Current Topics in Software Engineering: Software Evolution/Reengineering

**Analysis Report - Karaoke MusicVideoManager**

Alexander Isopp, Manuel Langer

# Initial Analysis

This chapter describes the Karaoke Music Video Manager Application and provides an initial analysis of the project.

## The Karaoke MusicVideoManager

The Karaoke MusicVideoManager Desktop Client is a standalone Java Application which provides a convenient way to manage music videos. Figure 1 shows the GUI of the Application.

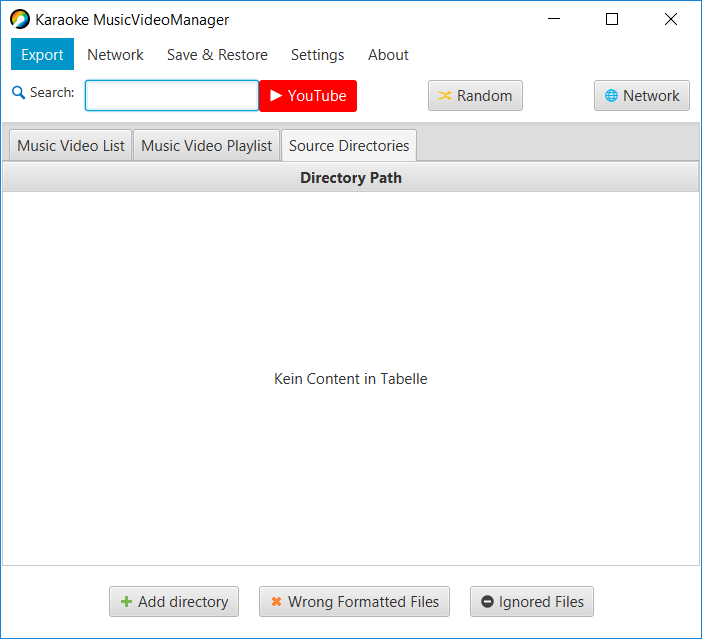


Figure 1: The Karaoke MusicVideoManager Desktop Client GUI

## Read the code in one hour

Inspecting the packages of the desktop client shows that following packages contain the most important source files:

* desktopclient.gui.controller
* desktopclient.handler
* desktopclient.libaries

Here, especially the classes MainWindowController and MusicVideoHandler seem to play a central role in the design of the application. For code documentation JavaDoc is used. From this documentation and some JavaFX specific annotations, we know that the JavaFX framework was used to build the UI. Unfortunately, there are just a few automated tests available.

## Do a mock installation

Importing the DesktopClient into an IDE like Eclipse works fine, but problems arise when using Java JDK 11 or higher. Here, the JavaFX framework is not included in the Java SE anymore, therefore it’s not possible to compile the project without manually adding the required JavaFX libraries. However, the project compiles and builds fine using Java 8. In order to run the automated tests, the JUnit library needs to be imported into the project.

## Essential Metrics using inCode

The tool inCode provides an overview of some basic metrics to get an idea about the size and complexity of the project. InCode reports the total LOC ~11500 with 37 classes and total 420 methods.

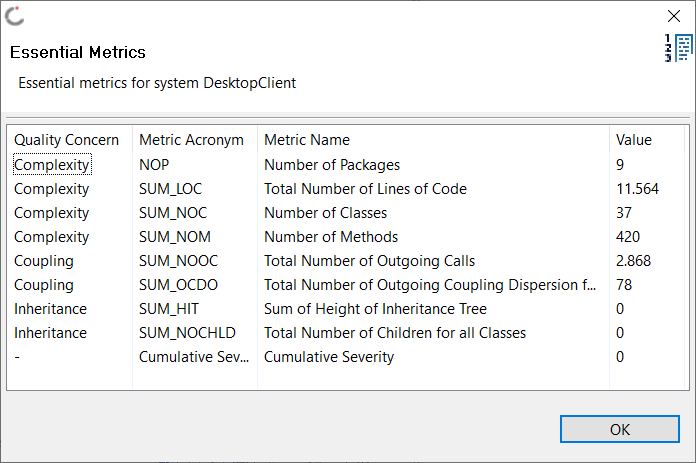


Figure : inCode - Essential Metrics

## Initial Analysis using SonarQube

An initial analysis using SonarQube shows that there are 13 bugs, 65 vulnerabilities and 578 code smells, as shown in Figure 2. Also, SonarQube reports 5% of duplications and 26 duplicated blocks. In order to fix these issues a more detailed analysis of the project is required.

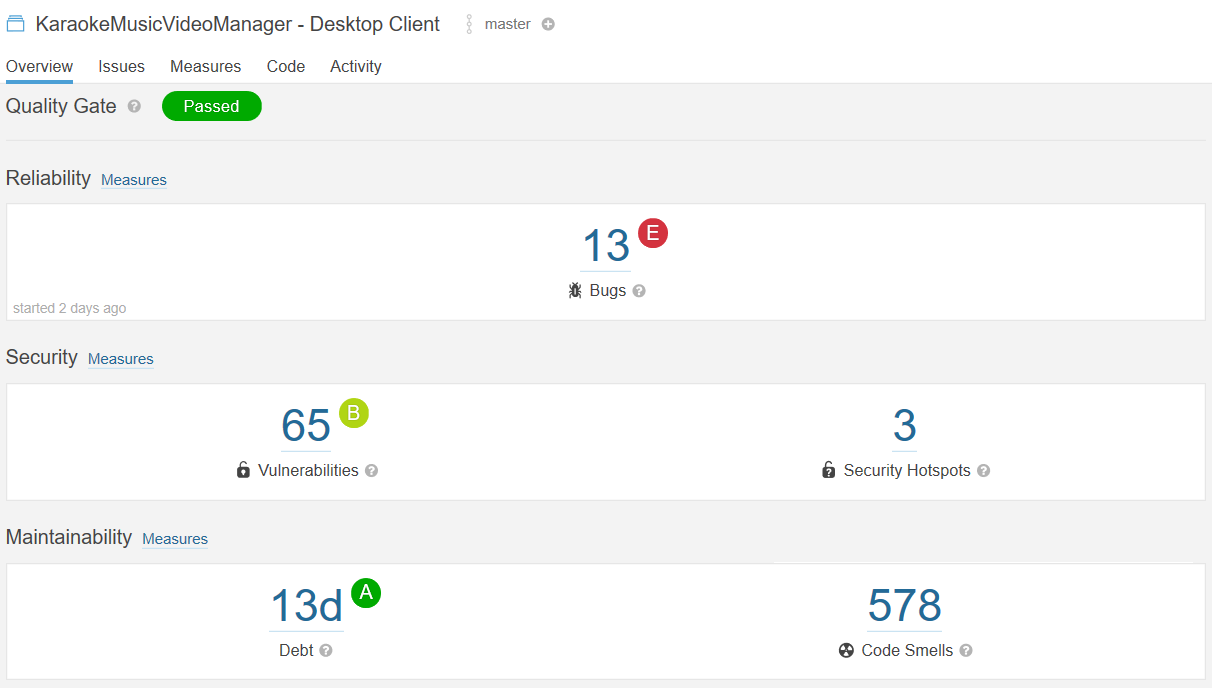


Figure 3: Overview of the project quality in SonarQube