User Requirements Specification

First Version



Group B

Todor Tsekov (2228777)

Wen Zhang (2351420)

Jiaqi Ni (2217774)

Yidi Wu (2197301)

Kalina Petrova (2221667)

Jianfei Feng (2204529)

Class: EI7s1/s2

Course: Project C-phase

Mentor: Bert Van Gestel

Table of Contents

1. *Introduction------------------------------------------------------------------------3*
2. *Functional Requirements-------------------------------------------------------3*
3. *User Interface---------------------------------------------------------------------9*
4. *Non- functional Requirements------------------------------------------------12*

Introduction

In this project, we have to build traffic-simulation program and the first step is to write the User Requirements Specifications. For this step we will provide the use-cases of all functionality that we can offer, a specification of user interface and also non-functional requirements. The objective of our application is to have less traffic jams.

Our application is supposed to help users to create traffic simulation and handle the traffic. It will allow the users to build simulation with crossings. The user will be able to put crossings in the grid, and the user can watch the cars flow in the street. The user can add start point and destination to see how the car flows in the fast and easy way. The user can set traffic light in the crossing and position options of lights will be provided.

Functional Requirements

The following use cases will show the actions the user can perform within the application. Based on that, we assume the user can create new grids, load existing ones, and manipulate crossings, traffic lights and flow of traffic.

# MoSCoW Table:

|  |  |
| --- | --- |
| Use-cases: | Implementation: |
| Create New Project | Must |
| Load Project | Must |
| Save Project | Must |
| Exit Application | Must |
| Go to Main Menu | Should |
| Undo | Could |
| Redo | Could |
| Reset | Could |
| Play Simulation | Must |
| Pause Simulation | Must |
| Stop Simulation | Must |
| Add Crossing | Must |
| Delete Crossing | Must |
| Change Crossing | Should |
| Rotate Crossing | Should |
| Add Traffic Light | Must |
| Delete Traffic Light | Must |
| Change Traffic Light | Should |
| Add Flow | Must |
| Delete Flow | Must |
| Alter Flow | Should |
| Navigate | Could |
| Save Statistics | Would |
| Open Statistics | Would |

# Use-cases:

1. **Name**: Open

**Goal**: Open a Project

**Actor**: User

**Pre-condition**: System is displaying the application main screen- traffic Advisor and there is a project to be loaded.

**MSS**:

1. User clicks the “Open” button;
2. A file-dialog window will pop-up and user chooses the file which is going to use;
3. User selects a file and clicks “Ok”;
4. System loads the file and displays the project-traffic simulation screen;

**Exception (Extension, Alternatives)**:

4 – a) If the system doesn’t have enough permissions to open the file, it displays a message “Not enough permissions to open this file” and the use case ends.

4 – b) If the system can’t parse the file correctly, it displays a message “This file could not be loaded.” and the use case ends.

**Post-condition**: The project-traffic simulation screen will be ready for the user.

1. **Name**: Save

**Goal**: Save to a file

**Actor**: User

**Pre-condition**: System is displaying project- traffic simulation screen

**MSS**:

1. User chooses “File-Save” option, on the top left corner;
2. System saves the current traffic simulation to a file;

**Exception (Extension, Alternatives):**

2 – a) If the current grid hasn’t been saved before, the system displays a file-dialog where the user needs to choose the folder and the name of the file he wants to save.

2 – b) If the system doesn’t have permissions to save the file, it displays a message “Not enough permissions to save this file” and the use case ends.

**Post-condition**: The file is saved by the system.

1. **Name**: Exit

**Goal**: Close a file

**Actor**: User

**Pre-condition**: System is displaying project- traffic simulation screen

**MSS**:

1. User chooses “File-Close” option, on the top left corner;
2. System asks the user for confirmation;
3. The system exits;

**Exception (Extension, Alternatives):**

2 – a) User clicks ’No’ option, the system will not exit.

2 – b) There are files that have been edited without being saved.

The system asks if you want to save opened files.

User may close file without saving.

**Post-condition**: The file is closed by the system.

1. **Name**: Undo

**Goal**: Undo a step

**Actor**: User

**Pre-condition**: System is displaying project-traffic simulation screen and the user did at least one step

**MSS**:

1. User clicks the “Undo” button;
2. The system shows traffic simulation before the user’s last action on the screen;

**Post-condition**: The grid is the way it was before the last step.

1. **Name**: Redo

**Goal**: Redo a step

**Actor**: User

**Pre-condition**: System is displaying project- traffic simulation screen and the user did at least one undo action

**MSS**:

1. User clicks the “Redo” button;
2. The system shows traffic simulation after the user’s last action on the screen;

**Post-condition**: The grid is the way it was after the last undo.

1. **Name**: Reset

**Goal**: Clear the current traffic simulation

**Actor**: User

**Pre-condition**: System is displaying project-traffic simulation screen

**MSS**:

1. User chooses the “File- Reset” option;
2. System displays a message, asking the user to confirm the action;
3. User confirms the action;
4. The system shows the empty traffic simulation on the screen;

**Exception (Extension, Alternatives)**:

3 – a) The user clicks in the “No” option, the system display traffic simulation the way it was and the use case ends.

**Post-condition**: The grid is empty.

7. **Name**: Go to main screen

**Goal**: Takes the user back to the main screen

**Actor**: User

**Pre-condition**: System is displaying project-traffic simulation screen

**MSS**:

1. User chooses “File- main menu” option, on the top left corner;
2. System closes project- traffic simulation screen and opens main screen-traffic Advisor;

**Exception (Extension, Alternatives):**

2 – a) If the user didn’t save the progress, the system will show a message asking if the user wants to leave without saving. The user can choose “Yes” or “No”.

**Post-condition**: Main screen- traffic Advisor is displayed.

1. **Name:** Pause

**Goal:** Stop the system for a while.

**Actor:** User

**Pre-condition:** System already start working

**MSS:**

1. Actor click Pause button
2. System stop working and stay in present status.

**Exception:** None

1. **Name:** Stop

**Goal:** Stop the system

**Actor:** User

**Pre-condition:** System is in working status.

**MSS:**

1. Actor click stop button
2. System stop working and back to main interface

**Exception:** None

1. **Name:** Play

**Goal:** The system start working

**Actor:** User

**Pre-condition:** Actor open the application

**MSS:**

1. Actor click play button
2. System start working

**Exception:** Some error happened when click play button ------System stop playing and back to main interface.

1. **Name**: Add traffic light

**Goal**: Adds a traffic light to the map

**Actor**: User

**Pre-condition**: System is displaying a new or a loaded project so that the user can modify the map.

**MSS**:

1. User chooses to put a traffic light
2. User drags traffic light icon to matrix
3. User drops the traffic light icon on the matrix
4. System asks user for an input for the intervals/durations for each lane
5. User inputs valid numbers
6. System places traffic light on specified spot

**Exception (Extension, Alternatives)**:

3.1: There already exists a traffic light on that spot

System asks the user if he wants to override current traffic light instead -> Go to use case modify traffic light

3.2: The user has specified a spot without a crossing

System outputs an appropriate message “You cannot place a traffic light there.”

5.1: User doesn’t input values

System creates a traffic light with default values

5.2: User inputs wrong values

System outputs an appropriate message “You have to input values of type integer 1-100”

**Post-condition**: There now exist a traffic light group on specified crossing.

1. **Name**: Delete traffic light

**Goal**: Delete an existing traffic light on the map

**Actor**: User

**Pre-condition**: System is displaying a new or a loaded project so that the user can modify the map and there must already be a traffic light

**MSS**:

1. User right clicks on traffic light
2. User chooses the delete option
3. System deletes the traffic light from this crossing

**Post-condition**: The crossing now has no traffic light on it.

1. **Name**: Modify traffic light

**Goal**: Modify an existing traffic light on the map

**Actor**: User

**Pre-condition**: System is displaying a new or a loaded project so that the user can modify the map and there must already be a traffic light

**MSS**:

1. User right clicks on traffic light
2. User chooses the modify option
3. System asks user for new inputs
4. User writes down inputs
5. System modifies traffic light with new parameters

**Exception (Extension, Alternatives)**:

4.1: User doesn’t input values

System creates a traffic light with default values

4.2: User inputs wrong values

System outputs an appropriate message “You have to input values of type integer 1-100”

**Post-condition**: The selected traffic light has now changed parameters.

1. **Name**: Add crossing

**Goal**: Adds a crossing to our map

**Actor**: User

**Pre-condition**: System is displaying a new or a loaded project so that the user can modify the map.

**MSS**:

1. User chooses a map he wants to place
2. User drags map icon to cell on the map
3. User drops the icon in the cell
4. System asks user if he’d like to put a traffic light

User confirms -> Go to use case add traffic light

User chooses not to put traffic light

1. System places the crossing on the map
2. System reconfigures flow

**Exception (Extension, Alternatives)**:

3.1: User drops the icon on a field that already has a crossing

System asks user if he’d like to change existing crossing

User chooses yes -> Go to use case Change crossing

User chooses no -> System doesn’t change anything

**Post-condition**: We now have a new map with one more crossing in it.

1. **Name**: Delete crossing

**Goal**: Deletes an existing crossing from the map

**Actor**: User

**Pre-condition**: System is displaying a new or a loaded project so that the user can modify the map and there’s already a crossing on the map to delete.

**MSS**:

1. User right clicks on existing crossing on the map.
2. User chooses the delete option
3. System asks the user for confirmation
4. The user confirms
5. System deletes crossing from the map
6. System alters the flow appropriately

**Exception (Extension, Alternatives)**:

**Post-condition**: Our map now has one less crossing.

1. **Name**: Change crossing

**Goal**: Changes an existing crossing on the map

**Actor**: User

**Pre-condition**: System is displaying a new or a loaded project so that the user can modify the map and there’s already a crossing on the map to change.

**MSS**:

1. User right clicks on an existing crossing
2. User chooses the change option
3. System drops down a list with crossings to choose from
4. User chooses a crossing he wants to change to
5. System deletes old crossing -> Go to delete crossing use case
6. System adds the new crossing -> Go to add crossing use case
7. System alters the flow appropriately

**Exception (Extension, Alternatives)**:

**Post-condition**: The map has now a different kind of layout.

1. **Name**: Rotate crossing

**Goal**: Rotates an existing crossing on the map

**Actor**: User

**Pre-condition**: System is displaying a new or a loaded project so that the user can modify the map and there’s already a crossing on the map to rotate.

**MSS**:

1. User right clicks on existing crossing
2. User chooses the rotate option
3. System swaps the properties of the 4 roads with the properties of the one on the right.
4. System rotates the crossing picture(90 degrees counter clockwise)
5. System alters the flow appropriately

**Exception (Extension, Alternatives)**:

**Post-condition**: The map has now a different kind of layout.

1. **Name:** Add Flow

**Goal:** Add cars flow for each directions to a crossing.

**Actor:** User

**Pre-condition:** At least one crossing on grid.

**MSS:**

1. User Right clicks the crossing and choose “Crossing Setting”.
2. System shows crossing setting window.
3. User gives number of cars from each direction. (east, west, south, north)
4. Setting will be update for the crossing and the window will be closed after user click “OK”.

Exception:

3.1 The number cars may limited. User cannot input the number above the limited number.

1. **Name:** Delete Flow

**Goal:** Flow number of cars will be empty. (Number becomes to 0)

**Actor:** User

**Pre-condition:** At least one crossing on grid.

**MSS:**

1. User Right clicks the crossing and choose “Crossing Setting”.
2. System shows crossing setting window.
3. Click “Restore Settings”.
4. The number of flow for each direction will become default ones.
5. Setting will be update for the crossing and the window will be closed after user click “OK”.
6. **Name:** Alter Flow

**Goal:** Alter cars flow for each directions to a crossing.

**Actor:** User

**Pre-condition:** At least one crossing on grid.

**MSS:**

1. User Right clicks the crossing and choose “Crossing Setting”.
2. System shows crossing setting window.
3. User alter number of cars from each direction. (east, west, south, north)
4. Setting will be update for the crossing and the window will be closed after user click “OK”.

**Exception:**

3.1 The number cars may limited. User cannot input the number above the limited number.

1. **Name:** Navigation

**Actor:** User

**Precondition:** None.

**MSS:**

1. User chooses “navigation” option.
2. User sets the starting point on the screen.
3. User sets the destination point on the screen.
4. User confirms the starting point and destination point.
5. System calculates the route and executes the “Add flow” use case for the related roads.

**Extension:**

4.1 User cancels the operation.

**Post-condition:** The system simulates the situation and changes the flow of the related roads correctly.

1. **Name**: Save Statistics

**Goal**: Save the simulation result as a statistical data

**Actor**: User

**Pre-condition**: System is displaying a loaded project that has been run and stopped.

**MSS**:

1. User chooses to export the result as statistical data/document
2. System opens browser pop-up
3. User searches for location to save file
4. User types a name
5. User clicks confirm
6. System saves file

**Exception (Extension, Alternatives)**:

5.1: There already exists a text file with that name

System asks user if he’d like to overwrite it

User can overwrite file

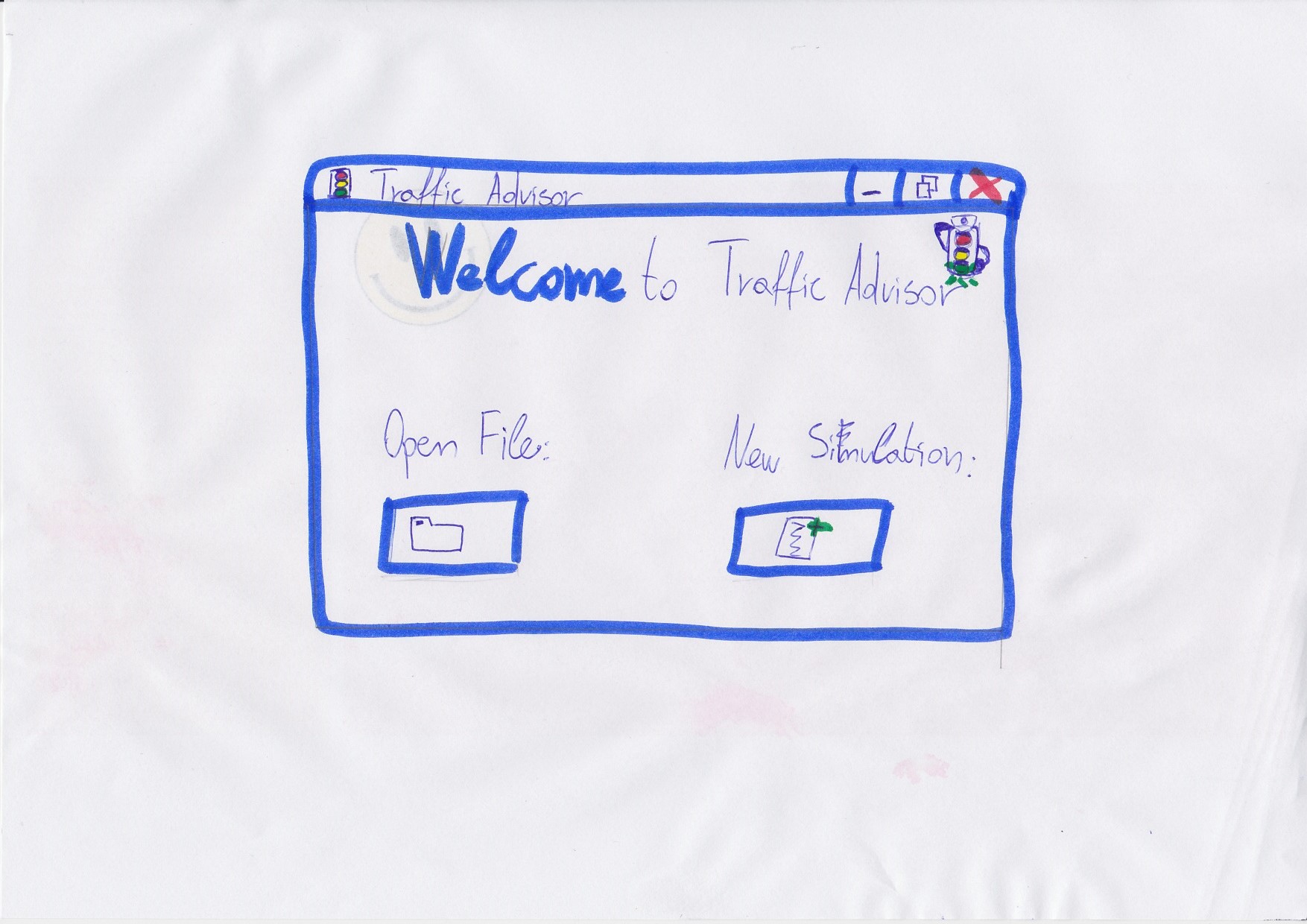
User clicks cancel-goes back to browser to type name

**Post-condition**: There now exists a file with statistical data from our simulation

User Interface

We created a storyboard of the user interface. These are the two screens that we are going to have:

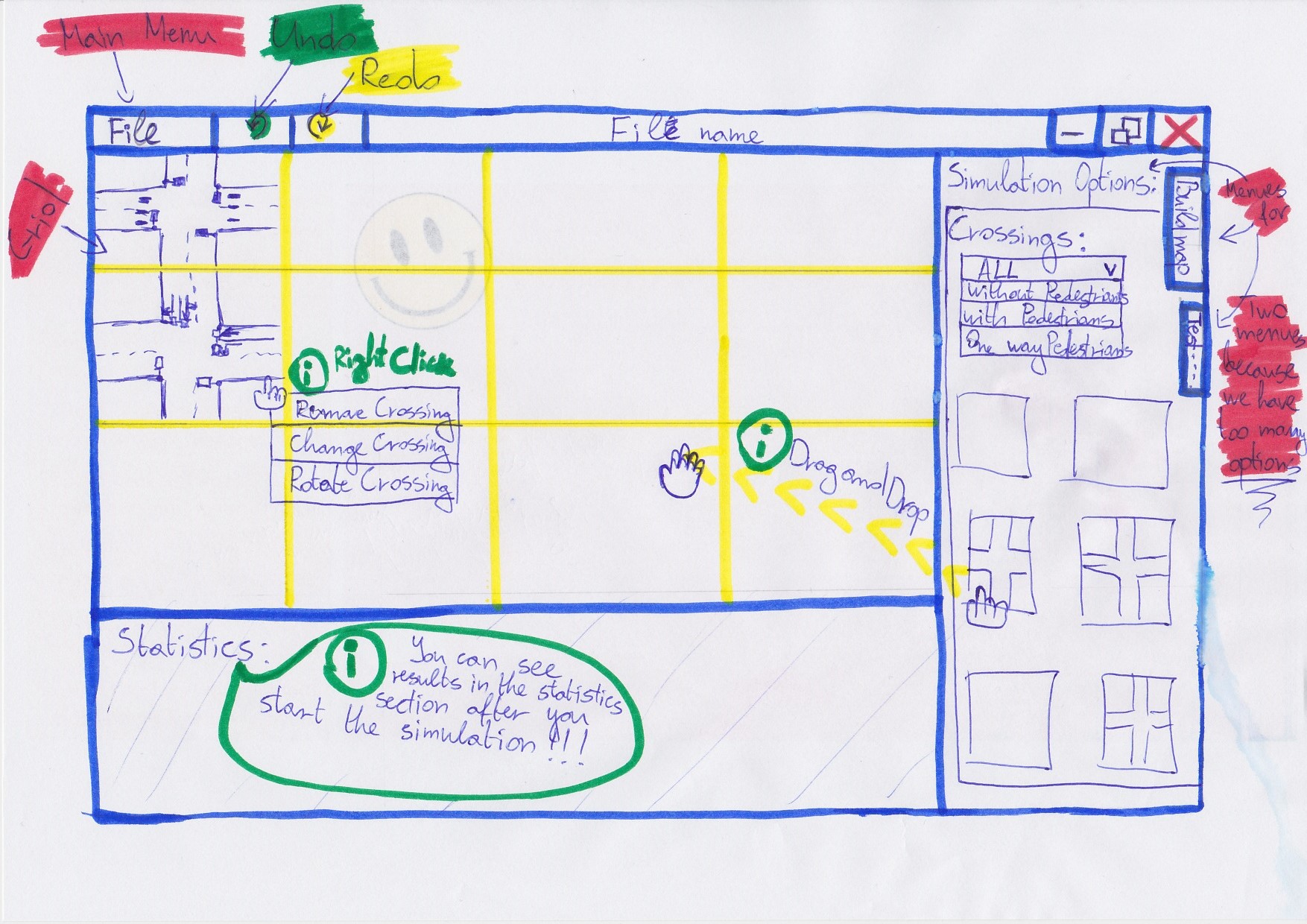
Main screen:



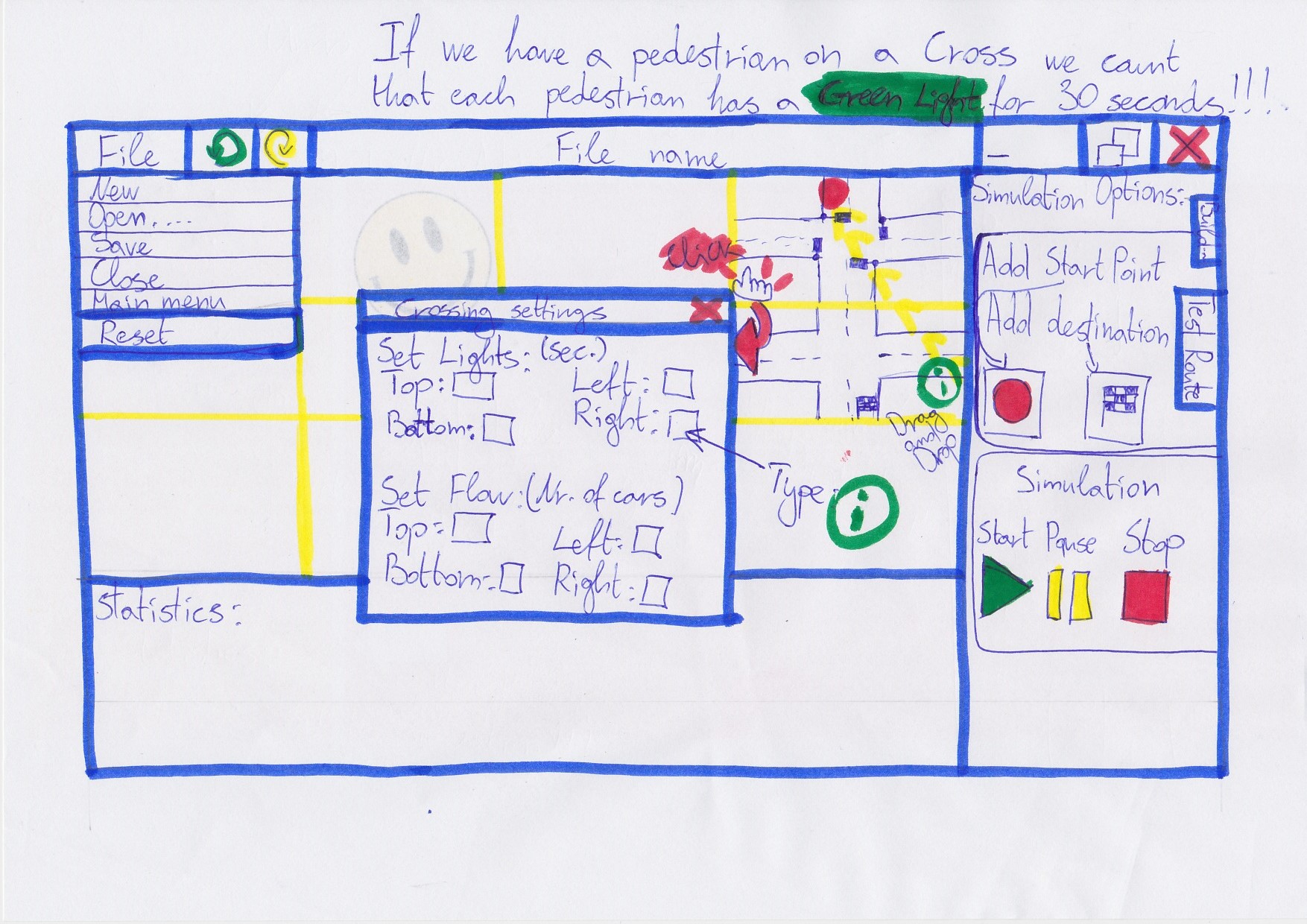
With ‘Open File’ button you can open a simulation file from your PC.

With ‘New Simulation’ button you can start new simulation on your PC.

Project-grid screen:



On this screen is the main functionality of the application, we could choose from a lot of options, so we made another look of this form so you could see the rest of the options.



# Rules:

* When a Cross is rotated all its data will be reset.
* If there is a pedestrian on a Cross the system counts that each pedestrian has a Green Light for 30 seconds.
* Results in the statistics section will appear on the screen after the simulation is started.
* Simulation will not start if a ‘Start Point’ and ‘Destination’ points are not chosen.
* Simulation will not start if there are not at least two Crosses on the grid.
* You could not put Cross on the grid if in that place you already have a placed Cross.
* You could add a Cross to the grid by selecting it by clicking on it and then dragging it through the grid and dropping it inside of it.

Non-functional Requirements

* The application will be programmed in C#, which means Windows operation system version windows 7 or windows 8 are the required environments for it.
* The application can run on a basic computer.
* You do not need Internet to use our application.