Maintenance Task: Refactoring

Refactoring is an activity that involves the analysis and alteration of an existing body of code. The alterations improve the structure of the code but do not change its functionality in any way.

In this task, your group must identify and analyse parts of the Saros code and implement a series of refactorings. This will require some background research on good patterns and bad patterns (also known as anti-patterns). Some refactoring types are here: http://www.refactoring.com/catalog/index.html - but they all usually involve renaming, restructuring, breaking up responsibilities or improving abstraction. It also helps to add code documentation when it is missing.

- You should focus on certain area(s) of the Saros architecture.
- You need to identify problems in the code structure and argue why they need refactoring (e.g. are they a “code smell” or even an anti-pattern?)
- Design and implement the refactoring.
    - What type of refactoring have you used?
    - Have you used a design pattern? Identify it.
    - You should add any appropriate comments in order to document the code.
- If you can use a metric to help back up your claims of improvement, you should do so.
- Make clear how you proved your refactoring didn't have side-effects (this may involve writing a few unit tests).