**Read all the code in one hour**

After noticing the lack of documentation, we went on to the step: 'reading the entire code in an hour.'

*Checklist*

* Coding style
* Tests
* Abstract Classes
* Classes high in the hierarchy
* Patterns

*Findings*

Project is divided in 10 top-level packages.

be.ac.ua.ansymo.cheopsj and be.ac.ua.ansymo.cheopsj.branding do not contain any code, only some plugin configuration and licensing information.

be.ac.ua.ansymo.cheopsj.changerecorders contains an abstract class AbstractEntityRecorder. This is subclassed for various programming constructs (e.g MethodRecorder, StatementRecorder,...). There are unit tests available. There is no documentation (no javadoc or comments) also the tests are not documented.

be.ac.ua.ansymo.cheopsj.distiller contains 4 sub packages (distiller, cd, popup.actions, svnconnection). The main distiller package has one class (Activator) wich subclasses an abstract interface from the eclipse SDK. The cd sub package also has one class ChangeDistillerProxy wich suggests it follows the proxy pattern. The popup.actions sub package contains some classes that implement an abstract interface from the eclipse SDK suggesting they handle some user interactions. the svnconnection sub package contains some classes the handle an SVN repository. However it seems that the svn url is hardcoded. There are no tests for this package and there is no documentation.

The be.ac.ua.ansymo.cheopsj.logger package has 4 sub packages (logger, astdiffer, listeners and util). The logger contains an Activator class wich manages the Cheops singleton class that creates a listener JavaElementChange. It seems that this part sets up a listener for discovering user modification in the source code. The astdiffer contains two classes ASTComparator and DiffVisistor wich seems to suggest they compute the difference between two AST's. The listeners package contains a singleton class ChangeRecorder. The JavaElementChange seems to listen to changes (has a member elementChangeListener) and calls ChangeRecorder if a change occurs. The ChangeRecorder heavily calls the changerecorders package. There are unit tests available with some documentation the code itself has almost no documentation.

be.ac.ua.ansymo.cheopsj.model contains 3 packages (model, changes and famix). The model package contains an Activator that interfaces with eclipse. The is also a serializable singleton class ModelManager that is managed by the Activator. There is a class ModelManagerEvent that seems to represent one event in the model. The class ModelMangerListener is an interface. In the changes package there is an interface IChange that is implemented by the Change class. AtomicChange is a sublass of Change. Change is sublassed for all possible changes to a source file (add, remove, modify,...). There is also a abstract class Subject. The famix package contains classes for some programming constructs(e.g FamixClass, FamixFunction,...). Wich are all subclasses of the abstract class FamixObject wich extends the Subject class.

be.ac.ua.ansymo.cheopsj.model.ui has 4 packages (ui, handlers, changeinspector and changegraph) wich all seem to be related to displaying the model in a graph.

*Conclusion*

Refactoring CheopsJ to include support for git, CVS, mercurial has high risks. The classes to handle the SVN repository (SVNConnector, SVNLogEntryHandler) have no interface available. Introducing an interface to handle multiple source code management systems can have a large impact on the system. There are no tests available. Tests will have to be written first to allow regression testing. The code is not documented and the current implementation uses a hardcoded svn url wich makes testing more difficult.