Design

*FieldName*: final (set in constructor)

|  |  |
| --- | --- |
| **StartUp** | |
| List of Players | CopyOnWriteArray<Player> |
| List of antbrains | CopyOnWriteArray<AntBrain> |
| List of antworlds | CopyOnWriteArray<AntWorld> |
| List of tournaments | CopyOnWriteArray<Tournament> |

*Tasks :*

* *Store files (access & modification done by Players)*
* *When a challenge is accepted, run the corresponding game and send the full tracking back to players.*
* *When a tournament reaches the required number of player, start it : run all of the games like if they were classic games but use a class to keep track of victories and defeats, manage results, etc.Then, same, send the results back. If scores are kept somewhere, their update should be launched by the StartUp at the end of the tournament.*

|  |  |
| --- | --- |
| **Player** | |
| StartUp | StartUp |
| Name | String |
| List of received challenges |  |
| Stats |  |

*Tasks:*

* *Manage files*
  1. *Consult lists on StartUp*
  2. *Possibility to DELETE one’s own files (using the “author” field)*
  3. *Or ADD new files : type (brain/world) has to be given, file musn’t already be on the StartUp and syntax is checked to make sure that only correct files are uploaded*
* *Play classic game*
  1. *Consult list of received challenges and accept one of them to start a game*
  2. *Consult list of players and send a challenge to one of them*
* *Play tournament*
  1. *Consult list of tournaments, join one*
  2. *Create a new tournament (possibility to have a password to make it private ?)*
* *Manage options for visualisation, name, …*

*Ideas :*

* *Allow players to change their name*

|  |  |
| --- | --- |
| **AntBrain** | |
| Author | Player |
| List of instructions | List = LinkedList of instructions  Instructions = LinkedList/Array of Strings ? 1st one is the instruction, others the arguments for this instruction |

*Tasks :*

* *Conversion file 🡪 array for upload*
* *(Conversion array 🡪 file for download ?)*

|  |  |
| --- | --- |
| **AntWorld** | |
| Author | Player |
| Map | 2-dimensional array |
| Tournament\_conform | boolean |

*Tasks :*

* *Converstion file 🡪 array for upload*
* *(Conversion array 🡪 file for download ?)*
* *Additional tests for tournament compatibility*

|  |  |
| --- | --- |
| **BrainVerifier** | |
| … | ... |

|  |  |
| --- | --- |
| **WorldVerifier** | |
| ... | … |

|  |  |
| --- | --- |
| **Tournament** | |
| List of games | Player |
| … | … |

*Tasks :*

* *Manage competition to determine final scores and ranking*

|  |  |
| --- | --- |
| **Game** | |
| Player1 | Player |
| Player2 | Player |
| Brain1 | AntBrain |
| Brain2 | AntBrain |
| World | AntWorld |
| Traces | File |
| Stats1 | Statistics |
| Stats2 | Statistics |

*Tasks :*

* *Run the game*

|  |  |
| --- | --- |
| **Challenge** | |
| Sender | Player |
| Receiver | Player |
| World | Antworld |
| Message | String |

*Tasks : nothing 🡪 when accepted, field values are transferred to a new variable game and the challenge is then deleted from the list of the receiver*

*Ideas :*

* *Add a list of sent challenges too ? (or reference them with the received ones)*
* *Save them in files for next connection to the game*
* *Add a date of creation to automatically delete awaiting challenges*
* *Allow Players to choose the duration of challenges in options*

|  |  |
| --- | --- |
| **Statistics** | |
| Kills | Int |
| Death | Int |
| Food | Int |
| RoundNumber | Int |

## Precisions

* The synchronization between different functions isn’t directly coded, it’s all done automatically by the functions available with the class CopyOnWriteArray (and the other concurrent collections available in Java).
* The arrays representing both types of files could be filled with Strings or enumeration items ?
* For now, the visualisation isn’t mentioned, because it’s separated enough from the rest not to have any impact on the software structure (options to add in menus, a few methods and new classes).
* Will have to determine the repartition of tasks around the tournament (between Tournament and StartUp)