# User Requirements specification

GROUP A

Student name: Student number  
Daniel Todorov [2480417]  
Georgi Chishirkov [2530090]  
Ilia Nikushev [2551721]  
Yu Hong [2598515]

Date : 26.11.2015

Version: 1

Contents

[User Requirements specification 1](#_Toc436064686)

[Introduction 2](#_Toc436064687)

[Functional Requirements ( use cases ) 2](#_Toc436064688)

[User Interface 4](#_Toc436064689)

[Non Functional Requirements 6](#_Toc436064690)

## Introduction

The network flow system allows the users to plan out a pipeline to see measure how the flow would be distributed. The user is provided with various elements that would replicate real life objects to control the flow on the pipeline.

This document will introduce you to the requirements of this system in order for it to provide an easy and intuitive way to structure the pipeline.

## Functional Requirements (use cases )

**Draw any element into the drawing panel**

1. User clicks on any of the elements from the toolbox
2. The system highlights the element clicked
3. The user left clicks inside the drawing panel
4. The system checks if the area is free
5. The element is placed on the drawing panel by the system

Alternatives:

3.1 The user attempts to draw outside the panel – nothing happens

4.1 The user attempts to draw on top of another element – nothing happens

**Remove a placed element**

1. The user selects the remove tool
2. The system highlights the tool
3. The user clicks on an element inside the drawing panel
4. The system removes that element and any connected pipes to it

Alternatives:

3.1 The user clicks outside or not on a element – nothing happens

**Remove a pipeline**

1. The user selects the remove tool
2. The system highlights the tool
3. The user hovers the mouse close to the pipeline
4. The system highlights it in red
5. The user clicks on it
6. The system removes the pipeline

Alternatives:

3.1 The user hovers the mouse over 2 overlapping pipelines – the system highlights the last pipeline added

**Select any element inside the drawing panel**

1. User clicks on the select tool from the toolbox
2. The system highlights the tool in the toolbox
3. The user selects an element
4. The system shows the current flow and max flow for the element

Alternatives:

3.1 The user selects anything outside the drawing panel – nothing happens

3.2 The user selects a pipe – nothing happens

4.1 The user has selected an adjustable splitter - a dropdown menu appears where you can adjust the percentages of the outgoing flow

**Move any element inside the drawing panel**

1. The user clicks on the select tool on the left hand side of the program
2. The system highlights the tool selected
3. The user then drags any element he wishes to move
4. The system sets the final position when the dragging has finished as well as the new position for the pipelines

Alternatives:

3.1 The user attempts to drag outside the panel – the element remains within

3.2 The user attempts to drag on top of another element – the element is reverted back to its last known position

**Connect to elements together**

1. The user clicks on the pipeline element button
2. The user hovers on an element’s connectable point
3. The system highlights the element’s point as available
4. The user clicks on an element’s connectable point
5. The system begins the pipeline
6. The system highlights this point as in use
7. The user repeats 3,4 to connect to an end point
8. The system connects the two points with a straight line

Alternatives

3.1The element has an active pipeline at that point – The system highlights the point as unavailable  
 4.1The element has an active pipeline at that point – The system does not allow this pipeline  
 4.2The user clicks outside of the pipeline point – Nothing happens  
 7.1The pipeline results in a circular network - the point is marked as unavailable

**Extra: Undo**

1. The user clicks the undo button
2. The system removes the last change done
3. The system registers to the redo stack this change

Alternatives

1.1Nothing to undo – the button is disabled, un-clickable

**Extra: Redo**

1. The user clicks the redo button
2. The system adds the last change un-done
3. The system registers to the undo stack this change

Alternatives

1.1Nothing to redo – the button is disabled, un-clickable

**Save a pipeline**

1. User clicks the save button
2. The system opens the file dialog
3. The user gives a name and a location and clicks save
4. The system stores the file as .pipeline

Alternatives

3.1The user specifies a file that exists in that directory – System checks if they want to overwrite  
 3.2The user specifies an invalid file name – System informs of the oddity  
 4.1The user’s operating system does not allow the save to occur – System informs the user

**Load a pipeline**

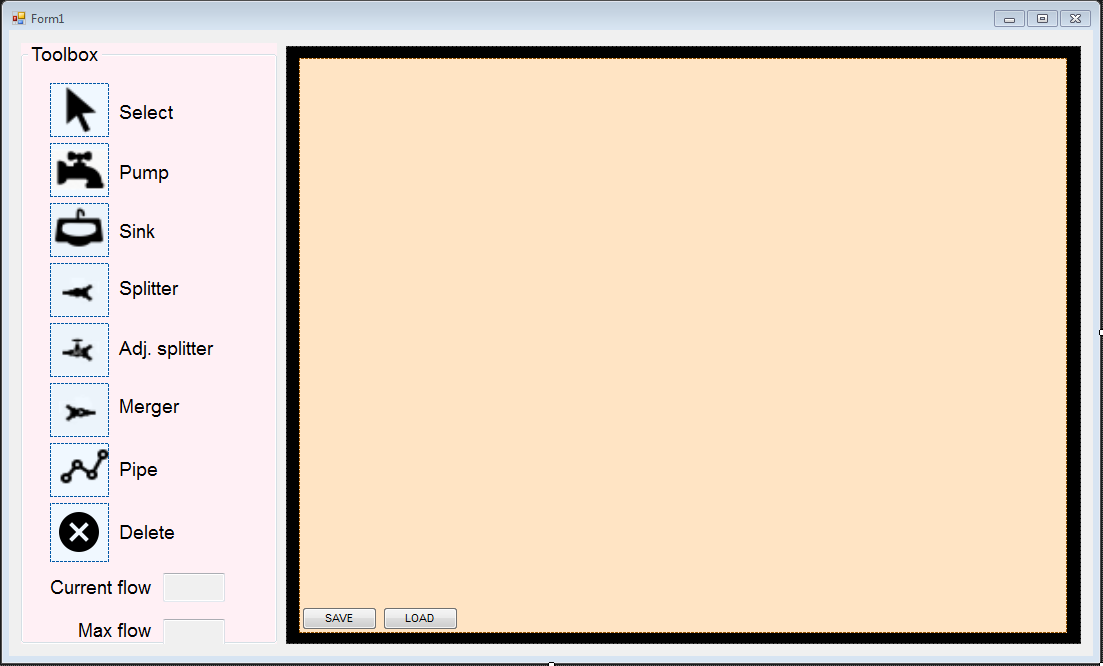
1. User clicks the load button
2. The system opens the file dialog
3. User selects a .pipeline to load
4. System loads the pipeline

Alternatives

1.1 User has unsaved changes – User is prompted if they want to save  
 3.1 User has selected a file that is corrupt or invalid – The user is informed of the oddity

## 

## User Interface

This is our design for the project

## Non Functional Requirements

These are the key features that will outline our projects main quality standards. At run time the 3 main things we are aiming for will be usability, stability and performance. Our program will be easy and intuitive to use, as long as the user has some basic knowledge about flow diagrams. Lastly we will focus on its overall performance and try to ensure everything will run as fast as possible.

The program also focuses on extensibility in the overall structure of the system and the code itself. Meaning that after launch, if we so desire new features will be easily added without having to redesign or recode the program.