SRS for Taffic Control International

# Introduction

This document specifies the requirements for an intelligent traffic control system developed for *Traffic Control International Inc. (TLI).*  
  
Purpose

The purpose of this document is to present a detailed description of the project “Traffic Control International Inc. (TLI).”. It will explain the purpose, features and the interfaces of the project, what the project will do and the constraints under which it must operate. This document is intended for users of the software and also potential developers. The project is based on the assignment of Fontys University of Applied Sciences. The result of group 2 is presented here.

The task is about isolated control of traffic lights, coordinated control of traffic lights (intersections) and intelligent traffic control (coordinated control of intersections, exchange of information for cars and traffic lights).

The task is to develop software that manages the traffic light system. It is important that the system is easily extensible. The system should work for several countries, for this it is important to note that there are different signal cycles in the different countries.

## Scope:

Due to time constraints this project will be limited to the implementation of a single light control system and a simple pedestrian crossing. Other types of crossings, basic control and intelligent control are not within scope of this project.

# Data Dictionary:

|  |  |
| --- | --- |
| Traffic light | A single traffic light having a shape and a behaviour. |
| Traffic light behaviour | The behaviour of how and when a traffic light changes signals. |
| Traffic light shape | The shape in which the traffic light signals are displayed. |
| Road | The area between two sidewalks which traffic participants use. |

# References:

Canvas:

**Traffic Control International: SCRUM Introduction, Case Introduction**

[**https://canvas.fontys.nl/courses/11006/modules**](https://canvas.fontys.nl/courses/11006/modules)

# Overall description

### User groups:

* Traffic participant: All people who make use of the road, whether by car or by walking.
* Traffic light: The main entity which the system controls.

### Product Perspective:

The project is intended to represent the traffic light system for Germany and the Netherlands. In addition, it should be easy to add further countries and their country-specific processes. Therefore, an easy and uncomplicated extensibility is necessary. The system represents an intersection and lets the traffic lights represented by our system depend on the country. Thereby the logic behind an intersection must be considered, so that no accidents can happen.

### Requirements:

* Focus on maintainability and extensibility (usage of design patterns)
* Stepwise feature extension by an agile approach with weekly sprints and customer demonstrations
* 1. Control of a single pedestrian light showing red and green

1. Typical pedestrian light behaviour: RED -> GREEN -> RED
2. Extended pedestrian light behaviour: RED -> GREEN -> GREEN BLINKING -> RED

- 2. Control of a single traffic light showing red, green and yellow

### User stories:

**German Pedestrian Light:**

* As a traffic light operator I want to be able to change signal (Red, Green), so that the pedestrian knows when to cross.

**Dutch Pedestrian Light:**

* As a traffic light operator I want to be able to change signal (Red, Green Blinking or Green), so that the pedestrian knows when to cross.

**German Traffic Light:**

* As a traffic light operator I want to be able to change signal(RED, RED-YELLOW, GREEN or YELLOW), so that the driver knows what to do.

**Dutch Traffic Light:**

* As a traffic light operator I want to be able to change signal(RED, GREEN or YELLOW), so that the driver knows what to do.

### Use case diagram:

### Use case descriptions:

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text

Description automatically generatedGraphical user interface, text

Description automatically generatedGraphical user interface, text, application

Description automatically generatedGraphical user interface, text

Description automatically generatedGraphical user interface, text, application

Description automatically generated

### Test scenarios:

Graphical user interface, text, application

Description automatically generated

Text

Description automatically generated

## Requirements for the implementation

A prerequisite for the implementation is the use of patterns.

The following patterns should be used:

* State design pattern
* Observer pattern
* Factory Pattern

### Benefits of using State Design Pattern:

It provides flexibility because you don't know the future requirement changes.

### Benefits of using Observer Pattern:

Defines a one-to-many dependency between objects so that when one object changes state, all of its dependents are notified and updated automatically.

### Benefits of using Factory Pattern:

Good testability ?

### Test Case Scenarios:

| **Name:** | **German pedestrian light test** |
| --- | --- |
| Scenario: | 1. Light is displaying red  2. The light updates  3. Light displays green  4. The light updates  5. The light is displaying red |
| Result: | The German pedestrian light is able to switch its signal after a specific amount of time |
| Extensions: | - |

| **Name:** | **German Traffic light test** |
| --- | --- |
| Scenario: | 1. Light is red.  2. Light updates  3. Light is red-yellow.  4. Light updates  5. Light is green.  6. Light updates  7. Light is yellow  8. Light updates.  9. Light is red. |
| Result: | The German traffic light is able to cycle trough signals. |
| Extensions: | - |

| **Name:** | **Dutch Traffic light test** |
| --- | --- |
| Scenario: | 1. Light is red.  2. Light updates  3. Light is green.  4. Light updates  5. Light is yellow  6. Light updates.  7. Light is red. |
| Result: | The German traffic light is able to cycle trough signals. |
| Extensions: | - |
|  |  |

| **Name:** | **console interface test** |
| --- | --- |
| Scenario: | 1. System displays: Choose what kind of function to display?  a. traffic light b. pedestrian light c. Pedestrian crossing d. intersection 2. Actor inputs option (a, b, c, d) 3. system displays: Choose what kind of behaviour to display:  a. Dutch b. German 4.actor chooses option (a or b) 5. system displays: Choose what kind of mode to display:  a. arrow b. night c. emergency d. normal  6. Actor chooses option (a, b, c, or d)  7. system displays what the actor chose to be displayed (e.g.: German traffic light behaviour with normal mode) |
| Result: | The system is able to display cycle of the chosen behaviour by country and mode for traffic lights, pedestrian lights, crossing and intersection. |
| Extensions: | - |

## Design Constraints:

**Visualization of the intersection**

Display the intersection in the console (terminal) – no GUI

On the intersection there are 4 traffic lights for the road and 8 traffic lights for pedestrians, where two pedestrian traffic lights represent a transition.