Project Description

The goal is to develop a fully-engineered distributed software system using the technologies learned about in SWA and applying principles of project engineering from PRE.

In order to get the effect of team-work, all people in SWA should be actually do some coding.

Brief overview: → general structure but specific application

There is a type of system referred to as a rule-based system. This type of system makes decisions based on a series of questions. These can be used to make all sorts of decisions. Some examples from theses:

- Decide if a hot strip mill investment pays off
- Decide which hood furnace for steel should be used next

The decisions can be yes/no, calculations, or "fuzzy".
All teams are in competition with one another – the best will "win" (in each class).

Customer requirements: • flexibility important

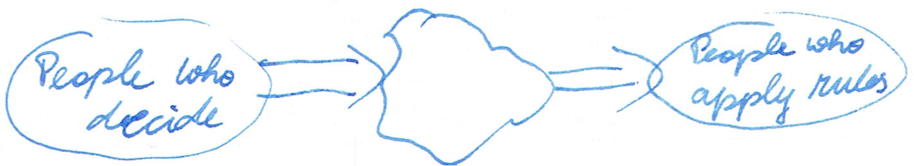
1. The system must be capable of reading in a set of rules.
2. For this class, one set of rules / one application can be selected as the focus.
3. The system must be capable of asking the correct sequence of questions to give an answer based on the (current) rules.
4. There must be a way to get an overview of the current rules. → which rules are in the decission tree → readable.
5. It must be possible to delete the whole set of rules.
6. The import format and any limitations of the rule database should be clearly documented.
7. Anyone in our class should be able to import the rules within 10 minutes using any provided help documentation. Furthermore, this group of users must be able enter the data with the information provided on the screen. answer the question
8. The technology (java, c#, etc.) can be chosen by the team, however, a distributed SW architecture must be implemented using either remote calls (e.g. WCF) or sockets. demonstrate to make sure you understand distributed SW.
9. The software should run on the computers in the EDV labs (Windows 7).
10. Optionally, it could run on multiple platforms (e.g. Mac, Windows 10, Linux).
11. The software should be able to be installed from a (provided) USB stick which also contains any needed user documentation. *
12. The project must be fully documented, including estimates, project plan (with tracking), risk list, kick-off, requirements specification, test plan, test results, issues/status, quick help, post-mortem and list of what each person contributed.
    Ideally also list of other user documentation that would be required and table of contents for developer's handbook
    For the technical side (SWA), in addition to a working program - the architecture overview (including justification), some design diagrams, code with file headers and quick help will be required.
    These should be delivered separately – either uploaded in moodle or on a separate USB stick.
13. The delivery will be done in class (usually the last class of PRE/SWA). Here it will be necessary to demonstrate that the system works. At this time a short presentation about your project is possible.

Note: Your company motto may affect the focus chosen in the development.

* If requested, the USB sticks can gladly be returned at a later date.

Szen 1 • central Rule Database → several users

Szen 2 •


People who decide
People who apply rules