|  |
| --- |
| Fontys university of applied sciences |
| User Requirements Specifications |
| Version II |
|  |
|  |
| **11/22/2015** |

|  |
| --- |
|  |

*Rosen Danev*

*Preslav Gerchev*

*Dimitar Vikentiev*

*Monica Stoica*

Table of Contents

[Introduction 2](#_Toc435961584)

[Functional requirements (use-cases) 3](#_Toc435961585)

[User Interface 6](#_Toc435961586)

[Nonfunctional requirements 7](#_Toc435961587)

# Introduction

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

The following document follows the development of an object-oriented software product by using UML techniques.

The goal of this software system is to build a flow network consisting of pipelines and components such as pump, sink, splitter, adjustable splitter and merger.In addition, the User Requirements Specification (URS) will be described such as functional and non-functional requirements and user interface.

The functional requirements will be analyzed using use-cases and used in determining the most appropriate user interface.

# Functional requirements (use-cases)

Below is given a list of use cases our group has created for the system we will build.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**I.**

**Goal:** Create an element.

**Pre:** There must be at least one element placed on the bar.

**Actors**: User

**MSS:**

1. The actor selects the element he/she wishes to create by clicking on it.
2. The actor clicks somewhere on the screen that is given for the flow network.
3. The element is added to a list that holds all the elements.
4. The system creates the element on the screen that is given for the flow network.

**Extensions:**

2a: There is existing element.

1. The system displays a warning.
2. The actor is returned at MSS-step 1.

II.

**Goal:** Adjust the percentage of an adjustable splitter.

**Pre:** There must be at least one adjustable splitter placed on the screen.

**Actors**: User

**MSS:**

1. The actor selects the splitter.
2. The actor clicks the Adjust percentage button.
3. The system displays a text box.
4. The actor enters a new value.
5. The actor confirms by clicking the Confirm button.
6. The system saves the changes.

**Extensions:**

5a: The actor enters a value above 100 or below 0.

1. The system displays a warning.
2. He is returned at MSS-step 3.

**III.**

**Goal:** Open an existing file

**Pre:**

**Actors**: User

**MSS:**

1. The actor presses the ‘Load file’ button.
2. The system displays a dialog box.
3. The actor presses the browse button and selects the file.
4. The actor confirms by clicking the Open button.
5. The system closes the dialog box.
6. The system loads the file.
7. The system displays all the information from the file.

**Extensions:**

**1. a) The system displays a dialog box asking the actor if he/she wishes to save the current file -> if yes, use case save file**

3a: The actor presses the ‘Cancel’ button and exits the use case

5a: The file is not in the correct form

1. The system displays a warning.
2. The actor is returned at MSS-step 3.

IV.

**Goal:** Save as a file

**Pre:**

**Actors**: User

**MSS:**

1. The actor presses the ‘Save as’ button.
2. The system displays a dialog box.
3. The actor chooses a location.
4. The actor chooses a name for the file.
5. The actor confirms by clicking the Save button.
6. The system saves the file.
7. The system closes the dialog box.

**Extensions:**

**2a. The actor presses the ‘Cancel’ button and he/she exits the use case**

4a: There is already a file with that name

1. The system displays a warning.
2. The actor is returned at MSS-step 2

**V**

**Goal:** Save a file

**Pre:**

**Actors**: User

**MSS:**

1. The actor presses the ‘Save’ button.
2. The system displays a dialog box informing the actor that the save is done

**Extensions:**

**1.a ) The file has no location or name on the disk-> continue with save as, step 2**

VI.

**Goal:** Delete an element.

**Pre:** There must be at least one element placed on the screen that is given for the flow network.

**Actors**: User

**MSS:**

1. The actor selects the element he wishes to delete by clicking on it.
2. The actor presses the Delete button.
3. The system deletes the element from the screen that is given for the flow network.

**Extensions:**

***4a: There is already a file with that name***

1. ***The system displays a warning.***
2. ***The actor is returned at MSS-step 4.***

VII.

**Goal**: Change the current flow of a pump.

**Pre**: There must be at least one pump placed on the screen.

**MSS:**

1. The actor selects the pump for which he/she wants to change the current flow.
2. The actor writes in the desired value in a textbox for the current flow.
3. The actor presses the confirm button.
4. The system changes the pump’s flow.

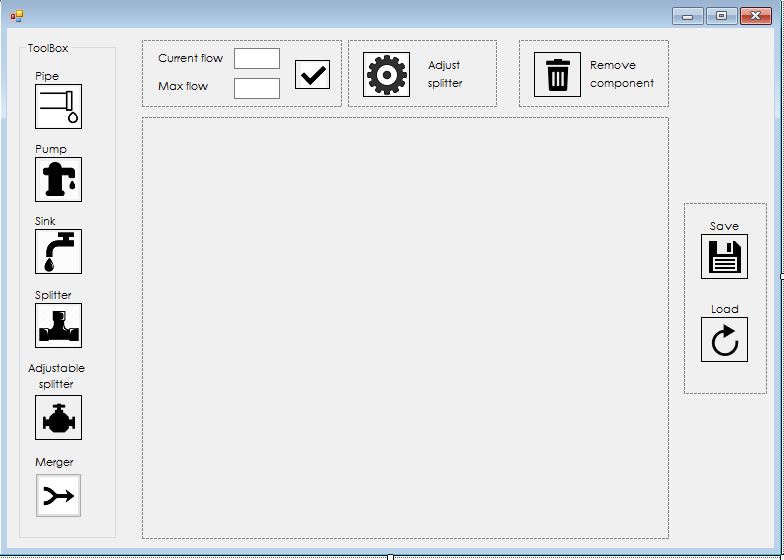
**Extensions:**

3a: The value is bigger than the max flow.

1. The system displays a warning.
2. The actor is returned at MSS-step 2.

# User Interface

Below is given the design that we will be using for our application. We have tried to design the application in such a way that everything is self-explanatory.



# Nonfunctional requirements

The software system will include the following nonfunctional requirements:

* *The max flow must be bigger than the current flow*
* *A splitter has one pipeline as input and two pipelines as output*
* *Half of the incoming fuel leaves the splitter via the upper output and half of it via the lower output*
* *The percentage fuel that leaves the splitter via the upper output*
* *A merger has two inputs and one output*
* *A sink has one input and no outputs*
* *Every pipeline has safety limits*
* *A pipeline starts at an output of a component and ends at an input of another component*
* *The pipeline has a currently flow of fuel*
* *The current flow of a pipeline will be shown*

To begin with, the software system will have a user-friendly interface so that all users, no matter how experienced they are, would be able to create a simple pipeline flow. The system will insure maintainability by being able to add and remove components, recoverability by saving and loading a file, capacity by adding as many elements as wanted and reliability by assuring the user that the application will not crash at unexpected times.