

# Assignment 1

Version: March 9, 2024

## Part 1: Reverse Engineering

### Objectives

- Practice Reverse Engineering
- Compare results of your manual reverse engineering and the automatic reverse engineering
- Use automatic forward engineering

### Prerequisites

1. The assigned reading on Reverse Engineering
2. Lecture on Reverse Engineering
3. Slides and videos about Class diagrams (optional)
4. Astah tool installed

### Tools

- Install Astah: <http://astah.net/student-license-request> it is free for students
- Install the Astah plugin for Reverse Engineering: <http://astah.net/features/code-reverse-plugin>. Make sure you check it is installed under "Installed PlugIns".

### Task 1: Manual Reverse Engineering (20 points)

Use the given partial Java code and create a class diagram from it in Astah. The "Main.java" class only contains some tests and should not be included. Also, do not change your diagram based on tasks 2.1 and 2.2; it will most likely not make it better.

If you like, you can import the project into Eclipse (or a different IDE) and run the Main method. This should help you understand what the code does and see how class diagrams and code work together.

You should only include the classes that are in the given code; do not include any other classes. Make sure to use associations (uni- or bi-directional is ok) and generalization and if necessary compositions and aggregations. You can include constructors but do not have to. Please look at the slides and the video that provide information on how your class diagram should look (it will be different from the ones in tasks 2.1 and 2.2 of this assignment).

At this level of abstraction, the relation between the classes should be shown through associations (do not include objects of a class type you have in your model as attributes in

another classâ€”use associations for that). You should include associations (aggregations/compositions if needed) with names and multiplicities, generalizations, attributes (with visibility and attribute types), methods (with visibility, arguments, and return types), and types. You should not include roles and stereotypes.

Export the diagram as an image and include it in a document (e.g., Wordâ€”more will be added to this document in the next tasks).

## **Task 2: Automatic Reverse Engineering (13 points)**

Now, we want to create our class diagrams automatically.

We will use 2 different methods:

### **Task 2.1: Automatic Reverse Engineering (5 points)**

Create a new Astah project (a new Astah file). Create an empty class diagram. Select all Java files in your file explorer and drag them into your Class diagram. If prompted, indicate that you want to parse the source code from the dropped files. This should generate your classes in your class diagram.

Export this new class diagram into your document under Section Task 2.1.

### **Task 2.2: Automatic Reverse Engineering (5 points)**

Create a new Astah project (a new Astah file). Create an empty class diagram. Now go to Tools -> Java -> Import Java and browse to your src folder for the project. Select all the Java files and click OK. Now it should ask you if you want to import the attributes as associations. Select all and let Astah import your classes.

You should now see the classes show up in the package structure on the left hand side in Astah. Now drag all your classes from this structure into your empty class diagram.

You should get another class diagram, which probably looks a bit chaotic. Rearrange it so we can see all associations.

Export this new class diagram into your document under Section Task 2.2.

Advice: Look at your diagram again and check if you have done it correctly, chosen the correct multiplicities, etc. Consider whether you should be closer to the first or second automatic export.

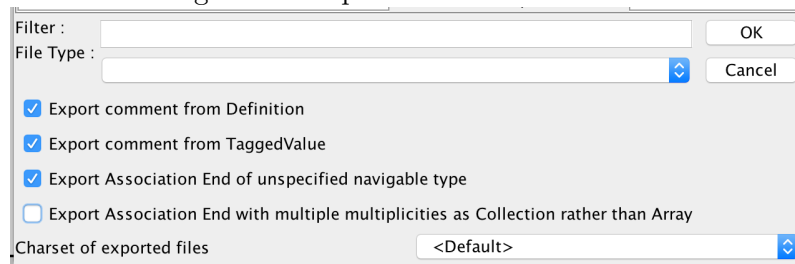
Explain in your document (3 points):

1. All three diagrams have a different level of abstraction. Explain the main differences.
2. Why do all three diagrams still represent the same system even though they look so different?
3. Why is an association basically the same as having the attribute in the class?

## **Task 3: Automatic Forward Engineering (7 points)**

Now export all three of your models into Java Code (create different folders for each export: Export1, Export2-1, Export2-2). You can do so by clicking Tools -> Java -> Export Java. This will get you a dialog for the export.

Use this setting for the export:



Filter :  OK

File Type :  Cancel

☒ Export comment from Definition

☒ Export comment from TaggedValue

☒ Export Association End of unspecified navigable type

☐ Export Association End with multiple multiplicities as Collection rather than Array

Charset of exported files

You need to select all classes in the folder to get a correct export.

Do not change the code; it will look different than the initial one, and that is intended.

Take a look at the different exports and answer the following in your document:

What are the main differences, and which diagram type would be the best to export to Java right away as a template? Explain your reasons under Task 3 in your document.

Tip: For your manually created diagram, check the code to see if it seems correct, e.g., does not contain duplicate attributes. In case it does, try to figure out why and fix it in your diagram (not code; after fixing, export it again into code).

## Submission

Submit

- **one** PDF document on Canvas which includes all your diagrams and answers, name this document `reverse_asurite.pdf` (where `asurite` is your `asurite` id, example for me it would be `reverse_amehlhas.pdf`).
- Submit all your Astah files
- Put all your exported folders with your Java code in one zip file and submit this zip file

Please submit exactly what is mentioned above. We will deduct points for different file types and different orders (e.g., the PDF in a zip file).