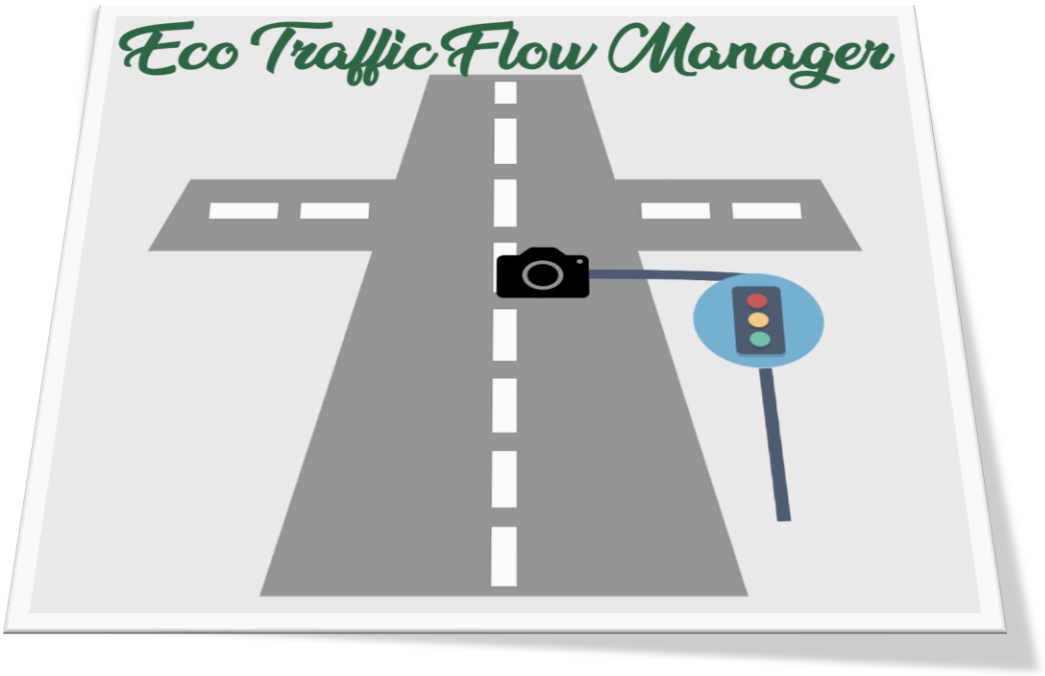


Department of Information Engineering, Computer Science and Mathematics
University Of Study Of L'Aquila, Italy

Submitted to Professor **Davide Di Ruscio**
Course Titled **Software Engineering for Autonomous Systems**

Member	Matricolation N°	Email Address
Enrico Simone Adamelli	282290	enricosimone.adamelli@stuent.univaq.it
Roberto di Stefano	289916	roberto.distefano1@student.univaq.it
Matteo Di Russo-Ciccarelli	291325	matteo.dirussociccarelli@student.univaq.it

1) Introduction

The developed autonomous system represents a solution for traffic light management aimed at minimizing traffic congestion, optimizing fuel consumption and reducing environmental pollution.

Utilizing machine learning techniques, the system employs real-time data analysis and image recognition to detect vehicles within the scenario. This approach is designed to enhance the efficiency of traffic flow, thereby reducing wait times and alleviating queues. The system relies on cameras strategically positioned at each intersection, capturing images that are subsequently used to assess the volume of vehicles present.

In addition to vehicle detection, the system incorporates a responsive feature to address the presence of people waiting at a traffic signal. If someone has activated the corresponding button, the system takes this into consideration, adjusting its operation accordingly. This approach not only optimizes vehicular movement but also prioritizes the safety and convenience of pedestrians at crosswalks.

2) Managed resources:

- **Traffic lights**:
  The system should dynamically assign the status of each traffic light based on the real-time traffic of both vehicles and people.
  The following possible statuses are assigned to groups of traffic lights:
  - **GREEN**:
    There will be at most one group of traffic lights with this status. This state is maintained for a time period dynamically established by the system.
  - **YELLOW**:
    This is an intermediate state between GREEN and RED ones. This state is maintained for a fixed time period.
  - **RED**:
    This is the default state for each traffic light. To prevent starvation this state is maintained for a time period dynamically established by the system such that it will not exceed a certain threshold.
    Upon reaching such threshold, such traffic lights will gain a priority in getting the GREEN status.

  The groups of traffic light are chosen by the cross road designer and could be changed anytime through a configuration file. Each group could be composed by one or more vehicles traffic light and/or pedestrian traffic light. The green status is assigned to the belonging group of the traffic light that optimized the total waiting time.

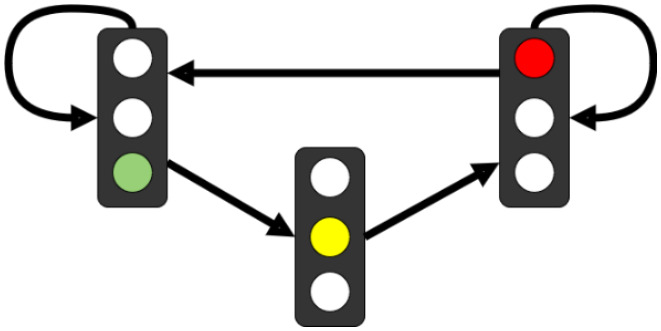


Figure 1: Traffic light phases

➔ **Traffic Switcher:**

The system also uses traffic switchers to change flux of incoming vehicles by turning on/off a signal that recommends drivers to follow a different street. Each switcher is assigned to one or more traffic lights per crossroad. The signal is expected to be before the cross. Moreover the switcher uses a predictor trained with data gathered over time to predict the next time to turn on and minimize the incoming flux on its assigned crossroad.

3) The sensors and the effectors for each managed resource

- **Effectors**:
  - One actuator for each semaphore to switch lights.
  - One actuator for turning on/off the traffic switcher.
- **Sensors**:
  - One camera at the center of each road to take the photos that will be used to count the number of vehicles and evaluate the traffic flux.
  - A sound sensor designed to detect emergencies by monitoring sound levels.
  - A button sensor for each pedestrian's traffic light that when pressed implies people are waiting to cross the road.
  - One humidity sensor that perceives if the road conditions are normal/anormal.

4) Employed architectural pattern for the development of the autonomic manager

Every component uses an MQTT broker to exchange messages relative to gathered and computed data.

1. A monitor that collect the values of all sensors and stores them to the knowledge component.
2. The analyzer that takes input from the knowledge and analyzes images, humidity, button pression and perceived sound to work on the relevant data and define the real time situation. Moreover, the analyzer component can make predictions for the switcher subsystem.
3. A planner that takes input from the analyzer, makes a decision that is turning on/off the traffic switcher and/or change the traffic light's status, and then sends it to the knowledge.
4. The executor that sends a signal to the actuator of the traffic lights to light up and to the traffic switcher to turn ON/OFF.

5) Adaptation goals of the autonomic manager

The adaptation goal is to minimize the traffic at the cross by evaluating the waiting time of vehicles. This purpose is obtained managing two class of resources: group of traffic lights and traffic switcher, each with the same aim but looking at different parameters.

▪ Adaptation logic of traffic light:

At the same time, all the cameras take photo of the associated road and by using mape-k components the system evaluates the reported parameters, according to the following definitions:

- NV = Number of vehicles
- t' = mean time to clear the road for one vehicle in good weather condition (crossing time for one vehicle)
- MV = Max number of vehicles covering the whole width of the road (number of road lanes)
- Starvation is implied when vehicles or pedestrians are stucked on a traffic light which didn't become green recently.

Goal	Description	Evaluation Metric
Crossing time estimation	The system keeps into consideration the parameters NV, t', MV to estimate the crossing time of vehicles with respect to one traffic light and adapts if such time exceeds MAX_THRESHOLD	$t_i = NV/MV * t' <$ MAX_THRESHOLD
Total waiting time minimization	To minimize the waiting time, the system keeps into consideration the t_i of each traffic light and	$t = \max(t_i)$ for $i=1,\ldots,N$

	select the group of traffic lights associated to max t_i.	
Priority assignment	When the perceived sound is higher than a threshold, the system detects an emergency.	perceived_sound < threshold
Starvation avoiding	To prevent starvation on traffic lights, the red time of a single semaphore has to be less than a threshold	red_time_i < RED_THRESHOLD

- **Adaptation logic of traffic switcher**:

The traffic switcher acts when the flux of incoming vehicles grows. It can be also adapted to consider the history of past actions such that it proactively prevents traffic congestion.

- incoming_flux: difference of number of vehicles between two consecutive photos in the same semaphore.

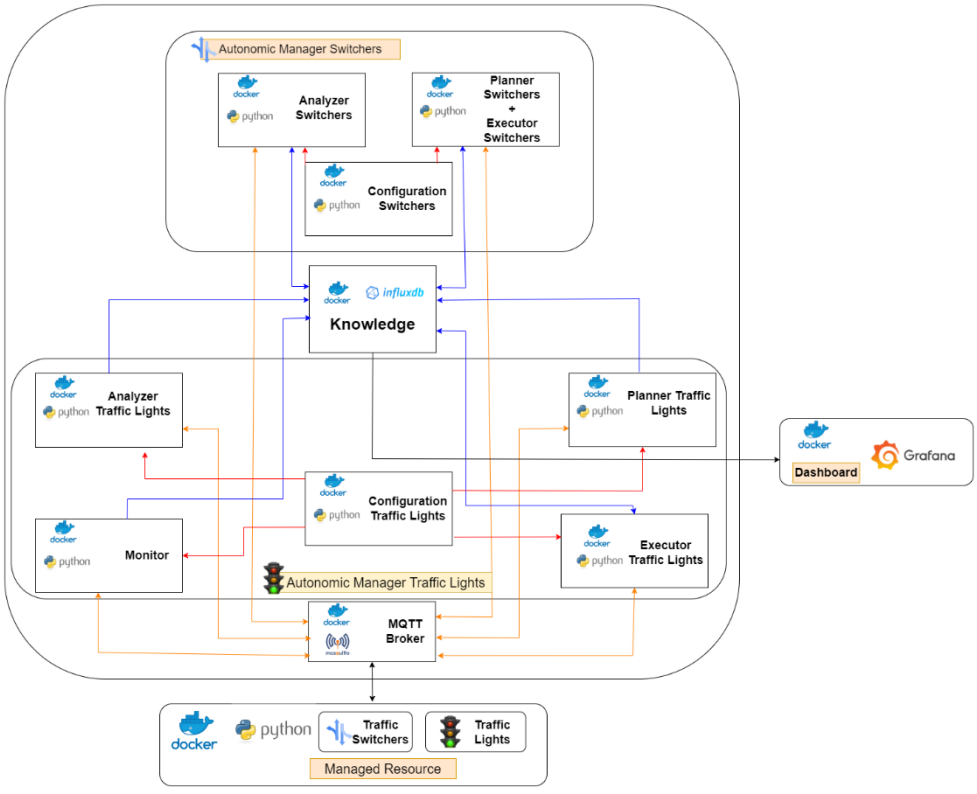
Goal	Description	Evaluation Metric
Traffic congestion minimization	The system keeps into consideration the incoming_flux and acts when such number is greater than 0 to reduce traffic congestion	Incoming_mean_flux ≤ 0

6) Architecture diagram

The system is composed by two autonomic managers one for adapt the traffic lights and one for the traffic switchers each with its own MAPE-K loop. The two subsystem share the same knowledge and MQTT brocker:

- **Autonomic Manager Traffic Light:** is the autonomic manager responsible to adapt the traffic lights of a single crossroad according to the current traffic status, emergency happen and weather.
  The components of this subsystem are: Monitor, Analyzer Traffic Lights, Planner Traffic Lights, Executor Traffic Lights, Knowledge and Configuration Traffic Light

- **Autonomic Manager Switcher:** is the autonomic manager responsible to adapt the traffic switchers according to the mean flux of each traffic light and its prediction. Respect to Autonomic Manager Traffic Light this has both reactive and proactive logic.
  The components of this subsystem are: Monitor, Analyzer , Planner + Executor Switchers, Configuration Switchers



In the diagram the communication between each component is represented by arrows with different color according to their scope:

- → **Communication to/from knowledge**
- → **Communication performed thanks to MQTT broker**
- → **Communication to/from configuration service**
- → **Communication with external service**

7) Implementation details of the system

The system can be divided in two subsystems: traffic light management and traffic switcher management. Both subsystems can be associated with one or more intersections and operate based on data from managed resources. An MQTT Broker is employed for the exchange of data from sensors and other messages among all components. InfluxDB is utilized for storing and retrieving data, while Grafana is implemented to create a dashboard that monitors the behavior of the entire system.

→ Grafana:

To show the data represented the system behavior was used Grafana that is a tool compatible with influx DB useful to query and show in a graph data taken from a database. The main graph represented are:

- Traffic light Status (*Figure 2*)
- Traffic Switcher Status (*Figure 2*)
- Sound trend (*Figure 3*)
- Humidity trend (*Figure 3*)
- Number Vehicles trend (*Figure 4*)
- Traffic Flux trend (*Figure 4*)

Figure 2: in the left side there is the graph relative to status of traffic lights groups compared with emergency while in the right side are represented the traffic switchers status compared with flux threshold. We can show that when emergency happen the status of all the traffic light group is setting to red in this way the crossroad will be free and the ambulance will pass easily. We can show that when the flux overcame the threshold the switchers are turned on for three minutes; the same happens according to the predictions.

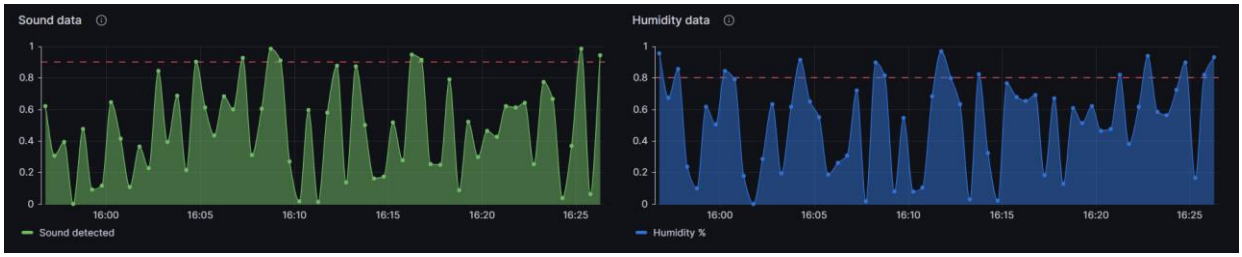


Figure 3: in the left side there is the graph relative to sound sensor value. In the right side there is the graph relative to humidity sensor value. In both graphs there is a line marked with red color that represent the corresponding threshold value (0.9 for sound sensor and 0.8 for the humidity sensor)

Figure 4: on the top there is the graph relative to number of vehicles for each traffic light while at the bottom of the image we can see the graph relative to the mean flux computed for each traffic light.

→ Knowledge:

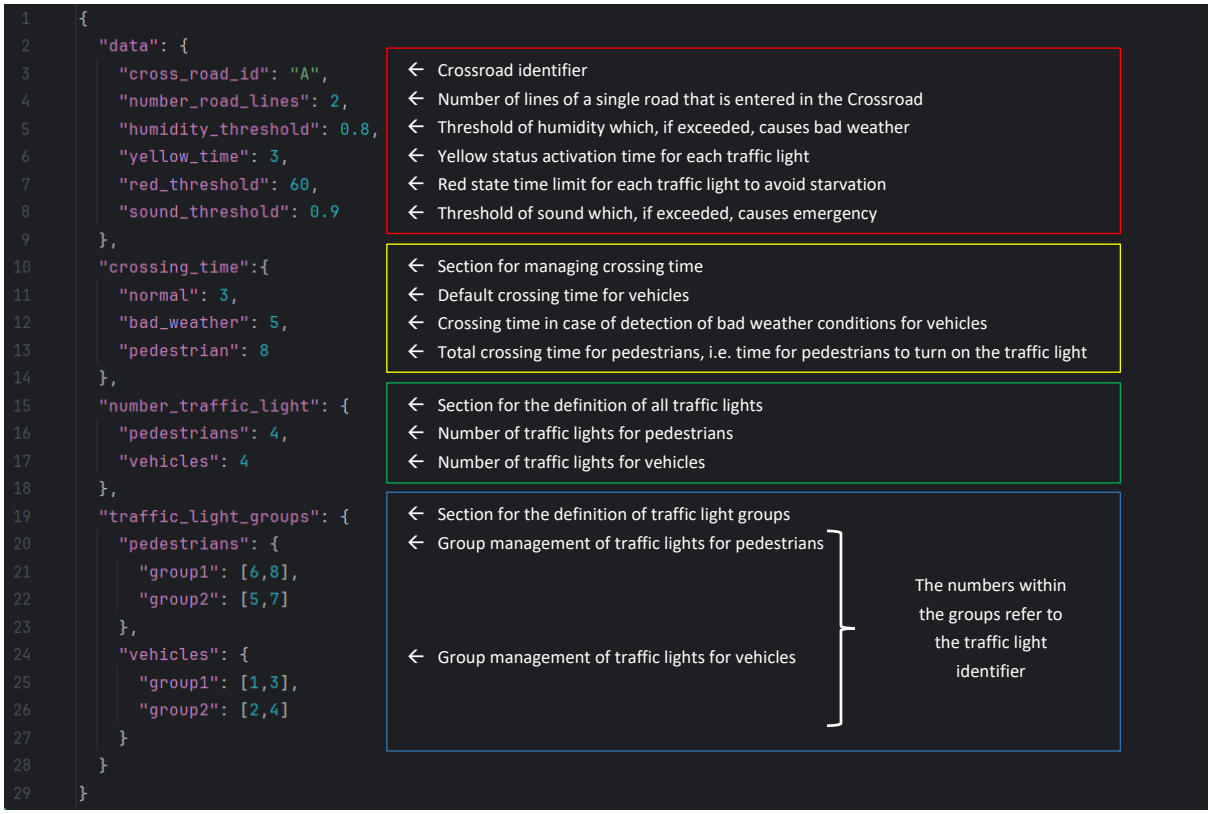
The Knowledge component is, together with MQTT broker, the only module with who all other component communicate in particular way is the central node that allows data exchange between component being part of the Autonomic Manager 1 and Autonomic Manager 2. The main contributor in storing data is the Monitor while other components add some information as a product from their own computations. In particular:

- *Monitor* → save sensor values;
- *Analyzer Traffic Light* → save the number of vehicles (value extracted from image analysis)
- *Planner Traffic Light* → save the button status of the pedestrian traffic light according with the computed plan
- *Executor Traffic Light* → save the status of the traffic light according with the action performed
- *Analyzer Traffic Switcher* → read the data relative to the number of vehicles that are used to compute the mean flux for each traffic light
- *Planner Traffic Switcher* + *Executor Traffic Switcher* → save the data relative to the switchers status according with the actions performed.

→ Subsystem_1: Traffic lights management (Reactive):

- Configuration (*Configuration*):

In the configuration component of the subsystem managing an individual crossroad, it is possible to define all the characteristics of the intersection. These include both structural aspects, such as the number of traffic lights for vehicles and pedestrians, and operational aspects, like crossing times and the management of traffic light activation groups.
All this information can be chosen and modified by the intersection designer based on the specific characteristics of the actual intersection in question. In particular, the traffic light groups encompass both lights for vehicles and those for pedestrians.
The decision to use groups is due to the ability to illuminate multiple traffic lights simultaneously, specifically those within the same group.
This choice allows the designer the flexibility to manage the illumination of traffic lights within the same group in a way that they do not interfere with each other, at their discretion.
The configuration file is shown below with brief descriptions.



- **Monitor:**

  In this subsystem, the monitoring component serves two primary purposes:

  a. It is responsible for forwarding all messages received from the sensors of managed resources—such as humidity, sound, camera (capturing images from each traffic light), and the pedestrian button (pressed by pedestrians awaiting to cross)—to *Analyzer_1.* This involves simply renaming the topic and republishing the message.

  b. For every incoming message from the sensors, the monitoring component records the corresponding values in the database (InfluxDB).

- **Analyzer (*Analyzer*):**

  Regarding the component related to *Analyzer_1,* its main tasks include:

  a. Gathers messages sent by the Monitor and categorizes them into four types:

     a. Messages related to the pedestrian button being pressed.

     b. Messages related to the perceived sound → if the value surpasses a threshold (defined in the configuration file) it detects an emergency. Emergency is indicated by the value True; otherwise, the value is False.

     c. Messages related to the perceived humidity → if the value surpasses a threshold, it detects a bad weather condition (defined in the configuration file). Bad weather is indicated by the value True; otherwise, the value is False.

     d. Messages related to the camera, specifically the captured photos. Utilizing a classifier (a Python script called Classifier.py), it determines the number of vehicles in the reconstructed image.

All these analyzed data are subsequently forwarded to a topic where Planner1 is listening.

b. Performs data storage in the database, focusing on the number of vehicles identified by the classifier for each traffic light.

- **Planner (*Planner_1*):**
  This component reads messages published by *Analyzer_2* and plans how the traffic light groups should be activated, answering the question: which group of traffic lights needs to be turned on, and for how long?
  The decision takes into account various factors and follows an algorithm to make the selection. Some considerations include:
  - The duration of the crossing time for vehicles is determined by weather conditions.
  - In case of an emergency, all traffic lights are set to RED until the emergency is resolved.
  - The group with the traffic light having the most vehicles waiting is given the right of way. For the unselected group, the waiting time is increased by the calculated GREEN time.
  - Traffic lights exceeding a maximum waiting time are placed in a waiting queue, and their corresponding group will be the next to be served with a GREEN time calculated based on the maximum waiting time among the traffic lights in the group.

  Additionally, in the database, the states of the traffic lights (GREEN/RED) and the states of the pedestrian buttons (PRESSED/NOT PRESSED) are saved.

- **Executor (*Executor*):**
  This component, once it receives the messages to activate the traffic lights from Planner_2 and after waiting for the programmed GREEN time, will command to capture a new image, initiating a new cycle of messages among the various components.

→ Subsystem_2: Switcher management (Reactive + Proactive):

- **Configuration (*Configuration_switcher*):**
  The subsystem responsible for managing traffic switchers has a configuration file which enables the customization of parameters that are closely tied to the specific environment in which the switchers are deployed. It is possible to specify the parameters to turn on the switcher for a specific time span and to schedule the analysis and the predictions of the traffic flow. Additionally, each switcher can be assigned to groups of one or more traffic lights, allowing a single switcher to analyze multiple traffic signals. The parameters are expressed in seconds.

- **Analyzer Switcher (Analyzer_*switcher*):**
  The Analyzer within the switcher subsystem uses a predictor to enable proactive behavior. The primary tasks of this component include:
  - Computing the mean flux for each switcher and its associated group of traffic signals. If the computed value is positive, the corresponding switcher is activated to divert traffic and minimize the incoming flux.

- Generating predictions based on recent historical data, which include Weekday, Time, and the computed flux mean values. The predictor employs a model to forecast the next positive flux occurrence and its corresponding time.
- Storing computed mean flux data in the InfluxDB database and transmitting prediction messages to the Planner component.

- **Planner Switcher + Executor Switcher (*Planner_switcher*):**
  The Planner component of the switcher subsystem uses information sent by the analyzer to effectively turn on switchers at planned times. The main tasks of this component are:
  - Checking the status of the switcher and act accordingly to turn on switchers based on the obtained data (see table below).
  - The switcher will remain active for the duration specified in the configuration file.
  - Activate/deactivate the switchers by sending messages (Executor component)

Behavior	ID	Condition	Action
Reactive	1	▪ Switcher ON ▪ Time of activation is not passed ▪ Detected positive mean flux	If we get positive flux mean, nothing happens
	2	▪ Switcher ON ▪ Time of activation is not passed ▪ Detected negative mean flux	Switcher turns OFF when the activation time is passed
	3	▪ Switcher OFF ▪ Detected mean flux	If we get positive flux mean, the switcher turns ON; otherwise nothing happens on the switcher
Proactive	4	▪ Switcher OFF ▪ Prediction is ready	The switcher turns ON earlier than the predicted time to prevent a higher influx at the anticipated moment.
	5	▪ Switcher ON ▪ Prediction is ready	Updates the activation time
	6	▪ Switcher ON/OFF ▪ Prediction is not ready	Generate predictions at regular intervals according to the configured settings.

Instructions to use the system.

1. git clone https://github.com/RobyBobby24/SE4AS_EcoTrafficFlowManager.git
2. In the cmd of the directory run: docker compose build.
3. In the cmd of the directory run: docker compose up.
4. To access InlfuxDB (admin:adminadmin).
5. To access Grafana (admin:admin).
6. To close, in the cmd of the directory run: docker compose down