|  |
| --- |
| Fontys university of applied sciences |
| Process report |
| Final version |
|  |
|  |
| **1/11/2016** |

|  |
| --- |
|  |

*Rosen Danev*

*Preslav Gerchev*

*Dimitar Vikentiev*

*Monica Stoica*

Table of Contents

[Introduction 2](#_Toc440287028)

[Rules and decisions 3](#_Toc440287029)

[Division of work 4](#_Toc440287030)

[Problems and solutions 5](#_Toc440287031)

[Individual part 7](#_Toc440287032)

[Dimitar 7](#_Toc440287033)

[Monica 7](#_Toc440287034)

[Rosen 7](#_Toc440287035)

[Preslav 7](#_Toc440287036)

# Introduction

Our group consists of four members: Rosen Danev, Preslav Gerchev, Dimitar Vikentiev and Monica Stoica, students of class EI6S3.

The present document presents the development of our Network system software in the following chapters:

* Rules and decisions section describes the decisions we have made during this project
* Division of work showing which task were carried as a group and which were individual assignments
* Problems we encountered and how we solved them
* Changes made since our final version of URS and Design documents
* Individual part representing personal point of view on the assignment and group work

# Rules and decisions

To begin with, we decided that the user has a limited working space. This working space i designated by a non-resizable panel. When an element has to be drawn on the screen, the user clicks on a point on the screen and the system will calculate the centre of the element (picture) and draw it. The components are represented by pictures while the pipelines are drawn as a lines.

When drawing one of the components on the screen we decided that the system registers the X and Y coordinates as the top-left corner of the figure. However, this does not apply for pipelines as well.

We did not implement an algorithm for overlapping lines because a line can be drawn by selecting multiple points on the screen (creating a path). Therefore, the user is responsible if the lines will overlap or not.

To select a component, the user has to double click on it and a red rectangle is drawn around it.

By default, all elements have a current flow and a capacity of 0.

Both adjustable and normal splitter have a default value of 50% on both outputs.

To adjust the percentage of an adjustable splitter, the user double clicks on the drawn splitter and a track bar is displayed.

The current and the maximum flow of any element cannot be negative numbers and the current flow must be a lower or an equal number to the maximum flow. Otherwise, the system will not register the changes. The flow is registered as a double and not integer because, for example, if we want to connect pipeline with a flow of 7 to a normal splitter, we will have the out coming flow of 3 and 3. Therefore, 1 part will be lost.

The pipeline flow is shown on the middle of the line.

The file is saved in binary format (.XML) and the user does not have the option to change this.

# Division of work

Most of the tasks have been divided between group members and we supported each other whenever we had issues with the code or with writing the documentation. Below you can find a brief description of our work distribution:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Nr | Task | Group | Dimitar | Monica | Preslav | Rosen |
|  | User Requirements Specifications |  |  |  |  |  |
|  | Design document |  |  |  |  |  |
|  | Process report |  |  |  |  |  |
|  | User interface |  |  |  |  |  |
|  | Class implementations |  |  |  |  |  |
|  | Serialization |  |  |  |  |  |
|  | Functionality of the form |  |  |  |  |  |

# Problems and solutions

One of the problems that we encountered was deleting a line. So that a line will be deleted, it has to be selected first. We tried drawing the line in a rectangle but this would work only if the line is perfectly straight. Therefore, we are using Graphics Path – given a set of points, it can detect whether a mouse click occurred inside or not. When selected, the colour of the pipeline changes.

To continue with, another issues related to pipelines was getting the right text location with information about the current flow. The impediment was that a pipeline can have either two points (start and end) or more (including the points in between).

The solution we found was to check first if there are any in between points. If so, we can take the middle one from the list and return it as a point for drawing the text. If there are no points in between, we will calculate the middle based on the starting and ending point.

Connecting a pipelines to components created an issue as well because the pipeline must connect the starting points and not any others. For this reason, we set predefined locations for connecting pipelines. For pumps and sinks we chose the starting and ending side (right and left respectively). However, for the rest of the components it was more difficult because, for splitters and mergers, we have three starting/ending points.

**Changes**

Since we submitted the last version of the User Requirements Specifications and Design documents, we have made the following changes:

* Updated use cases
* User interface
* Class diagram and description of the members
* Updated the rules and decisions section
* Sequence diagrams

# Individual part

### Dimitar

Monica

To begin with, I believe that the tasks should have been better and equally distributed. However, the group work was strong as well as communication. Whenever we disagreed on something we made sure to explain our point of view and compromise. The deadlines were respected and everyone tried to do their best in spite the fact that we had a quite busy period.

As far as the subject of the project is concerned, I do not personally think it was the best approach. The theme, Pipeline network, was not very attractive. Designing a very good user interface and user friendly was not an option because it would have taken too much time which we spent on implementing the functionality.

Overall, I believe that it was a successful project and the task were completed as required.

### Rosen

The idea of the assignment was good. Coding-wise it made me understand and study new things. According to the flow of the assignment, the deadlines and the meetings with the teacher every week showed me how a project should be done. Following it I learn how to work with deadlines, work in team and be more organized. In the documentation part we used what we’ve learned from the previous course of OOD which is good because you get to use what you’ve learnt. Overall, the flow of the project, the documentation and the work done was good. However, it would’ve been more interesting if we put the time spent and the effort to build something that can be actually used by someone (project from a company or online project and etc.) but I guess that comes in future.

### Preslav

Overall I really enjoyed working on this project. It presented a couple of problems that I had lots of fun trying to solve and optimize. The best part was that we got a chance to use everything we’ve learned so far – OOP principles, quality code conventions, etc. At first I thought the idea itself – a flow network was not too interesting but as the project went on it became more interesting and more interesting to me.

The part I liked the most was coding – I prefer it to documentation. However doing use cases and sequence diagrams was very useful .They helped me get a grasp of how a real project is done – from start to end.

To sum it up – it was lots of fun and I am looking forward to the next project.