Ciro Larrazabal
Samuel Bissig
Emanuel Indermühle

Report for reverse engineering ESE06 Projects

ESE1

Overview

GUI is not really that, is a mix between GUI and god class. The functionality for the application is not really intuitive, there are also a complex mix with the concept playlist and library, because you can only import leads to a current playlist but not a playlist or a library. There os not a status message or visual signal to indicate that something is happen.

Extensibility of the Design

There a lot of information and classes the ones can be easyly changed in order to improve the functionality of the application, but not in all the directions, there are constrains in some classes which make very hard to change.
They try to follow the MVC pattern but they couldn't. Most of the problem would come from musicplayer

Difficulty to replace the view

Very hard because in the GUI part there are part of the program's logic. But is not a nightmare because there are many good design structures.

Difficulty to replace the persistence strategy

It would be easy to implement a persistency strategy in this software.

Tests

They are a little tricky but good implemented. They cover a lot but fail with a dense approach. All the tests were successful.

Error handling

It is an Elegant handling, mostly there a couple of case where they try to make funny catch but mostly good handled.
The main approach is catch und ignore.

Design Patterns

The MVC design pattern is implemented and documented.

Code Smell

Dublicated code is not significant, but there are a lot of methods that are commented, because hat many problems or lead to unadvertisements errors

Documentation

The code documentation is a great help, but the design document isn't useful.

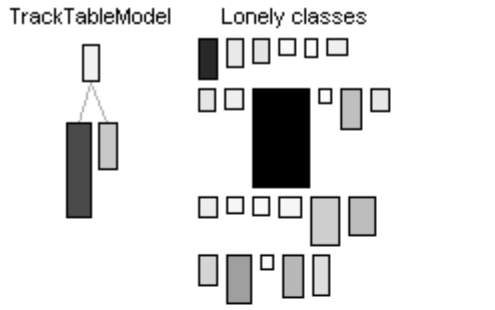
Team work

From the CVS log there are member who, participated in only one document and these time, just once.

Estimation for M3U implementation:

Because of the good design we would get fast into the code.

Get into the sourcecode 1.5h
Write export import and export: 2h



ESE2

Overview

The Project is quite compact. Its seams that the intention was to implement a MVC architecture there is a model, a GUI and two controller packages. The controller not only handles the communication between model and view, but also do some model activities.

Extensibility of the Design

The use of the MVC architectural pattern promise a good extensibility, but it depends on the implementation. In this case the coupling between the model and the view is hardcoded. If the Observer pattern would have been used, it would be quite flexible

Difficulty to replace the view

As mentioned before coupling between view and model extensible, but that could be changed quite fast and so it would not be difficult.

Difficulty to replace the persistence strategy

Two classes are responsible for the data storage, It seams easy to implement an other persistence strategy. The design do not support plugins so its not pluggable, but this is not need in an project with such a size.

Tests

Test classes for five classes of the model exists. This is not a lot moreover a lot of action is defined in the controller package. Only 17 tests are made, this is not enough, but all the the tests are successful.

Error handling

Errors are catched and ignored. The application reacts bad on errors, eg. if a file is removed, the hole system crashes and eclipse crashes too.

Design Patterns

There was no pattern jumping up to my mind when reading the code, besides MVC architectural pattern. No documentation is found about patterns.

Code smell

The project smells good.

Documentation

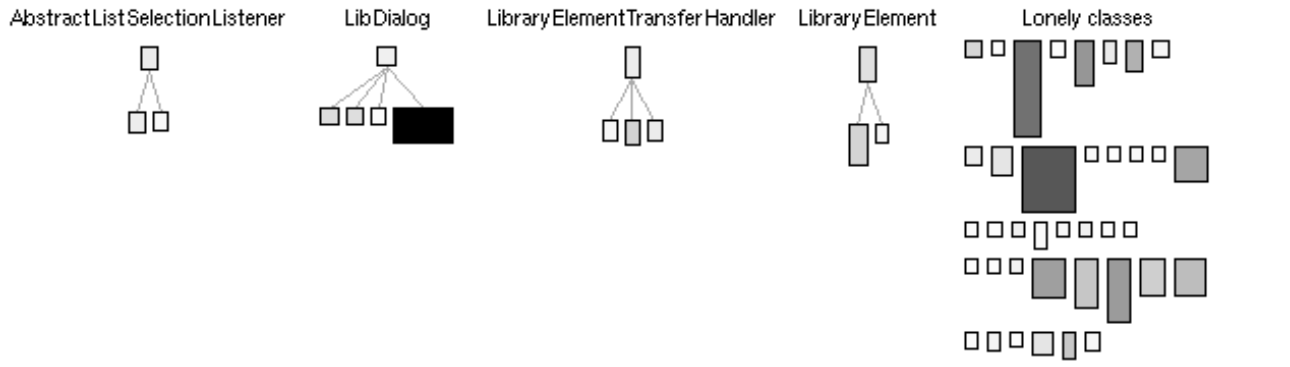
It is not very useful because there wasn't a lot of documentation, but what was there was useful. There was a user manual, but since the application is self explaining, this wasn't useful ether.

Team Work

A lot of work is done by zeradun, he seams to control the project. Mariusmoser and mbaumg are also doing a lot. Silvan is not commiting to Java files and Mike as three commits

Estimation for M3U implementation:

4 hours.



ESE3

Overview:

The Project has a bigger than others and its not easy to get a overview. The model is quite small and clear, but the view has a lot of classes and the architecture is not very clear. The application is seams cool and it even has an equalizer and a visualization, but documentation did not kept up with it.

Extensibility of the Design

It is not really extensible. The responsibilities are spread over the hole project. The data are not centralized

> eg1. There is a PlaylistLibrary, a list of playlists, To display this list, there is a PlaylistTableModel that copy the data of Playlistlibrary in an array , only for handle it further to a JTable
>
> eg2. Every class has its instance variable referring to a lot of objects of the system. Better would be to have a defined interface for the view to access the model and for the controller to access the view and the model. This will be a problem if you like to

replace an instance of a model element.

The View and the Model are not separated enough

eg. The class DirectoryImporter is responsible to import music files from the file system. But it also displays a dialog for user interaction and it handles with controller.

Difficulty to replace the view

The view is not implemented flexible, it is to deep integrated in the model.

Difficulty to replace the persistence strategy

And for the same reason it is not easy to change this strategy

Tests

The tests are covering the basic model and the access to the database. For this it is useful. But a lot of action take place in the Controllers and in the View, this is not tested The implementet tests are working good.

Error Handling

The errors are mostly captured and ignored. The system is robust, but the user don't know when and why an error occurs.

Desing Patterns

No patterns are discovered, if there are patterns, they are hidden good.

Code Smell

Some instance variable are accessible "protected" and not "private". There are a lot of eclipse warnings, so it is smelly

Documentation

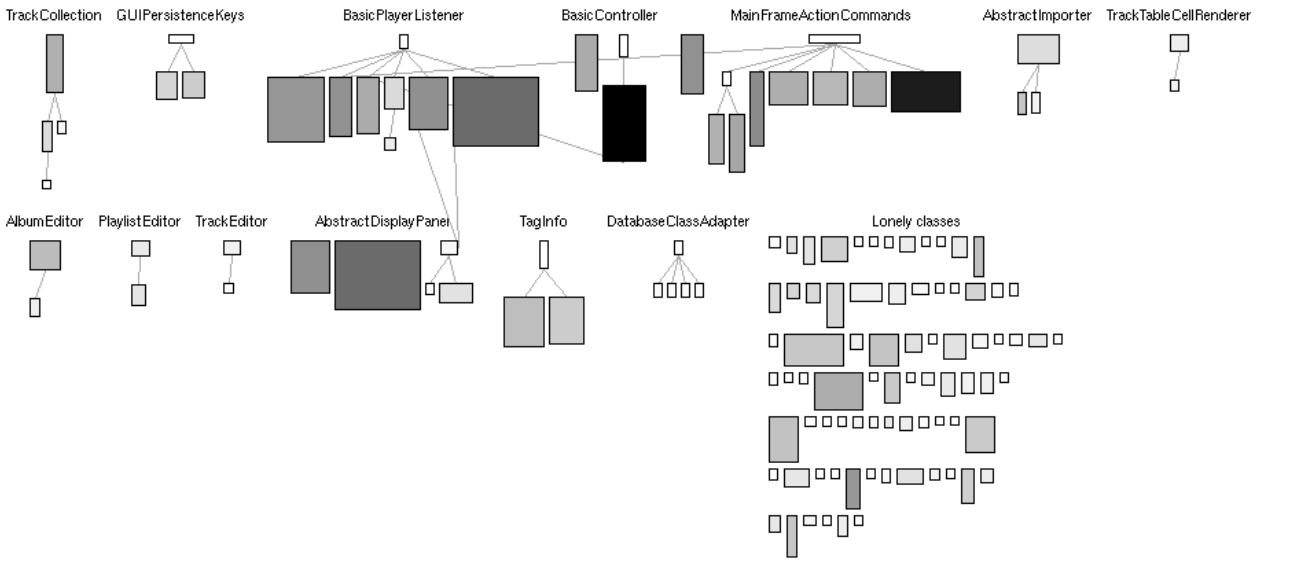
It is not useful. Only the simple classes are documented. There is only documentation for the methods but not for the classes. There is a nice UML diagram for the model and a HTML user manual. Inline documentation is also missing

Team work

There is one member that controlles the project (cami). (pkofmehl) und (mab) are active in on a few files.(rusmus) does even less. (daniel) does some small commits (debugging , formatting). (Hispanico) does a few big commits

Estimation for M3U implementation:

6 hours.



ESE4

Overview

- GUI is quite large, could more compact.
- Would be nice to have functionality with right click (e.g. edit playlist).
- It's not intuitive to add a new track.
- During adding a second time new files to a playlist the application throws exceptions, after that I am not able to add any other files.
- No progress bar, it is neither possible to go forward and backward in the files nor jumping from one to an other song.
- No readme found.
- There is a folder "files" and a package "ressources"? Why?
- Following the package names it is not clear, which package contains the view and which contains the model.

Extensibility of the Design

We are not able to find any structure in the design. If the classes would be saved in packages like gui, model, controller it would be much easier to see which class does what. There is a god class: jTunes.java. It contains all the view an manages the model. This should be splittet off.
There seams to be a big change in the project. There are classes with the same name (e.g. Playlist.java) in different packages. It looks like there will be added or removed some external sources (javazoom).

Tests

- All tests where successfully.
- Tests gives a good overview over functionality.
- Test coverage is dense.
- No Bugs, all the the tests are ok,

Error handling

There are a lot of catch statements, but the errors are not handled yet. Errors are not realy handled yet. The Team plans to add an "AlertBox" (//TODO Alert Box).

Design Patterns

Factory Pattern

Code Smell

There was a some dublicated code but we didn't understand the smalldude graphics.
There are some magic numbers.

Documentation

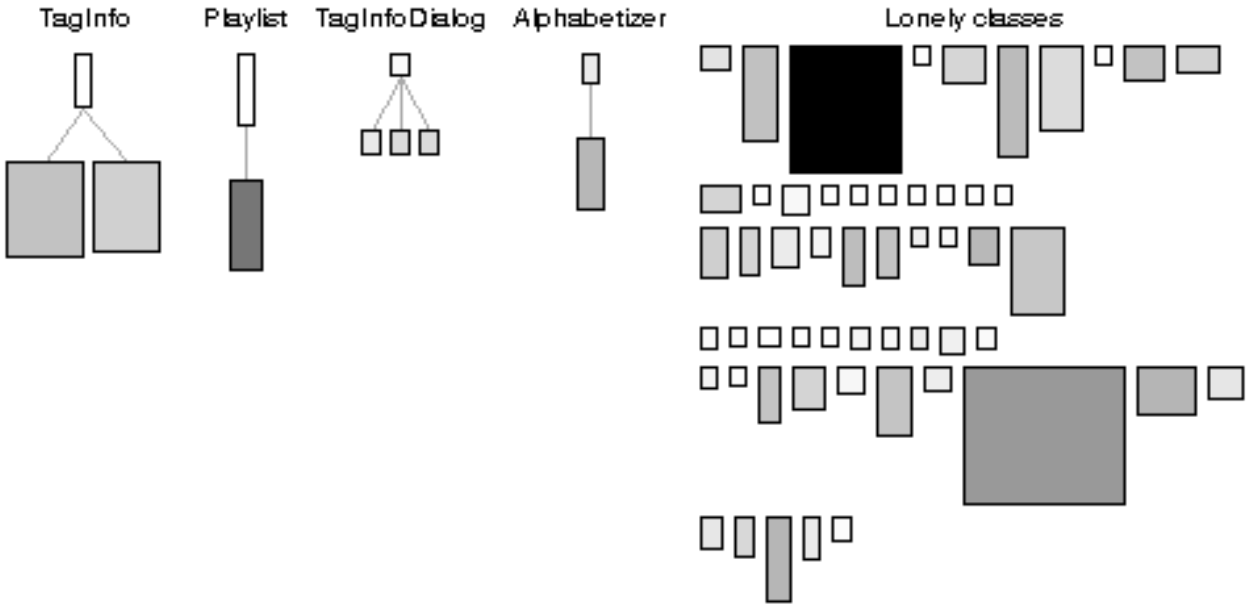
Some classes and methods are not well documented

Team work

All users participate more or less the sam. The user who did the initial commit owns most of the code, but after the initial commit he hasnt' much more commits.

Estimation for M3U implementation:

Because of the bad design we think that it would take some more time for an external programmer to add this futuer. As in other ESE project we found also in this one a Playlist Class. There we could implement this future.
Time Estimation:
Get into the sourcecode 2.5h
Write export import and export: 2h



ESE5

Overview

- A lot of external libraries.
- Exceptions: /home/sam\musicLibrary.xml (Permission denied)

- User Dokumentation (readme.txt) is good.
- Strang name: newGui.java? There is no old one? (Same for newTableModel.java)

Extensibility of the Design

There are some Objects like the Library, which are saved localy in the different classes. It could be an advantage to hold the Library only localy in the model and get it allways from there. Therefore the model should be a singelton.
This team has used the MVC pattern quite strict. The data is stored central, so it would be easy to extend the project.

Difficulty to replace the view

Because this team has splitted the project in model, view and controller it wouldn't be hard to implement a different view

Difficulty to replace the persistence strategy

We think it would be possible. All the playlist data is stored locally and managed in a Playlist class. So it would be easy to adapt or extend this one.

Tests

The tests throws errors, and not every method is tested, there are only 17 tests. Tests 4 Errors (out of 17) Could be because of the musicLibrary.xmlwriting permission.

Error Handling

Somethimes errors are not handled at all.System continued running, no message, that musicLibrary.xml could not be written

Desing Patterns

- Observer
- MVC

The observer pattern is documented, the MVC not explicitly.

Code Smell

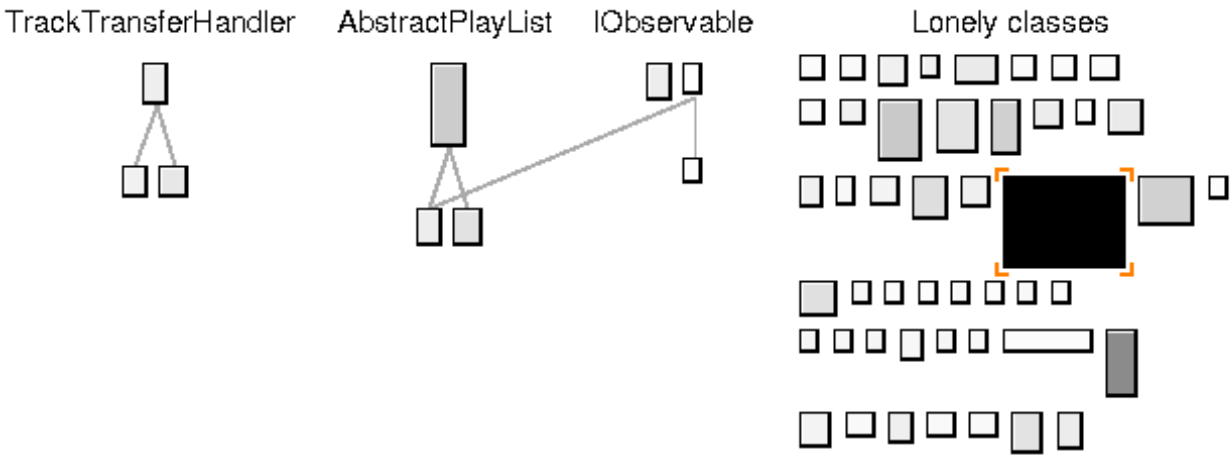
There was a some duplicated code but we didn't understand the smalldude graphics. The class addPlaylistGUI violated the naming conventions (AddPlaylistGUI). Shortcuts are sometimes written in lower letters. Different names for the same thing: playBtnController stopButtonController (once used a shortcut, once not).

Documentation

Most of the bigger methods are documented. The classes are not documented

Team work
The team member participate more or less the same.

Estimation for M3U implementation:
Because of the good design we would get fast into the code. Get into the sourcecode 1.5h
Write export import and export: 2h


TrackTransferHandler
AbstractPlayList
IObservable
Lonely classes