Pipe Lines Design Application

|  |  |
| --- | --- |
| **Document:** | URS |
|  |  |
| **Authors:** | Zisis Damianidis |
|  | Tsanko Hadzhiev |
|  | Richard Dyer |
|  |
|  |  |
|  |  |
|  |  |
| **Creation Date:** | 16/11/16 |
| **Last Revised:** | 16/11/16 |
| **Group Name:** | Tanks & Co.™ |
| **Version:** | 0.0 |

## Contents

[Contents 2](#_Toc462721078)

[INTRODUCTION 3](#_Toc462721079)

[Purpose of URS 3](#_Toc462721080)

[Index 3](#_Toc462721081)

[PRODUCT DESCRIPTION 4](#_Toc462721082)

[Background Information 4](#_Toc462721083)

[Performance 4](#_Toc462721084)

[Users 4](#_Toc462721085)

[Assumptions 5](#_Toc462721086)

[REQUIREMENTS 5](#_Toc462721087)

[Functional requirements 7](#_Toc462721088)

[Non-Functional requirements 7](#_Toc462721089)

[Use cases 8](#_Toc462721090)

[Adding crops 8](#_Toc462721091)

[Removing crops 9](#_Toc462721092)

[Updating water resources 9](#_Toc462721093)

[Updating fertilizers to crops 10](#_Toc462721094)

[Simulating growth of crop: 10](#_Toc462721095)

[Retrieving profit/loss report 10](#_Toc462721096)

[Exit application 11](#_Toc462721097)

[Save statistics 12](#_Toc462721098)

[APPROVALS 12](#_Toc462721099)

[Sign-off Sheet 12](#_Toc462721100)

# INTRODUCTION

## Purpose of URS

This document is the definitive specification of the user requirements for Pipe Lines Design Application Project to be developed by Tanks & Co.™ The application simulates the design of a pipeline using a pump, slink, and adjustable splitters and mergers.

The URS defines the functional requirements and non-function requirements. This document works as a base between the client and designers on how the application will be designed and implemented so that both parties can agree on a shared vision for the functions of look of the application.

## Index

This part of the document will serve as an explanation of the terminology that will be used throughout the document the client may not be familiar with.

**URS** – User Requirements Specification. Refers to this document which specifies what the user expects the application to be able to do.

**GUI** – Graphical User Interface. A type of User Interface that allows users to interact with the application through graphical icons and visual indicators such as secondary notation, instead of text-based user interfaces, typed command labels or text navigation.

**MoSCoW** – **M**ust have, **S**hould have, **Co**uld have and **W**ill not have. This method is a prioritization technique used to reach a common understanding with the client on the importance that is placed on the delivery of each requirement.

**Input** – In this document is referred to the configurations the user has assigned to the application.

**Actor** – Specifies a role played by a user or any other system that interacts with the application in the use case.

**Functional Requirement** – Defines a function of the application or its component. A function is described as a set of inputs, the behavior, and outputs.

**Non-Functional Requirement** – A requirement that specifies criteria that can be used to judge the operation of the application, rather than specific behaviors. It is contrasted with Functional Requirements that define specific behavior or functions.

**Global Variable** – Indicates configurations that will affect the whole current simulation.

**Use Case** – A list of actions or event steps, defining the interactions between the client and the application, to achieve a goal.

**Sea-Level** – Level of detail of the use cases. At Sea-Level one deals with users and how their goals are achieved.

**Pre-condition** – Prerequisites needed before the Use Case can be initiated.

**Trigger – Method of initiating the Use Case.**

**MSS – Main Success Scenario. Used to describe the Use Cases of the application and their primary way of completion.**

**Extension** – Used to describe deviations from the Main Success Scenario of a Use Case during a certain step and the alternative ways of executing it.

# PRODUCT DESCRIPTION

## Background Information

The application simulates the design of a pipeline using a pump, slink, and adjustable splitters and mergers.

The user will be able to place these components anywhere on the flow network and the simulation will calculate the capacity of each pipeline. Each pipeline must have a pump and a sink that will show how much fuel is being transported through the pipeline. The user will be able to add, remove, and adjust the flow of splitters, the flow of a pump, and the safety limit of the system. A pipeline design can be saved and loaded.

## Users

This application can be used an individual or team to create a base design for a pipeline system which would include the flow of that network.

## Assumptions

Following are some assumptions for this project made by Tank&Co.™:

* Pipeline will be placed between components
* Pipeline will start with a Pump and end with a Sink.
* The input of a component can be connected with 0 or 1 pipeline with the expcetion of a pump (0 pipelines) and a merger ( 1 > pipelines)
* A merger will be able to have up to 3 pipelines as an input.
* The output of each component can be connected with 0 or 1 pipelines with the exception of a sink ( 0 pipelines) and a spilitter( 0 – 2 pipelines)
* The Splitters output will be adjustable from 0% to 100%.
* Every pipeline will display its current flow.
* Each sink component wll show how much fuel has been transported to the Sink.
* Component could be replaced with another component.

## Constraints

* The application will be created in C# Visual Studio.
* Components will not be overlapping
* The flow of a pipeline cannot exceed its source capacity.

## REQUIREMENTS

In the table below you can find the MOSCOW for every requirement during the project.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | Requirement name | Must | Should | Could | Will Not |
| 101 | Add Component | V |  |  |  |
| 102 | Remove Component | V |  |  |  |
| 103 | Add Pipeline | V |  |  |  |
| 104 | Remove Pipeline | V |  |  |  |
| 105 | Save Pipeline Network | V |  |  |  |
| 106 | Load Pipeline Network | V |  |  |  |
| 107 | Change Capacity |  | V |  |  |
| 108 | Set Network Safety Limit | V |  |  |  |
| 109 | Adjust Splitter output |  | V |  |  |
| 110 | Show Current Pipeline Flow | V |  |  |  |
| 111 | Exit Application | V |  |  |  |
| 112 | Replace Component |  |  | V |  |
| 113 | Change the flow of a pump. |  | V |  |  |
| 114 | Over Safety Limit Warning | V |  |  |  |

Functional requirements

|  |  |  |
| --- | --- | --- |
| ID | Requirement name | Description |
| 101 | Add Component | Allows user to set a crop in a plot. |
| 102 | Remove Component | Removes a component and connected pipes from the network system. |
| 103 | Add Pipeline | Add a pipeline in between two components. The components cannot have  An exceeded input or output pipe capacity. |
| 104 | Remove Pipeline | Removes a pipeline from a component. |
| 105 | Save Pipeline Network | Saves the current network model to a binary file. |
| 106 | Load Pipeline Network | Loads a previously saved binary file and displays the content of that file into the pipeline network flow application. |
| 107 | Change Capacity | The Capacity of the flow of the pipeline can be increased or decreased. |
| 108 | Set Network Safety Limit | The system will be able to limit the flow capacity. |
| 109 | Adjust Splitter output | One of a splitter component’s output is adjustable from 0% to 100% |
| 110 | Show Current Pipeline Flow | In each pipeline the current flow will be visable to the user. |
| 111 | Exit Application | Before exiting the application, the application will prompt the user to save if nessarry. |
| 112 | Replace Component | A set component will be replaced when a new component is placed over it. |
| 113 | Change Pump Flow | The user will be able to change the flow of a pump. |
| 114 | Over Safety Limit Warning | The user will be warned if safetly limit of a pipeline has been exceeded. |

## Non-Functional requirements

Platform compatibility

* Application should work most optimally in the Windows environment. This application should work best on the Windows platform as it was designed for such.

Usability

* The application should be user friendly and incorporate elements of good user interface design. For example, the buttons are easily recognizable and familiar to the user in terms of expressing what function the button serves easily accessible to the user.
* Program is simplified and allows the user to reach his goal without any problems. User must be able to access the program without registration, account etc.

Performance

* The application should respond within 1500milliseconds with each button click, and 4000 milliseconds when loading a report on a modern machine (Processor greater than 1Ghz).

Reliability

* In case of exception or error, the program displays info messages without crashing, allowing user to continue his work.

## GUI

### GUI Details

## Use cases

All use cases have the system and user as the only actors involved. Furthermore, all the use cases are of the sea-level.

### 101: Adding Component

**MSS:**

1.    User

**Extensions:**

# APPROVALS

## Sign-off Sheet