City simulation

**User Requirement Specification**

Dimo Popov

Peter Hadzhiev

Zhicheng Yu

Nikolay Kolev

Xiangkuan Peng

[**Document Revision History**](#_gjdgxs) **3**

[**Document Introduction**](#_30j0zll) **4**

[Purpose of This Document](#_1fob9te) 4

[Overview](#_3znysh7) 4

[**Requirements**](#_tyjcwt) **5**

[Functional Requirements](#_3dy6vkm) 5

[Non Functional requirement](#_2s8eyo1) 6

[**Use Cases**](#_17dp8vu) **7**

[Use case 1.1](#_3rdcrjn) 7

[Use case 1.2](#_1ci93xb) 7

[Use case 1.3](#_3whwml4) 8

[Use case 1.4](#_2bn6wsx) 8

[Use case 1.5](#_qsh70q) 8

[Use case 1.6](#_3as4poj) 9

[Use case 1.7](#_1pxezwc) 9

[Use case 1.8](#_49x2ik5) 10

[**GUI**](#_2jxsxqh) **11**

[**Other**](#_2xcytpi) **12**

# Document Revision History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Version | Status | Description | Date | Name |
| V1.0 | C | Create initial URS | 18/2/2018 | Zhicheng Yu |
| V2.0 | A | Add use case | 3/8/2018 | Zhicheng Yu  Xiangkuan Peng |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

\*Change status: C means Create, A means Add, M means Modify, D means Delete.

\* The version number of the document is compiled from 0.1, and the document version number starts from 1.0. When a small change to the document, the version number to 0.1 advanced; large change version number to 1.0 advanced.

# Document Introduction

## Purpose of This Document

This document is a User Requirement Document for use in a project. It describes the requirements specification for the city simulation application which is intended to assist the relevant management or technical staff, the client to understand the requirements needed for the application.

## Overview

This document describes general software constraints of the application. The main focuses of this document is on the specific requirements for the application and followed by some use cases that will be implemented in the application.

## 

# Requirements

## Functional Requirements

Must Require:

1.1) User must be able to place building, roads and different objects on the application.

1.2) Objects must not overlapping each other.

1.3) User must be able to remove objects.

Should Require

1.1) An exception should be thrown when user uses wrong object.

Could Require

1.1) It could have some explanatory note.

Wont Require

1.1)

Constraints

1.1)

## 

## 

## Non Functional requirement

Must Require

1.1) The application must have no bug at run time.

1.2) The interface of this application should be as user-friendly as possible.

1.3) The output must be correct.

1.4) The application must be windows application in C#.

Should Require

1.1) This application should be programmed in an object oriented to be easily extendable

1.2) The application should be clear for user.

1.3) The source code should be kept clear for future use.

1.4) We should attach a comment to every method to explain what a method does.

1.5) This application should run on windows.

Could Require

None

Wont Require

1.1) We do not care if user open multiple application at the same time.

1.2) We do not care when user want to close application.

# Use Cases

## **Use case 1.1**

**Name**: Adding an object  
**Actor**: A user (employee in the electric company).  
**Pre-condition**: /

**MSS**:

1. User selects an object in the toolbox.
2. User clicks on a fixed location for the object on the grid.
3. System sets the object in selected location on the grid.

**Extension:**

**2a**. If user tries to click on a location with an existing object, system will not allow the placement and will not create the object.

## **Use case 1.2**

**Name**: Select a created object  
**Actor**: user  
**Pre-condition**: /

**MSS**:

1. User clicks somewhere on the map.
2. System checks if there is an object and there is.
3. System selects that object.
4. System shows the info of the selected object.

**Extension:**

**2a.** If there is no object where user is clicking, end of use case.

## **Use case 1.3**

**Name**: Open a configuration panel of a created object  
**Actor**: user  
**Pre-condition**: Use case 1.2 Select a created object

**MSS**:

1. User clicks somewhere on the map.
2. System checks if there is an object and there is.
3. System selects that object.
4. System shows configuration panel of the selected object.
5. User makes changes to the parameters of the object
6. User clicks on save button
7. System saves changes
8. System closes configuration panel

**Extension:**

**2a.** If there is no object where user is clicking, end of use case.

## **Use case 1.4**

**Name**: Delete selected object from map  
**Actor**: user  
**Pre-condition**: Use case 1.2 Select a created object

**MSS**:

1. User clicks somewhere on the map.
2. System checks if there is an object and there is.
3. System selects that object.
4. System shows configuration panel of the selected object.
5. User clicks delete button on panel
6. System deletes the object from the map
7. System closes configuration panel

**Extension:**

**2a.** If there is no object where user is clicking, end of use case.

## **Use case 1.5**

**Name**: Save map configuration  
**Actor**: user  
**Pre-condition**: Use case 1.1 Adding an object

**MSS**:

1. User clicks on ‘save configuration’ button
2. System checks if there is an object placed on the map.
3. System asks where to save file
4. User chooses location
5. User chooses file name
6. User clicks save
7. System saves map configuration in file

**Extension:**

**2a.** If there is no object on the map, error message pops, end of use case.

**5a.** If the file name is in incorrect format an error message is shown and system goes to **5**.

## **Use case 1.6**

**Name**: Weather setting  
**Actor**: user  
**Pre-condition**: Use case 1.1 Adding an object

**MSS**:

1. User clicks on weather type in weather toolbox
2. System changes the weather according to request
3. System changes traffic behaviour according to weather.

**Extension:**

**2a.** If the selected weather type is already been simulated nothing is changed.

**3a.** If the selected weather type is already been simulated nothing is changed.

## **Use case 1.7**

**Name**: Setting time frame  
**Actor**: user  
**Pre-condition**: Use case 1.1 Adding an object

**MSS**:

1. User presses up/down arrow on keyboard or clicks up/down buttons in GUI
2. System checks which button is pressed
3. System changes how time passes according to user input

**Extension:**

**2a.** If user did not press up/down arrow on keyboard or click up/down buttons in GUI, then end of use case

# 

# 

# 

# 

# 

# GUI

The Graphical User Interface will be presented in 2D model. The application will contains several sections: Toolbox, Grid Area and Menu. They will always be provided in the interface. Toolbox will provide some items that can be interacted and used to fill in the Grid Area. The Grid Area will be the stage where the object being placed and displayed.

The toolbox includes the images as well as the names on the button as components able to be added into the Grid Area, in order to reduce any confusion.The menu button is located below of the grid as it is separates itself from the toolbox.

# 

# 

# 

# 

# **Other**

Development Tool

This application will be developed using C#

Language Choices

This application will be presented in English

Future choices

Possible changes for the use case