Software Requirements

Traffic control international inc.

Name | Titel des Kurses | Datum

# Introduction

Traffic Control International inc. will revolutionize the traffic control market. By moving from selling physical traffic lights to selling services for already exiting traffic lights. This document gives an overview over the different specifications from the traffic control management system, that we sell. It gives an overview over the different functions and helps our developers to know exactly what to implement and what to look out for.

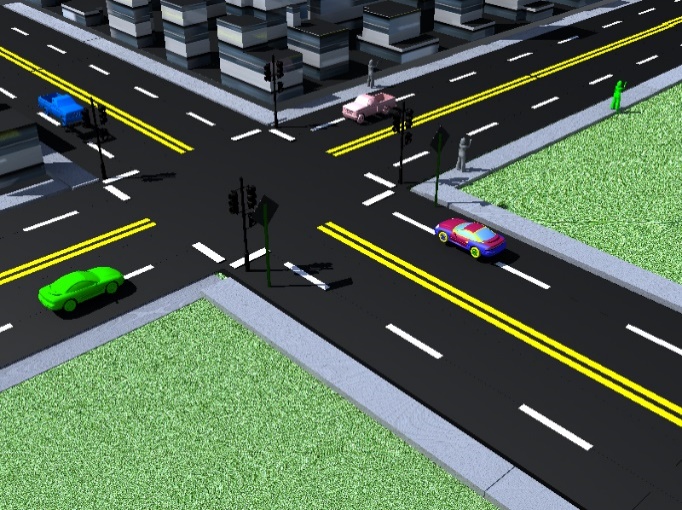
## Purpose

Traffic Control International Inc. (TLI) wants to develop a traffic lights management system for intersections. This project is about developing such software. The revolution in the traffic light market TLI is aiming for, lays using already in place equipment and implementing behaviour over multiple intersections.

## Scope

This project is about delivering a software to manage traffic lights at an intersection. All the possible sorts of traffic light present on an intersection, should get managed by the system. But it’s not part of the project to apply its functionalities to the intersection’s hardware. The Netherlands and Germany are the targeted markets for the system. Therefore, the system should apply a traffic light logic of both countries. Other countries should be able to be added. The intersection consists of two different traffic light types. One for cars and one for pedestrians. Both should not be able to allow passing at the same time on the same lane. A lane should be able to determine how much traffic there is. So, the intersection can adjust timing accordingly. The intersections should be able to communicate with each other to implement behaviour over multiple intersections. Because the communication to the hardware is not in the scope, a UI should be created to present the system to the customer.

## DefInitions and acronyms

The yellow lines are displaying the two lanes of the intersection. In this document they are referenced as lanes.

The section where the pedestrians cross the street are referenced as crossings.

MVP – Minimal viable product

## Overview

The remainder of this document includes three chapters. The second one provides an overview over the systems functionality and system interactions with other systems. The chapter mentions constraints, assumptions, and functions of the product as well. In the chapter the skills that are needed to use the system are mentioned as well.

The third chapter informs about the user interfaces and specific requirements that shall be met by the system.

## references

A prioritized list of requirements provided by the customer is used as a guideline during development. This list can be found under the following Link:

https://connect.fontys.nl/instituten/fhtenl\_studies/studies/INF/PRJ3/StudyMaterial/PRJ3-Case-Traffic-Control-International.pdf

# Overall Description

The following chapter will give an overall description about the product and the constraints.

## Product perspective

## Product functions

* Set traffic light behaviour for specific date, time, or situation.
* Set pedestrian light behaviour for specific date, time, or situation.
* Intelligent traffic lights control.

The mentioned functions are to be executed in various situations:

* Set green light for pederstrians when button is pressed.
* Set green light wave for car drivers on rush hour.
* Set green light wave for special vehicles (ambulance/police/president).
* Set emergency mode in emergency situations.
* Set emergency mode for events (exhibitions/festivals).
* Crossings communicate with cars and each other and adjust signalling times/signalling sequence accordingly.
* Crossings detects cars and adjust signalling accordingly (intelligent night mode – single approaching car).

## Constraints

The following chapter will give an overview about system constraints

### Design Constraints

* The system shall be designed with respect to the applicable laws (EU laws, Dutch and German laws and any other individual country laws).
* The system should be **extendable and easily maintainable.**

### Software Constraints

* CamelCase naming convention shall be used for naming variables, classes, interfaces etc.
* The system shall be written in Java Programming Language.
* The system may have a user interface or may accept input via console, the developers shall decide.
* The business logic of the system shall be tested with unit tests.
* Mocking code packages shall be used for independent business logic testing.

### Hardware Constraints

The system shall be a simulation of an intelligent traffic lights control system; therefore, **no external hardware is required**.

* The system shall be runnable on most used operation systems. (Windows, Linux, Mac).

### Interfaces to other applications

Project management applications:

* GitHub – source control and versioning.
* Jira Agile board – tasks, tasks timing and tasks priority.
* Fraankly – Retrospective
* Planning Poker Online

## Assumptions and Dependencies

We assume several things during developing the software:

* The pedestrians shall be able to control the traffic light system by pressing a button which requests that green light is displayed for them.
* Traffic lights systems shall be able to detect traffic and adjust their behaviour accordingly.
  1. User Characterestics

The only user of the system is the planner. He works for the city and is responsible for traffic management. The system in the end enables him to do so.

# Specific Requirements

In the following chapter the external interfaces and functions are described.

* 1. External Interfaces

After the project is finished the software will be connected to physical lights. The software should be able to be displayed on any computer system.

* 1. Functions

The MVP according to the roadmap contains of the sections A and B. This means the customer wants at least functioning lights and a functioning lighting cycle. A traffic light should have the three working states (Red, Yellow and Green) and shall be open for extension. It can be changed and adapted after deployment of the software. In order to handle traffic, the intersection must be able to implement a lighting cycle. The lighting cycle switches the states of the different traffic lights. The cycle handles pedestrians crossing the street and opposing lanes getting the same signal. Also, the intersection should be able to block traffic on all lanes except one. The reasons for that differ, therefore a trigger is needed. In order to fully satisfy the customer’s needs, the end product must have functionalities on top of the already mentioned functionalities in the MVP. All functionalities are mentioned in section 2.2. In order to implement a green wave behaviour, the intersection should be able to communicate with the other intersections.

|  |  |
| --- | --- |
| Name | A car crosses the intersection |
| Actor | Car |
| Description | A car crosses the intersection in any direction it wants. |
| Precondition | The traffic light on the drivers crossing shows red. |
| Scenario | 1. The traffic light signals red. 2. The driver approaches the crossing. 3. The intersection recognizes the car. 4. The intersection checks if other cars are waiting or approaching. 5. The car stops at the crossing. 6. The system switches the lights according to the light cycle. 7. The traffic light switches to green 8. The car drives off. |
| Exception | 4.1. If there are no other cars waiting or approaching. Jump to step 8.  4.2. If there are other cars go on. |
| Result | A car crosses the intersection |

|  |  |
| --- | --- |
| Name | A pedestrian crosses the street |
| Actor | Pedestrian |
| Description | A pedestrian wants to cross the street on an intersection. |
| Precondition | The traffic light for the pedestrian displays red. |
| Scenario | 1. The pedestrian could notify that he wants to cross the street. 2. The intersection takes notice. 3. The system switches the traffic lights according to its cycle. 4. The pedestrians traffic light signals green. 5. The pedestrian crosses the street |
| Exception |  |
| Result | A pedestrian crosses the intersection safely |

|  |  |
| --- | --- |
| Name | An emergency vehicle approaches the intersection |
| Actor | Emergency Vehicle |
| Description | An emergency vehicle approaches a crossing. The intersection should enable a green signal for the emergency vehicle and stop all other traffic |
| Precondition |  |
| Scenario | 1. The emergency vehicle comes near to the intersection. 2. The system takes notice of the approaching vehicle, way before it is near the intersection. 3. The vehicle approaches the intersection. 4. The system enables a green light on the vehicles crossing. It signals red for everyone else. 5. The vehicle passes the intersection. 6. The intersections switch back to the lighting cycle. |
| Exception |  |
| Result | The emergency vehicle crossed safely |

|  |  |
| --- | --- |
| Name | A green wave gets enabled. |
| Actor | The planner |
| Description | The planner selects different intersections, a date, time when the green wave should be implemented. |
| Precondition |  |
| Scenario | 1. The system gives the planner the option to select intersections. 2. He selects a direction in which way the green wave should be implemented. 3. The system saves the way. 4. The planner selects the day and time for the green wave. 5. The system saves the date and time. 6. The planner selects a duration for the green wave. 7. The system saves the duration. 8. The planner activates the green wave. 9. The system activates the green wave on the given date and time. |
| Exception |  |
| Result | A green wave got planned. |

|  |  |
| --- | --- |
| Name | A green wave gets activated |
| Actor | System, intersections |
| Description | The system enables the previously planned green wave. |
| Precondition | The green wave is planned. This means the date, time and duration is set and the green wave is activated. The given date and time is reached. |
| Scenario | 1. The system informs the intersections about the green wave. 2. The intersections adapt their lighting cycle. 3. The system informs the intersections to go back, after the duration is over. |
| Exception |  |
| Result | A green wave got activated. |

|  |  |
| --- | --- |
| Name | Prioritize heavy traffic |
| Actor | System |
| Description | The system detects the waiting cars on a crossing and prioritizes them. |
| Precondition | Multiple cars are waiting at one crossing. |
| Scenario | 1. The system detects a car approaching the crossing 2. The number of cars waiting is above a certain number. 3. The intersection adapts its lighting cycle, so the crossing is next. 4. The cars cross the intersection. 5. The intersection switches back to the old lighting cycle. |
| Exception |  |
| Result | The cars got prioritized. |

|  |  |
| --- | --- |
| Name | Green arrow |
| Actor | Car |
| Description | A car takes a turn with a green arrow. |
| Precondition | The light is red |
| Scenario | 1. The car stops at the crossing. 2. The car waits for three seconds 3. The car turns. |
| Exception |  |
| Result | The car used the green arrow. |

|  |  |
| --- | --- |
| Name | Cycle lights |
| Actor | System |
| Description | The traffic light switches between go and stop state. |
| Precondition | The lights are on the stop state |
| Scenario | 1. The system waits for some kind of trigger. 2. The system switches lights to the next state according to trigger 3. The system recognizes switched lights are on the go state. 4. The system waits a set amount of time 5. The system switches before switched lights to the next state 6. The system recognizes that the switched lights are on the stop state |
| Exception | 3a. The system recognizes the switched lights not on the go state   1. The system waits a set amount of time 2. Return to step 2   6a. The system recognizes the switched lights are not on the stop state   1. The system waits a set amount of time 2. Return to step 5 |
| Result | The system cycled the triggered lights. |

|  |  |
| --- | --- |
| Name | Set night mode |
| Actor | Planner |
| Description | The planner plans and activates the night mode on one intersection. He selects time and duration for the night mode. |
| Precondition | None |
| Scenario | 1. The planner selects an intersection 2. The system gives the planner the option to enable the night mode. 3. The planner sets a timespan. 4. The system saves the timespan. 5. The planner activates the night mode. 6. The system saves this setting. |
| Exception |  |
| Result | Night mode set and activated. |

|  |  |
| --- | --- |
| Name | Enable night mode |
| Actor | Intersection, traffic lights |
| Description | The intersection switches to the night mode. |
| Precondition | The nightmode is set and activated. |
| Scenario | 1. The time for the night mode to activate passes by. 2. The system tells the traffic lights to switch to night mode. 3. The traffic lights start blinking yellow. |
| Exception |  |
| Result | The night mode got activated. |

|  |  |
| --- | --- |
| Name | Add light |
| Actor | Planner |
| Description | The planner adds lights to a crossing on an intersection. |
| Precondition | The planner gets the option to select an intersection. |
| Scenario | 1. The planner selects an intersection. 2. The system displays the intersection. 3. The planner selects a crossing where he wants to add a light. 4. The system adds traffic lights to the crossing. |
| Exception |  |
| Result | The system added a traffic light to the intersection. |

|  |  |
| --- | --- |
| Name | Delete light |
| Actor | Planner |
| Description | The planner deletes a traffic light from a crossing on an intersection. |
| Precondition | The planner gets the option to select an intersection. |
| Scenario | 1. The planner selects an intersection. 2. The system displays the intersection. 3. The planner selects a crossing and selects that he wants to delete the traffic lights. 4. The system gives him the option to either select a pedestrian or car light. 5. The user selects the traffic light. 6. The system deletes the light. |
| Exception |  |
| Result | The planner deleted a traffic light. |

|  |  |
| --- | --- |
| Name | Create a new intersection |
| Actor | Planner |
| Description | The planner adds a new intersection into the system. |
| Precondition | None. |
| Scenario | 1. The system displays an overview over all intersections. 2. The planner wants to add an intersection. 3. The system gives him the option to do so. 4. The planner adds the known neighbouring intersections. 5. The systems save this settings. 6. The planner adds traffic lights to crossings as mentioned in add light step 7. The system saves the changes. |
| Exception |  |
| Result | A new intersection got added. |

|  |  |
| --- | --- |
| Name | Delete Intersection |
| Actor | Planner |
| Description | The planner deletes an intersection. |
| Precondition | None |
| Scenario | 1. The system displays an overview over the different intersections. 2. The user selects an intersection. 3. The system displays the intersection. 4. The user wants to delete the intersection. 5. The system deletes the intersection. |
| Exception |  |
| Result | The selected intersection got deleted. |

## Performance Requirements

* The system should be able to switch between a light without a break.
* The operator changes should be applied direct.

## Design Constraints

In order to be displayed properly on various different computer systems, the end product should be able to run on these different computer systems. The average user should be able to know how the software works, when he looks at the UI.