system is able to create and manage tournaments for the RLBot community. It can automatically start matches and fetch results. ***A user can edit match-results.*** It is able to save and load tournament-states, along with import and export of player data. It allows support for an in-game overlay.

Application Domain

The system is used by members of the RLBot community to run tournaments that are live-streamed. They will also use it to record match history of players.

Conditions

The system can be used during a live-streamed tournament and while Rocket League is running. The system will be used by experienced, and novice, hosts of RLBot tournaments.

Technology

The system will be written in Java and will run 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 player, team, match, and stage.

Responsibility

The system is responsible for generating and schedule matches, store tournament-data.
The user is responsible for checking if match result are correct.

Classes

- Player
- Match
- Team
- Tournament
- Stage
- Bridge ???

Not used

User	- not a part of problem domain
Host	- not a part of problem domain
Organizer	- not relevant for problem domain

- Rank - just their position in a collection sorted by score
- Developer - relevant, but just a field in bot profiles
- Streamer - not a part of problem domain
- Viewer - not a part of problem domain
- Bot - same as player
- Human player - same as player
- Player datafile - same as profile
- Match history - just a collection of matches
- Final - the name of a stage
- Callback message - not a part of problem domain
- Plugin - same as bridge
- Profile - same as player
- Bracket - same as format and stage
- Format - same as bracket and stage
- Result - attribute of match
- Round - does have logic of its own

Events

- Match created/started/concluded/skipped/deleted/edited
- Signed up/profile created
- Disqualified
- Stage created/started/concluded/skipped/deleted/edited
- Stage seeded
- Team created/deleted/edited
- Result created/deleted/edited
- Result received ?? is this part of problem domain

Not used

- Stream started - Does not affect any objects
- Halftime started - Does not affect any objects
- Bracket started/concluded - A bracket is just the structure of a stage and can't start
- Match planned - Same as match created
- Profile viewed - Not an event in problem domain
- Bot crashed - Does not affect model, should be solved with tools instead
- Bridge connected - Not an event in problem domain
- Match rearranged - Matches does not have order
- Format changed - Format is the same as stage

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.

Actor

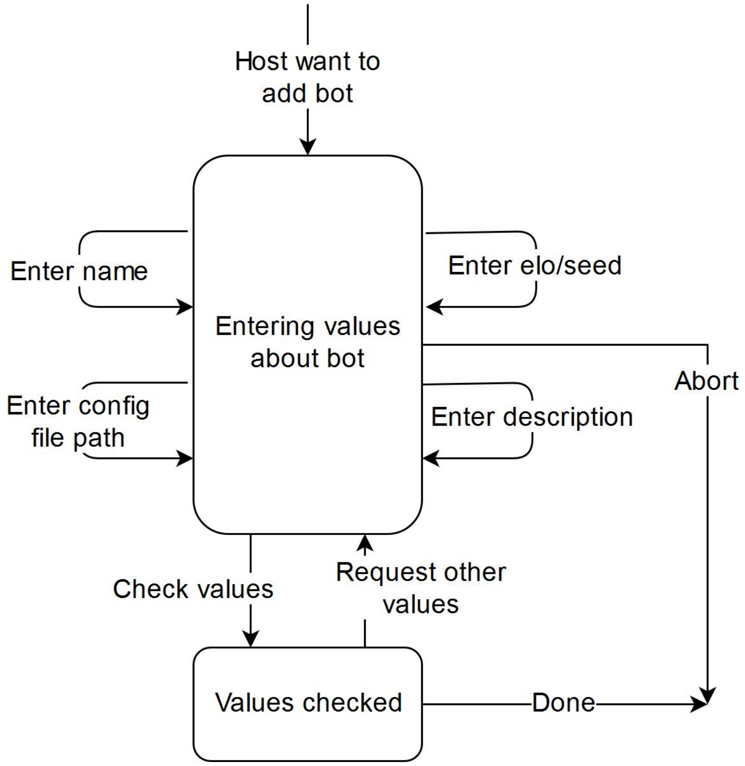
- Host
  - A human
  - Organises the tournament and (maybe) streams the matches.
- Streamer
  - A human

  - Streams the tournament and manages results and the pace of the tournament.
  - Acts as an announcer
- Challonge
  - A system
  - A third part service that can administrate a tournament
- RLBot
  - A framework
  - Facilitates the actual gameplay
- RLBot-bridge
  - A plugin to the actual system

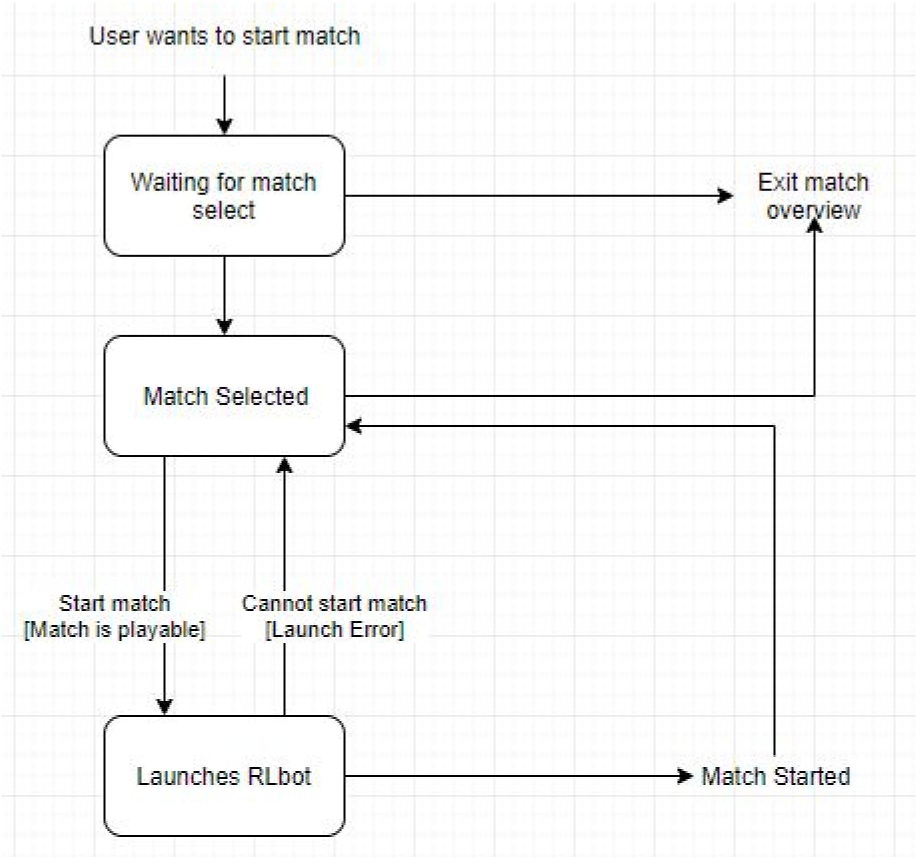
- A plugin/bridge between the **SYSTEMM**, and the RLBot-framework.

Use cases

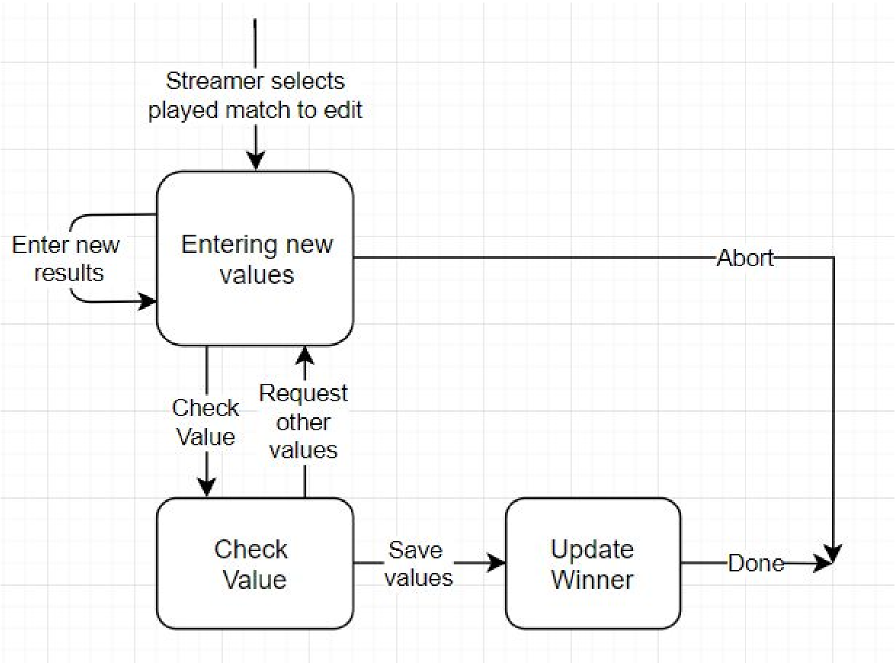
- **Register bot:**
  - **Use case:** The host decides to register a bot for the tournament. The host must enter the bots name, its path to its config, its elo, and optionally a description. Values can be entered in any order, and when done, the task i complete. The task can also be aborted instead of entering a value.
  - **Objects:** Bot, Profile
  - **Functions:** (TODO)



- **Start match:**
  - **Use case:** 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' names.
  - **Objects:** Match, player, team.
  - **Functions:** (TODO)

- 
User wants to start match
Waiting for match select
Exit match overview
Match Selected
Start match [Match is playable]
Cannot start match [Launch Error]
Launches RLbot
Match Started

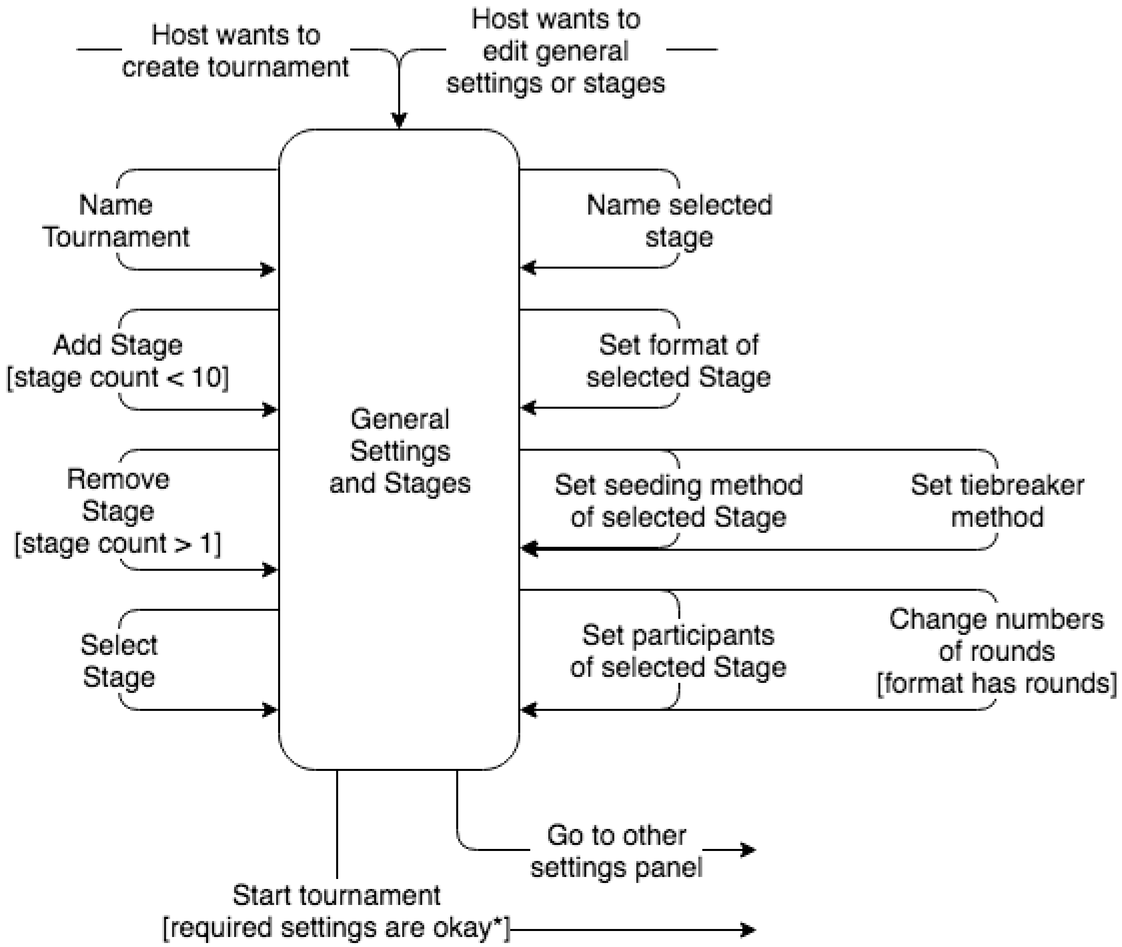
- **Edit a played match:**
  - **Use case:** The streamer decides to go back a edit the results of a played match. The streamer will select a match to edit, and enter the desired results. The streamer ends the task at any time, either saving the values or abouting without save.
  - **Object:** Match, results, player/team.
  - **Functions:** (TODO)



- **Change general settings:**
  - **Use case:** The user has either just created a tournament, or wants to change some general settings for the tournament. This leads the user to a general settings screen. Here the user can:
    - Name the tournament
    - Add a stage (but there can be no more than 10)
    - Remove a stage (but there must be one)
    - Select a stage

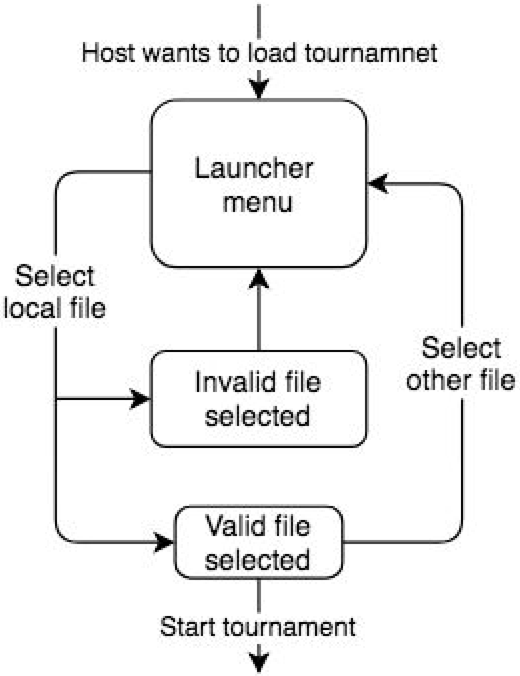
    For the selected stage the user can:
    - Name the stage
    - Set the format
    - Set the number of rounds, if the format support rounds
    - Set the seeding method
    - Set the tie breaker methods
    - Set the participants participating in the stage

    At any time the user can also choose to go to another settings panel. If all required values has been entered, the user can also start the tournament, which ends the task.
  - **Object:** Tournament, stage
  - **Functions:** (TODO)

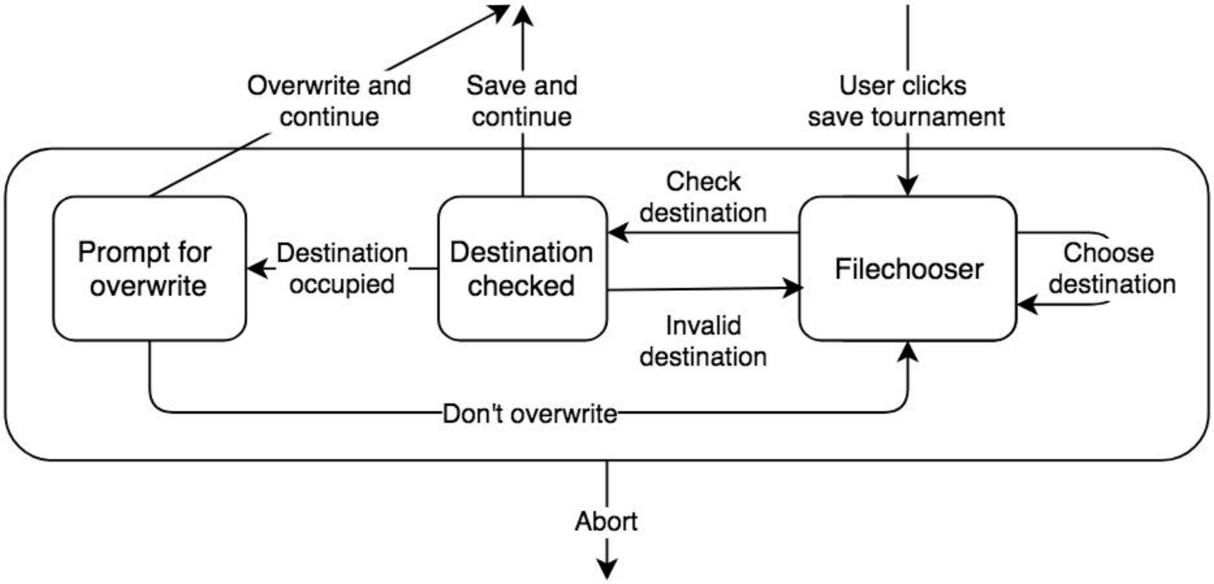


Host wants to create tournament
Host wants to edit general settings or stages
General Settings and Stages
Name Tournament
Add Stage [stage count < 10]
Remove Stage [stage count > 1]
Select Stage
Name selected stage
Set format of selected Stage
Set seeding method of selected Stage
Set tiebreaker method
Set participants of selected Stage
Change numbers of rounds [format has rounds]
Go to other settings panel
Start tournament [required settings are okay*]

- **Load tournament**
  - **Use case:** The host wants to load a tournament.
  - **Objects:** Tournament?
  - **Functions:** (TODO)

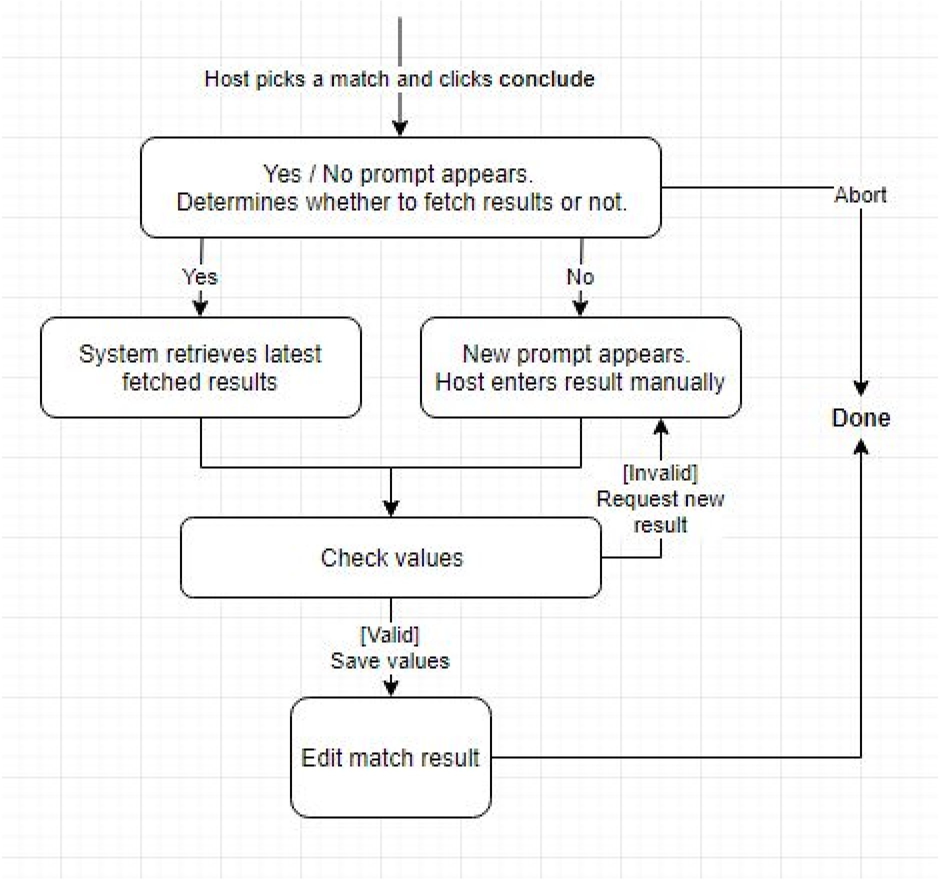


- **Save tournament**
  - **Use case:** The host wants to save the tournament, so it can be loaded later on.
  - **Objects:** Tournament?
  - **Functions:** (TODO)



- **Conclude a match**
  - **Use case:** The streamer wish to conclude a match. They pick the desired match and choose “Conclude match”. This action will prompt a “Fetch result” box which allows the host to fetch the results from the framework. If no is chosen, the host will be prompted to manually enter a result.

- **Objects:** Match and results
- **Functions:** (TODO)


Host picks a match and clicks conclude
Yes / No prompt appears.
Determines whether to fetch results or not.
Abort
Yes
No
System retrieves latest fetched results
New prompt appears.
Host enters result manually
Done
[Invalid]
Request new result
Check values
[Valid]
Save values
Edit match result

Actor Table

Use Cases	Actors				
	Host	**Streamer**	**Challonge**	**RLBot-framework**	**RLBot-bridge**
Register bot	X				
Start match		X		X	
Edit a played match	X	X			
Register tournament	X		X		

Criteria

Criteria	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			

System definition 1 (outdated)

Functionality

Manage different tournament systems and record results, both from manually inputted information or importing real-time/post-match directly from games. Analyse unfairness in ranking-based tournament-systems. Store history of players, teams, and matches.

Application-domain

The system should be an administrative tool for hosts of tournaments that manages stages and matches, supplies overlay-information for a stream, as well as produces statistics and history on matches for external use.

Conditions

The system will be used by experienced, and novice, hosts of tournaments, for a wide array of sports and competitions.

Technology

The system will be developed in Java and run on a networked x86-machine optionally able to run alongside computer games. The user-interface will be accessed through interacting with the host machine.

Objects

Player, profile, team, match, stage, result, tournament

Responsibility

Administrate tournaments and arrange sequence of matches, along with keeping track of results and relaying this information through an overlay. Will warn about when a change affects the fairness of the tournament.