PLD Agile

Duration of the PLD: 8 sessions of 4 hours

Pedagogical team: E. Egyed-Zsigmond, P.-E. Portier, C. Solnon

1 Description of the application

You must design and implement an application for optimising delivery tours in cities. For more sustainable cities, these deliveries are done with bicycles.

When launching the application, the user loads an XML file which describes a city map. This file gives the list of all intersections (such that each intersection has a latitude and a longitude) and the list of all road segments (such that each road segment has an origin intersection, a destination intersection, a name and a length in meters). When a map is loaded, the application displays it.

To prepare a tour, the user first loads an XML file which gives the address and time of departure and, a list of requests, such that each request is composed of a pickup address, the duration of the pickup, a delivery address, and the duration of the delivery. Durations are given in seconds, and all addresses correspond to intersections. The application displays the position of each address of the file in the map.

Then, the application computes a tour which starts from the departure address, visits all pickup and delivery points (so that, for each request, the pickup address is visited before the delivery address), and returns back to the departure address. The duration of the tour is equal to the travel time plus the duration of every pickup and delivery. To evaluate the duration needed to travel from the origin to the destination of a road segment, you will assume that the travel speed is constant and equal to 15 kilometres per hour. The total duration of the tour must be minimal. The tour must be displayed on the map. The application also displays, for each pickup or delivery point, the arrival and departure time on/from this point.

Finally, the user may need to modify the tour (delete some requests, add new requests, or change the order points are visited). After each modification, the application updates arrival and departure times. The user may need to undo/redo these modifications.

2 Organisation

The project is done by groups of five to seven students, using an iterative development process. The project will be composed of at least two iterations, and the first iteration will be 4 sessions long. For the next four sessions, you may choose to do between one and four iterations.

First iteration: This iteration corresponds to the inception phase. The goal is to identify the main use cases, design a first architecture of your application, and implement some use cases in order to have a first minimal but operational software.

Deliverables that are required at the end of the first iteration are:

- Glossary;
- Use case diagram;
- Brief format description of the main success scenario of all identified use cases;
- Structured description of all use cases that have been analysed and/or implemented during the first iteration;
- Class and package diagrams designed during the first iteration;
- Actual planning of the first iteration (time spent by each member of the team on each activity).

You will organise a demo with the client at the end of this first iteration (during the fourth session).

Next iterations: At the beginning of each iteration, you will choose some use cases (or use case scenarios) that will be analysed, designed and /or implemented, and you will estimate the time needed for each of these activities. At the end of each iteration, you will compare this planning with the actual time spent on each activity.

The deliverables that are required at the end of the PLD are:

- Document where you explain architectural choices and used design patterns;
- Code of your application, including unit tests
- Javadoc documentation
- Class and package diagrams (reverse engineered from your code), with an explanation of the differences between these diagrams and those designed during the first iteration;
- A report on test coverage (automatically generated);
- For each iteration: Initial planning and actual planning
- Technical and human review of the PLD

Development environment: We recommend using Java for implementing the application. If you wish to use another object oriented language, you must first discuss your choice with us. You will use an IDE and tools for automating unit tests, for evaluating test coverage, for controlling versions, and for automatically generating online code documentation. You may use the Eclipse IDE[1] and its plug-ins: ObjectAid[2] (for reverse engineering diagrams from code), and JUnit[3] (for unit tests).

You may use the demo version of StarUML[4] to draw UML diagrams.

The online documentation of your code will be generated with JavaDoc, and you will apply the style guide of Oracle[5].

To compute a tour, you may use the Java code available on Moodle. The class *TSP1* implements the most basic (and naive) variant of algorithms studied in AAIA. You may implement more efficient variants by overriding methods *bound* and *iterator*. You may also implement another solving approach (dynamic programming or tabu search, for example), or even use open source libraries (provided that you credit authors of these libraries!).

[1]`http://help.eclipse.org/juno/index.jsp` (see the Java Development User Guide)
[2]`http://www.objectaid.com/`
[3]`http://www.junit.org/`
[4]`http://staruml.io/`
[5]`http://www.oracle.com/technetwork/java/codeconventions-150003.pdf`