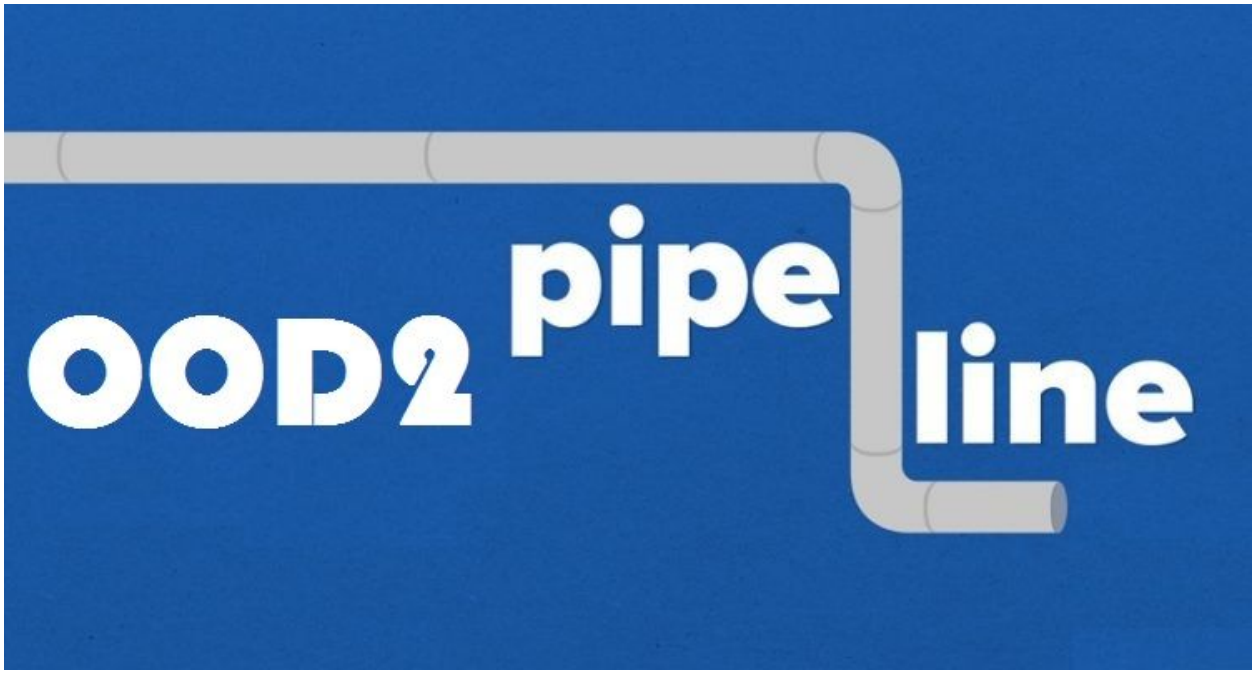


User requirement specifications

Date

12 - June - 2017.

OOD2 group 1

Team (Names and student number)

Hoang Linh - 2233495

Fei Pei -2585413

Simon Onumajuru - 2727897

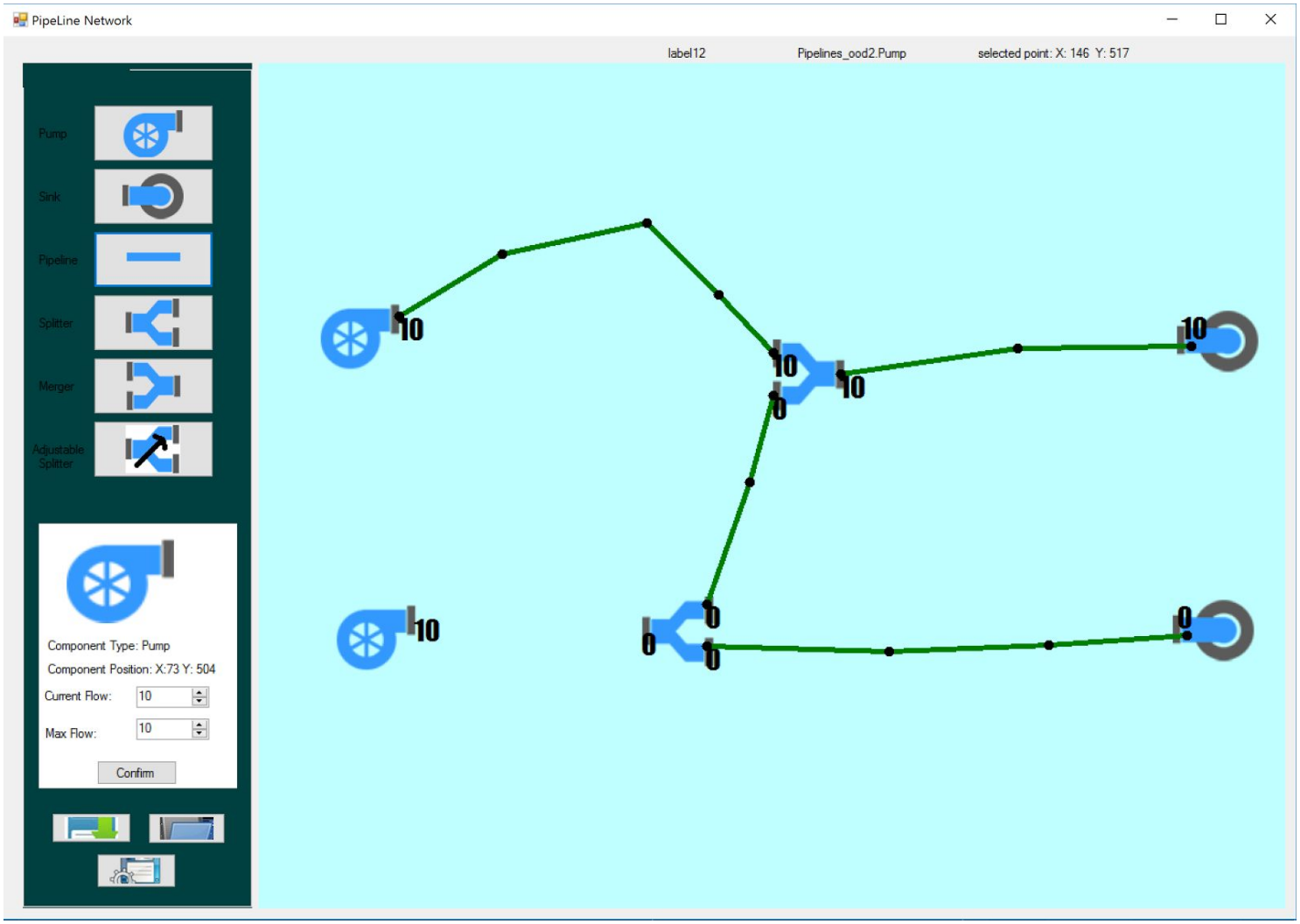
Xiankuan Peng - 2624109

Status

Final version URS.

Table of Contents.

User interface:



The User Interface consist of:

Main panel: a light aqua board whereby user can place several objects selected from the tool box. The user can select any object and change its property in the below numeric dropdown values.

Toolbox: consist of 6 components, 2 numeric dropdown values where the user can edit, modify and change the values of the flows and pipelines for the network. Also contains 3 buttons for Save, Load, and Export.

1. Introduction

1.1. Product overview

In this project, we are going to develop an application to build a flow network consisting of pipelines and components to transport fuel from one place to the other.

1.2. Product overview

The application provides a simulation where the flow of pipeline network is displayed, it also provides possibilities to add, remove, edit virtual pumps, sinks, splitters, and mergers for the user in order to show the correlation and connections of these components and the flow of the fuel.

The network will consist of the followings components.

pump, splitter, adjustable, merger, sink, pipeline.

· **Pump** pumps fuel into a pipeline. A pump has a certain capacity. The capacity is the maximum amount of fuel that can leave the pump. It also has a currently flow (the current amount of fuel leaving the pump every time-unit).

· **Splitter** has one pipeline as input and two pipelines as output. Always half of the incoming fuel leaves the splitter via the upper output and half of it via the lower output

· **Adjustable splitter** it is possible to adjust the percentage fuel that leaves the splitter via the upper output (and the rest leaves via the lower output, of course),

· **Merger** has two inputs and one output. It merges the incoming fuel together.

· **Sink** is a destination for the fuel. A sink has one input and no outputs.

· **Pipeline** starts at an output of a component and ends at an input of another component. A pipeline has a currently flow of fuel. For safety reasons every pipeline has a safety limit. If the current flow is exceeding the safety limit it might be dangerous. For every pipeline you must be able to show its current flow of fuel through the pipeline.

1.3. Scope

The project focuses on establishing the pipelines network that meets the defined requirements written on the workbook.

1.4. Users

The user of the application is: Our Client (Mr Bert, any user)

Requirements summary

Application:	Use case	Must have	Should have	Could have	Will not have
Functional Requirements					
Add new component	UC-01	✓			
Add new pipeline	UC-02	✓			
Remove component	UC-03	✓			
Change flow percentage of adjustable splitter	UC-04	✓			
Remove a pipeline	UC-05	✓			
Change a current flow of pump	UC-06	✓			
Saving the project	UC-07	✓			
Loading the prjoect	UC-08	✓			
Export the network (PNG)	UC-09	✓			
Save or Save as	UC-10	✓			
New project	UC-11	✓			
Close Project	UC-12	✓			
Testing				✓	
Non-Functional Requirements					
Runtime errors				✓	
Performance		✓			

2. Requirements defined

2.1. Functional Requirements

2.1.1 Must have:

- **ID: FR-01**: It must be possible to add components to a flow network.
- **ID: FR-02**: It is not allowed to have components overlapping each other.
- **ID: FR-03**: It must be possible to place pipelines between components. A pipeline should be between the output of a component and the input of another component and it has some in-between-points.
- **ID: FR-04**: Every input of a component can be connected with 0 or 1 pipeline.
- **ID: FR-05**: Every output of a component can be connected with 0 or 1 pipeline.
- **ID: FR-06**: it must be possible to remove components. When a component is removed, all pipelines connected to that component should be removed too.
- **ID: FR-07**: It must be possible to remove pipelines.
- **ID: FR-08**: It must be possible to change the current flow of a pump. Of course, the current flow cannot exceed its capacity.
- **ID: FR-09**: It must be possible to change the percentage of fuel leaving by the upper output of an adjustable splitter. Of course, this percentage is at least 0% and at most 100%.
- **ID: FR-10**: It must be possible to see the current flow through every pipeline. You should warn the user if it exceeds the safety limit (for instance by using colours: a colour for a safe flow and another colour for the a critical flow (a flow bigger that the safety limit)).
- **ID: FR-11**: For every sink it must be possible to see how much fuel will be transported to this sink.
- **ID: FR-12**: It must be possible to store a flow network and it must be possible to use a stored flow network again.

2.1.2 Should have:

- **ID: FR-13**: User interface for data inputs.

2.1.3 Could have:

- **ID: FR-14**: Improved friendly user interface (tools, pipelines, background image with nice graphics).
- **ID: FR-15**: Pipelines do not overlap each other.

2.1.4 Will not have:

- **ID: FR-16**: Graphical animations.
- **ID: FR-17**: Zoom-in, zoom-out functions.
- **ID: FR-18**: Multi-platform application.

2.2. Non-Functional Requirements

2.2.1 Must have:

- **ID: NFR-01**: The system will not have runtime errors.
- **ID:NFR-02:** Performance of the system.

2.2.2 Should have:

- **ID: NFR-03**: The codes are easy to understand with meaningful comments.
- Will be written in C#.

2.2.3 Could have:

- Testing.

2.2.4 Will not have:

- **[N/a]**

2.3. Platform

The application can run on Windows 8 or higher.

Minimal screen size of the device should be 15” or higher.

3. Use cases:

Name: add new component

ID: UC-01

Actor: user

Precondition: Actor already selected one of the components

Main success scenario:

1. The actor clicks anywhere on the canvas.
2. System checks if the click was on blank area and it is.
3. System creates the selected component and draws it on the canvas.

Exceptions:

2.1. There's 1 component on the selected area on the canvas.

- System informs user with a messagebox and ask user to select another area.

Postcondition:

The component is placed on the canvas.

Name: add new pipeline

ID: UC-02

Actor: user

Precondition: Actor already have the application running (Canvas and toolbox panel are displayed) and there are at least 2 sinks to be connected.

Main success scenario:

1. The actor clicks on pipeline box on the toolbox panel.
2. Actor enters the current/max flow of the pipeline
3. Actor validtates the flow choice
4. System checks the validation of the max flow
5. Actor clicks (draws) components on the canvas panel.
6. System checks if there's any component on the canvas.
7. System checks if the component is available to be connected.
8. The actor clicks on a different component on the canvas panel.
9. System checks if the component is available to be connected.
10. The New pipeline connection is added.
11. End use case.

Exceptions:

3.1. If the input is not valid, system notifies the actor

5.1. There's 0 component on the canvas.

- System informs user with a messagebox and asks user to select another component.

6.1 The component has already connected by another flow.

- Inform user with a messagebox and acquires user to redo.

7.1 The component has already connected by another flow.

- Inform user with a messagebox and asks user to redo.

Postcondition:

The 2 components are connected by a flow, capacity of the flow is shown.

Name: remove a component

ID: UC-03

Actor: user

Precondition: the program is running and there are component(s) on the canvas screen.

Main success scenario:

1. Actor clicks on the component.
2. Actor right clicks on the component to remove the component.
3. System displays an option to delete
4. Actor clicks on delete
5. System check if there are associated pipeline(s)
6. System delete the associated pipelines
7. System delete the component
8. End of use case.

Exceptions:

- 5.1. if there is no pipeline associated with it, go to 7

Postcondition:

selected component has been deleted

Name: change flow percentage of adjustable splitter

ID: UC-04

Actor: user

Precondition: App is running and a adjustable splitter is already selected on the canvas screen

Main success scenario:

1. Actor right click on the adjustable splitter.
2. System display option with properties
3. Actor click on properties and 'changes percentage' from the list and confirms.
4. System shows a window with current percentage of each flow.
5. Actor type in new percentage of one flow.
6. System check if new percentage exceed the limit.
7. System calculate and show the percentage of of the other flow.
8. Actor click on 'confirm' button.
9. System stores the flow.
10. System saves the changes.
11. Window closes.

Exceptions:

- 6.1 if exceed, keep the original number and alert the actor the new value is invalid
- 8.1. if actor click cancel button, go to 11

Postcondition:

- Percentage of the splitter has been changed.

Name: remove a pipeline

ID: UC-05

Actor: user

Precondition: Actor have the app running.

Main success scenario:

1.Actor selects a pipeline connected to components.

2.Actor right clicks on a pipeline connected with components

3.System display an option menu with delete.

2.Actor clicks on delete.

3.System removes the pipeline.

4.System updates the flow.

5.End use case.

Exceptions:

Postcondition:

Pipeline removed.

Name: change a current flow of a pump

ID: UC-06

Actor: user

Precondition: Actor has the app running and a pump is added on the flow system and the current selected function is none.

Main success scenario:

1. Actor clicks on the pump.
2. Actor selects the percentage flow from the toolbox.
3. Actor changes the flow value in the configuration panel and confirms the change.
4. System check if the value exceeds the pump's capacity and it has not.
5. System changes the flow in that pump and inform the Actor.

Exceptions:

1.1. If there is no pump where the mouse is clicked, the system does nothing and informs the actor with a message box no pump was selected.

5.1. If the value exceeds the pump's capacity,System will inform the Actor,and the value will not be changed.

Postcondition:

The flow of that pump is changed.

Name: Saving the project

ID: UC-07

Actor: User

Precondition: A project is already started

Main success scenario:

1.The actor clicks on save application.

2. The system checks for prior savings if exist or is new.

4. The system saves the project.

5. End of use case.

Extension:

2.1 - Jumps to UC-10

Exceptions:

System shows informations.

1.0 - System displays a message box, with choice choose folder to save.

File name already exist.

2.1 - System ask user if he wants to overwrite and confirm?

2.2 - User either confirms or cancels and back to MSS 5.

Postcondition:

Project is saved.

Name: Loading the project

ID: UC-08

Actor: User

Precondition: A project is not started

Main success scenario:

1.The actor clicks on load project(a previously saved project).

2.The system displays currently saved projects.

3.The actor selects the project he wants to open.

4.The system then loads the project.

4. End of use case.

Extensions:

3.1. If the actor tries opening the same project system does nothing.

3.2. If actor choose new project then system displays the new project.

Exceptions:

User unintentionally clicks the close button.

3.1 - System ask user to cancel or continue.

Postcondition:

Project loaded.

Name: Export the network (Png)

ID: UC-09

Actor: User

Precondition: A project is started

Main success scenario:

1.The actor clicks on the button export to a file.

2.The system displays dialog box.

3. The actor chooses type of export choice.

4. The system displays folders or location to choose for export.
5. The actor selects choice folder.
6. End of use case.

Extensions:

2.1 - System shows user with available folder locations for saving.
2.2 - User selects export location.
2.3 - System saves to selected location.

Exceptions:

File name already exist.
5.1 - System ask user if he wants to overwrite and confirm?
5.2 - User either confirms, or modifies the name or cancels.

Postcondition:

File exported.

Name: Save or Save As(Conitnuation of UC-07)

ID: UC-10

Actor: User

Precondition: A project is started

Main success scenario:

1.Previously from UC-07 system checking for prior savings if exist or is new.

Extension:

1 - If save file exits.
2 - If save file does not exist.

Exceptions:

1.1 - Then system asks user to choose another name or overite.
2.1 - System provides user with choice to create new save file.

Postcondition:

File saved.

Name: New project

ID: UC-11

Actor: User

Precondition: A project is about to be started

Main success scenario:

1.Actor clicks on a new project to start.

2.Systems launches the application.

3.System displays the window with the canvas, and toolbox panels.

4. End use case.

Exceptions:

Postcondition:

New project started.

Name: Close project

ID: UC-12

Actor: User

Precondition: A project is already started

Main success scenario:

1.Actor clicks the close button on the right most part of the application.

2.System requires actor to confirm closing.

3.Actor confirms.

2.System closes the application.

3. End use case.

Exceptions:

2.1 - System requires actor choose to save as or cancel.

2.2 - If actor choose to save as jumps back to UC-10.

2.3 - if actor choses cancel nothing is saved and falls back to MSS 3.

Postcondition:

Project closed.