"Software Engineering 2"
– *Metrics*

In this task several metrics should be calculated and evaluated.

1 Evaluating one Project

In the first task a single project should be evaluated. You may either use a given project or a (relatively large) project of your own. If you want to use the given project download the eclipse workspace "`Metrics Workspace.zip`" from the Moodle course, unpack it, open it and do the measurements on the project `TreePanel – Core v1.3`. Otherwise load your project in Eclipse or open the workspace containing your project.

1.1 Preparation

Either which project you choose, do the following actions in Eclipse:

- Install the "Eclipse-metrics plugin continued" (https://sourceforge.net/projects/metrics2/) – for university computers, this is already installed.
- Open the "Metrics View" and the "Layered Package Table View" in Eclipse
- Select the project you want to measure and "enable metrics" in the project properties.
- If the metrics are not calculated, perform a rebuild of your project.

1.2 Evaluation

Now evaluate the results:

- Check which metrics are beyond threshold (marked as red) and open the crucial methods (can be done in the metrics view).
- Inspect and rate the methods personally – are they really potential troublemakers?
- Decide if these methods really need to be done that way, or what could be done instead.
- Try to find out the crucial 10% classes in the project.
- Export the metrics report as xml-file (through the metrics view menu)
- Especially inspect the Martin-metrics
    - Efferent and afferent coupling: how can the results be interpreted?
    - Calculate / Map the Distance for each package: which is in the zone of pain / usefulness?
- Inspect the layers in the layers view: do you have separate layers, do they have a common concern?

1.3 Programmatic Export

For a (possible) nightly build metrics should be collected for each build (and probably evaluated over time). To do so create the following ant-script in your project:

```
<project name="Metric Export" default="main" basedir=".">

     <basename property="project.name" file="."          />
     <property name="build.delay"        value="5"      />

     <target name="init.stamps">
          <tstamp/>
     </target>

     <target name="export" depends="init.stamps">
          <echo>Exporting metrics for '${project.name}'</echo>
          <eclipse.refreshLocal resource="${project.name}" depth="infinite"/>
          <metrics.enable projectName="${project.name}"/>
          <eclipse.incrementalBuild project="${project.name}" kind="full"/>
          <echo>Wait for ${build.delay} seconds
               for the build to be finished</echo>
          <sleep seconds="${build.delay}"/>
          <metrics.export
               projectName="${project.name}"
               file="${project.name} - metrics - ${DSTAMP}-${TSTAMP}.xml"/>
          </target>

     <target name="main" depends="export">
     </target>

</project>
```

Remark: The incremental build is a background process, as I could not find a way to find out, when the process is finished, I simply wait for 5 seconds. Then the export starts. Otherwise the export may fail, as the project is still busy and cannot be accessed – maybe you find a better solution for this.

To successfully execute the script, run it in the same JRE as the workspace (otherwise you cannot access eclipse and metrics-tasks). To do so call “Run as..” and edit the launch configuration at the tab “JRE”).

After the script has been executed, a xml-file with the collected metrics should be available in your project. Open the xml-file and inspect the results.

2 Evaluation of a Set of Metrics over Time

The Metrics Workspace contains my development versions of the previous task (“Expression Parser”). This includes some initial test versions as well as some in-between (erroneous) versions (the last version is the actual solution to the task).

Within this task the development of selected metrics should be observed through the development of the projects (with 00.1 - … being the oldest and 03.5 - … being the newest).

2.1 Creating Reports for all Projects

You will find all necessary tasks and classes in “SE2 - Metrics - Tasks and Classes.zip” in Moodle, so download and unpack this first.

- First create a Java-Project and copy the build.xml (from Tasks and Classes) into the project.
- Then create a folder called libs in your project and copy the ant-contrib-....jar into this folder. The ant-contributions add control-statements to ant (e.g. a for-each-task). While this contradicts the idea of ant, this is necessary if you want to make script which goes beyond building – like here, where you have to build a set of projects. The ant-contributions are included with the task-def.
- Now run the build file (again within the workspace-JRE). As a result you should have for all projects (specified by a prefix in the script) the individual reports in the folder “metrics”.

2.2 Aggregation of the Reports

This is the actual programming task.[1]

Given (in the above archive) are three classes ReportEvaluator, Metric, and Value. The ReportEvaluator iterates over the reports and collects the summaries for each report and creates a metric object for each file. The result is a list of metric objects, which are printed at the end.

You should extend this Class in a way, that it generates a csv-file. A csv-file is a plain text-file which can be read by excel. All values are separated (in a German environment) by “;”. It is basically an excel-sheet without any formats, formulas, etc – just the plain text.

The csv-file should tabulate the overall results:

- The header should contain the name of the respective project (stored in the Matric as xmlFileName)
- Each row should contain the values for one respective metric.

[1] Of course the main goal is a report – so you are free to use any other tool for generating the overall report instead of programming it on your own.

- In the end something like this should be written:

	00.1 - Metric	00.2 - Metric	00.3 - Metric	…
Cyclomatic Complexity [VG, avg]	1,23	2,45	3,3	…
Cyclomatic Complexity [VG, max]	7	10	12	…

- The suggested point for adding this export is at the end of the method `processDirectory` (or in the main method).
- You may select only certain values, e.g. only the means or only certain metrics.

2.3 Visualize your results

After the csv-File open it (with Excel) and display the values in a diagram and evaluate it Remark: If you have values in extremely different ranges, you may use a secondary axis in your diagram.