Genius 2.0

Overview of the documents we are creating (or still need to create) on our way to Genius 2.0

* [Requirements doc](https://docs.google.com/document/d/13yz40f_nMOci5_9jmaEGCUpuZAx-aObRA9fdIOPBXlg/edit#)
* [User stories](https://docs.google.com/document/d/1AWmP2dmNV6LHbo9dg0ZtJg7G-Amf0fdcdjop-3gJ4MQ/edit#heading=h.f7ev093b4j1u)
* [Comparison to other platforms](https://docs.google.com/document/d/1vjYVLuIa2YhEpJzkJi0Ctb_4lHdNWTKML4fcYY93mi0/edit#)
* [Phases of development](#_t2p1kbi8awsv)
* [Architecture and class diagrams](https://docs.google.com/document/d/13Sb89GNoyv8KGTXmBXBbf9UxPw9emIq4xTs-RaCvau0/edit#heading=h.t0ebdlwua4ek)

# Discussion

* How to divide the work amongst us.
* Jupiter calls Genius if they want to use a java agent (e.g., old agents).
* Jupiter calls http as communication channel, REST interface.
* PforJ
* Jupiter plans: we have to discuss about that, create a platform that is accessible from the internet. What would the platform do? Agents can submit to server, learn on the server. Agents can also be on a remote server and then negotiate with us on the server.
* Also for part of the ANAC competition? Yes, let’s see the strategy. It depends on the time line of the development.
* Who will do the development? These two students? From April we have 8 new students. One or two can join the development. They will graduate on that as a project.
* They could come to Delft or CWI.
* <we can send an invitation>
* Share the current design of Jupiter. We can modify it or use it.
* Domain & profile editing and upload: we discussed a few things:
* Security concerns related to uploading. Current design does not allow “upload”, users just put the data on their own web server (domain server, partyfactory server)
* Provide an editor? Just a java app (more in line with overall approach)? Or a web based editor (not so in line but maybe more convenient for users? But we need the ajax interface anyway)s
* Custom protocols? We can’t allow users to upload web server functionality directly (security). It seems better to provide some adjustable protocols instead.
* Upload domain; after that upload profile. Edit of existing not possible (we do not manage ownership of files). Currently we don’t allow upload, see above.
* Different running methods: (1) run without checking the domain (negotiators can do anything) (2) check the bids are in the domain (negotiators use their own, non public profile) (3) run sessions with given domain PLUS profile. Currently checking (or not) is a decision made by the protocol.
* Web interface to see progress on running session/tournament
* Download the logs/results (ajax call?)
* Analyse the results of a session/tournament (based on the logs for that session I suppose)
* Strictly, we do not need “persistent data storage” for parties anymore, users can implement this themselves as they like on their machine. But if we do it that way, agents may not run on other machines (like our own server if we want to test agents). So it seems we need to discuss this. Currently we leave this to let users figure it out, store however they like on their own machine.
* How about switching from XML to JSON also for file formats. We have many objects that need to be serialized for the function calls in the services and using both XML and JSON just seems to double the amount of work in the serialization area. I need to test if this would work in practice. Current plan is to use JSON all the way

# Phases in which we do this

1. Domain description (we keep the flat space with issues&values?)
2. Profile description
3. Domain & profile editor
4. Protocol description
5. Session runner
6. Tournament runner
7. Result logging
8. Statistic logging
9. Real time result display