Instituto Tecnológico de Estudios Superiores de Monterrey

Computerized Control

Final Project Progress: IoT Water Valves

Team

Ricardo Abraham Chapa Romero	A00824335
Juan Carlos Juárez	A00824524
Jesus Garcia Marcos	A00825235

Monterrey, N.L. April 25, 2022.

Introduction

In the present document we show the overall final project progress involving the established assignments in the project proposal along the progress with respect to our two main engineering goals which were to design an efficient control system and an API Interface for IoT communication.

The progress made will be detailed in the following sections, along with the methodology and actions performed to reach each of the solved tasks.

Assignments & Goals Progress

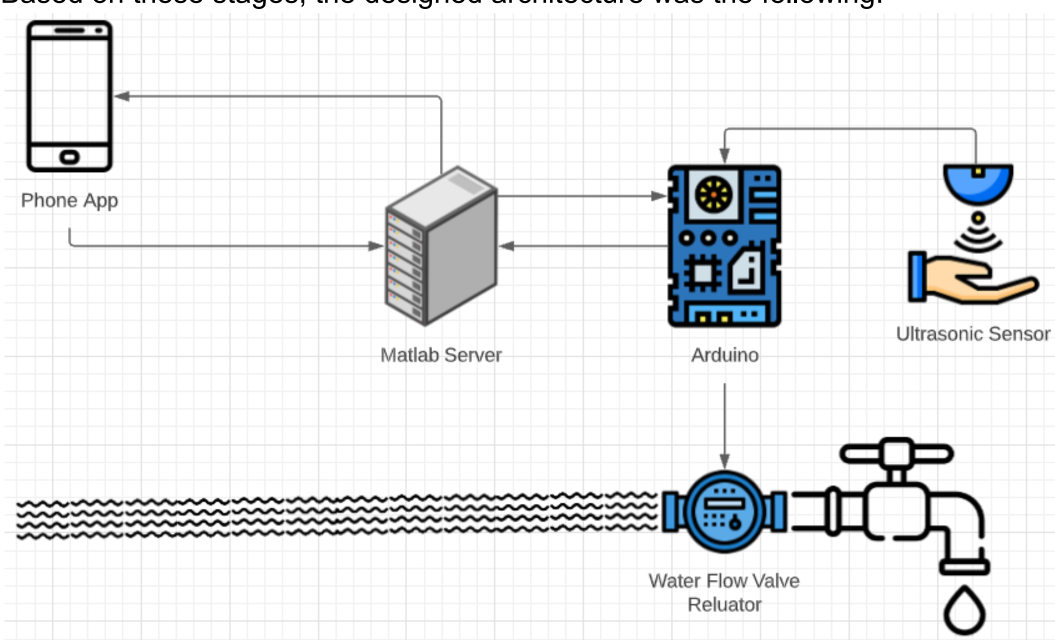
Before beginning the overall execution of the system, we designed a complete project architecture which served as a starting point for the system analysis, which helped us analyze and understand the separate parts on which the project consists.

The prototype to be designed will be a functional system ready for actuator adaptation, which will be able to maintain control of the water pressure provided to the system.

The project prototype consists on 3 stages:

1. **Physical Microcontroller:** Represents the physical system that will interact with humans and provide an analog output for the water pressure control.
2. **Control Server:** For this prototype, it will function as the Main Processing and Computing system, which will be in charge of creating the control system design as well as providing an IoT Interface.
3. **IoT User Interface Web Server:** This will be the Web Server which will provide the user the ability to change the Water Limit for the system.

Based on these stages, the designed architecture was the following:



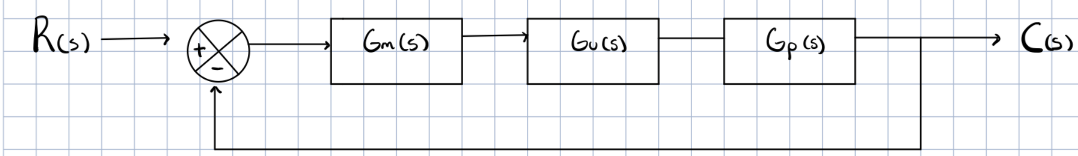
After having identified the architecture and the overall system design to be used, we proceeded with the specific engineering assignments for this project, always having in mind to produce an innovative solution that not only involves a well designed control system, but a complex and interesting IoT architecture.

Equipment

- Analog flow control valve → To change water pressure
- Analog water flow meter → To measure the water flow
- Arduino → Microcontroller
- Server → Hooks matlab and posts the data through HTTP requests
- Communication Devices:
  - Phone → Used to communicate with the app and matlab server
  - Computer → Used to communicate with the web application and matlab server

Assignment 1.- To design a Software Control System that depending on a set value for daily consumption dims the analog output value (which will be used to dim the water pressure) as it reaches its daily maximum.

• Block Diagram



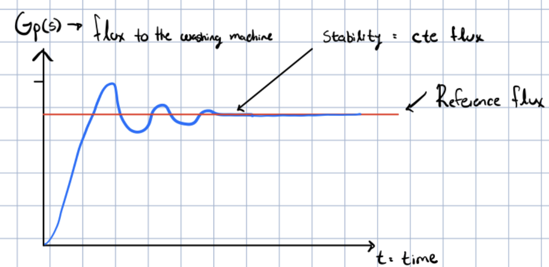
R(s) = Water flux coming from the water intake

Gm(s): PID manipulation that will control the incoming water flux. (pressure).

Gu(s): percentual control from user water safe.

Gp(s): resultant water flux to the washing machine. (Bernoulli?, Poiseuille?).

C(s) = resultant water per second going to the washing machine.



In the first stage, we define that the input is the water flow coming, we then process this signal with the water flow valve, as for the variable input the user will be allowed to specify the limit of water consumption per day. We then have the water flow that's coming out as the output signal.

The signal will be processed through the matlab server and the user will be able to specify the limit of consumption through the phone web application. This will allow connectivity through IoT as well as having a microcontroller apply the data that matlab has sent out to the microcontroller. The closer the water consumption is to the limit the more the output signal will stabilize.

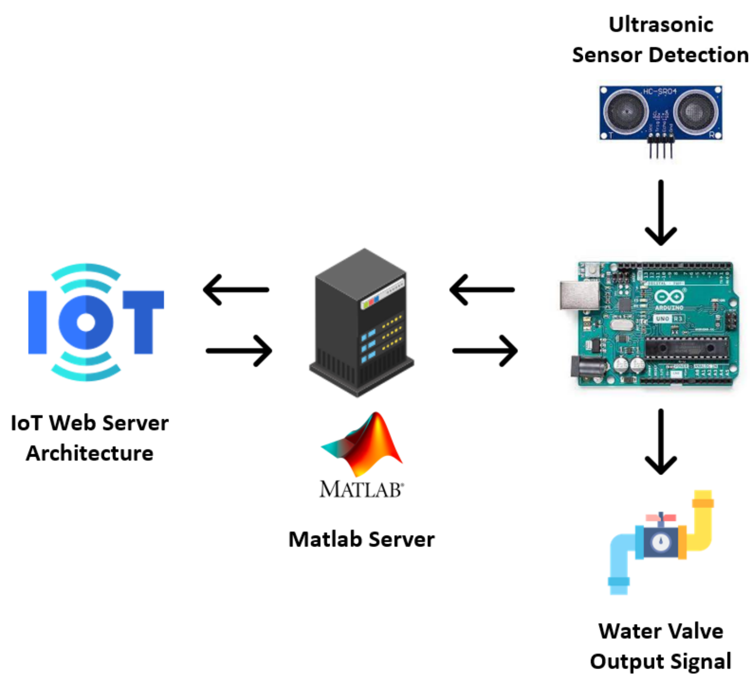
Once that we had the first assignment defined and we knew where our project was headed, we moved on to the next two engineering assignments:

Assignment 2.- To implement a physical device that represents the dimmer of water pressure within the valve, which will act based upon the analog output value coming from the Control System.

We wanted to not only provide a system that would be able to control a signal, but also a project that would be highly stable, and most importantly, something with a complex and vanguardist architecture using the newest software engineering technologies. We provide references to these technologies in the References section of this document.

We started now with the second assignment, which refers to implementing a physical device so the prototype does not stay just in a simulation, but also in a physical stage very close to a complete adaptation.

By doing some research, we found an interesting architecture which consists of an Interface - Server Model by using an Arduino Uno as the Interface and a Matlab Instance as the computing Server. This is possible thanks to the Matlab Arduino Support Package, which provides several libraries for some sensors, and allows the arduino to be used as a Human - Interface Tool.



The way the system would work would be by using an Ultrasonic sensor which would be able to detect the presence of people in front of the water valves and would be able to provide water to flow through the system. This information would be then transmitted to the Matlab server and all the Control System's computations would be performed inside there (which also helps the debugging and analysis of the system's output). The architecture to be used for the first assignment is the following:

Once we defined this, we proceeded with the next crucial step for this assignment: the code development. For this we did some research about how the Matlab Arduino Support Package worked, and for this we had to upload a server code and update the Arduino firmware from Matlab. Once we had done this, we proceeded with the design of the main processing unit, which would be inside the Matlab Server. The code for this part of the system was straightforward:

- A new Arduino Connection is defined within Matlab.
- A new Ultrasonic Sensor Instance is defined.
- Predefined variables are defined, such as the ones used for the control system and other global variables like the current water limit.
- A While True Loop to execute the system in a repeated way
- An API call to the Web Server to obtain the water limit (which is explained in the next assignment explanation).
- The overall Control System execution as well as the check for water flow with the Ultrasonic Sensor

The code designed for this part was the following:

```
% Tecnológico de Monterrey
% Computerized Control
% Final Project
% IoT Water Valves

% Create connection to Arduino Uno
arduino_uno = arduino('COM6','Uno','libraries','Ultrasonic');

% Create Sensor Instance
ultrasonic_sensor = ultrasonic(arduino_uno,'D2','D3');

% Variable for While True
true_variable = 10;

% Initial Water Limit
water_limit = 0;

% Control System predefined variables
% ###################################

% Main Loop
while true_variable > 1

    % Check Water Limit -----------------------

    api_url = "http://localhost:2000/api/check-water-limit";

    api_call = webread(api_url);

    new_limit = str2double(api_call);
```

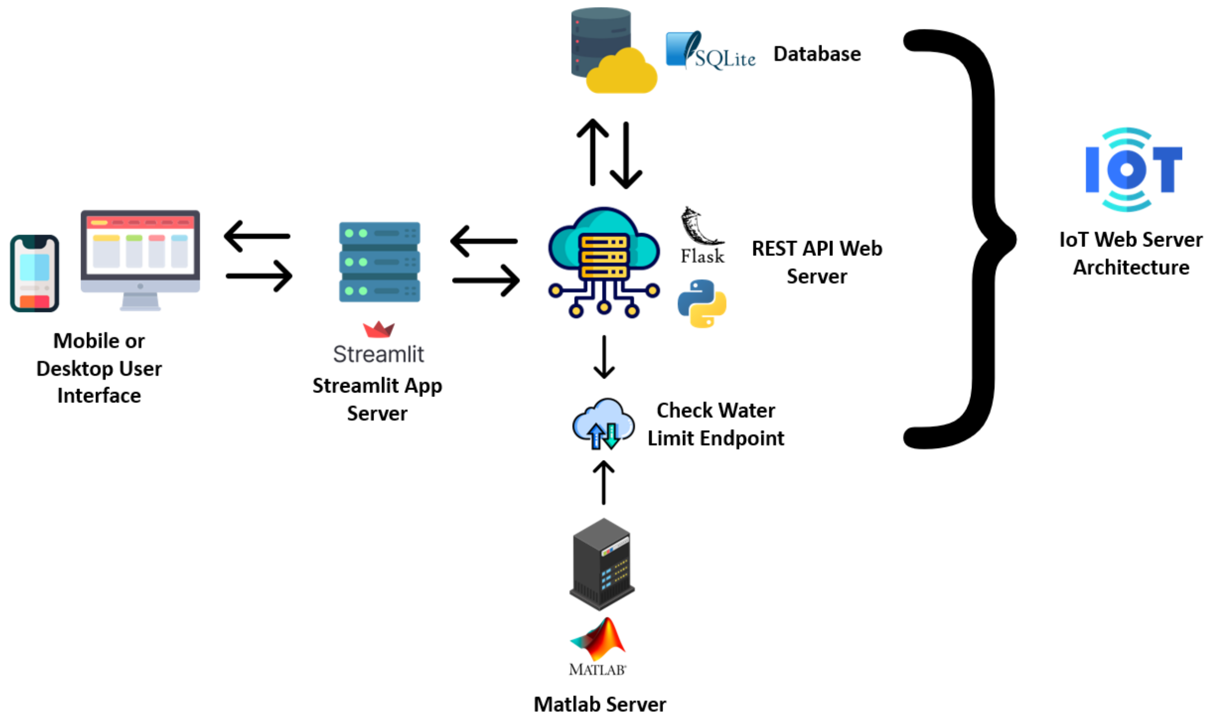
```
    if ~eq(water_limit, new_limit)

        fprintf('Water Limit has been changed from %f to %f.\n', water_limit,new_limit);
        water_limit = new_limit;
        pause(3);

    end

    % --------------------------------------

    % Regular System Processing --------------

    % Control System Execution
    % ########################

    % Distance in cm
    distance = readDistance(ultrasonic_sensor) * 100;

    % If distance is less than 10 cm detection occurs
    if(distance < 10)
        disp(distance);
    end

    % --------------------------------------

end
```

Finally, we reached the third assignment:

Assignment 3.- To develop a Mobile IoT app that interacts with the Control System, in order to set the daily maximum values from a remote device in an interactive and simple way.

For this final part, we decided to build a highly available software architecture that would be able to process the data transfer between several endpoints.

For this we designed an architecture based on a REST API on Flask, which is a Python Framework for API and endpoints interfacing. This system would provide a UI Web Server built over Streamlit App Framework on which the users would be able to change the water limit as they desire. This Graphical Web Server would then interact with the REST API in order to retrieve the current water limit and to also update it. The REST API would then retrieve and update such data on the Database, and the endpoints would provide the usage of the data. The architecture designed for this was the following:



Once we had the IoT Web Server architecture, we proceeded with the code development. The first part consisted of creating the REST API Web Server by using the Flask Framework for the Endpoint for checking the water limit. This endpoint was used by both the Matlab Server as well as the Streamlit App Server, as both need to access the information of what is the current water limit inside the database. The REST API code was the following:

```python
# Tecnológico de Monterrey
# Computerized Control
# Final Project
# IoT Water Valves

from flask import Flask
import sqlite3
import os

current_dir = os.path.dirname(os.path.abspath(__file__))

app = Flask(__name__)

# Main Route
@app.route("/")
def entry():
    return "Server is running..."

# Water Limit
@app.route("/api/check-water-limit")
def check_water_limit():

    connection = sqlite3.connect(current_dir + "/system.db")

    cursor = connection.cursor()

    data_query = "SELECT * FROM system_limit"
```

```python

    cursor.execute(data_query)

    data = cursor.fetchall()

    current_water_limit = 0

    # Get the Max Capacity which is the second argument
    current_water_limit = float(data[0][1])

    current_water_limit = round(current_water_limit, 2)

    current_water_limit = str(current_water_limit)

    connection.commit()

    return current_water_limit


if __name__ == '__main__':
    app.run(host='0.0.0.0', port=2000, debug=True)
```

After having the REST API ready, we continued with the Database creation by using a .sql file in SQLite3 which would contain the water limit value:

```sql
CREATE TABLE system_limit (
    main_id TEXT NOT NULL,
    water_limit REAL NOT NULL
);

INSERT INTO system_limit (main_id, water_limit)
VALUES("main", 1);
```

Finally, we finished with the creation of the User Interface that runs on the Streamlit App Server. This uses some other files for its execution as well:

app.py

```python
# Tecnológico de Monterrey
# Computerized Control
# Final Project
# IoT Water Valves

import streamlit as st
import time
from datetime import datetime, timedelta
from fetch import *
from update import *

st.title("Computerized Control - Water Control System")

# Total Number of People currently Inside

current_water_limit = get_water_limit()

st.metric("Current Water Limit in Liters: ", current_water_limit, delta=None,
delta_color="normal")

# Change Max Capacity Input

number = st.number_input('Insert a number to change Water Limit per day (Liters): ')

if st.button('Change Water Limit') and number >= 0:
    update_water_limit(number)
    st.write('Water Limit has been changed succesfully.')
    time.sleep(3)
else:
    pass

# -----------------------------
# Re-Run App every half a second
# -----------------------------

time.sleep(0.5)
st.experimental_rerun()
```

fetch.py

```python
# Tecnológico de Monterrey
# Computerized Control
# Final Project
# IoT Water Valves

import requests

def get_water_limit():
    r =requests.get('http://localhost:2000/api/check-water-limit')
    return r.text
```

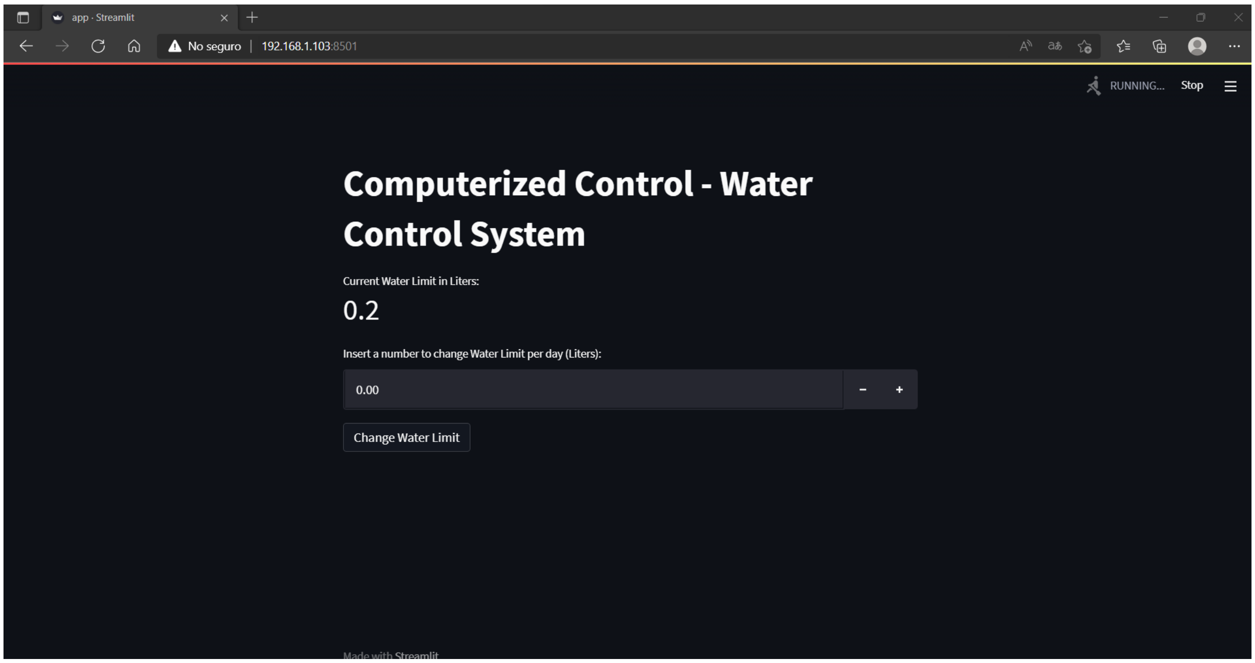
update.py

```python
# Tecnológico de Monterrey
# Computerized Control
# Final Project
# IoT Water Valves

import sqlite3
import os
import pandas as pd

def update_water_limit(limit):

    current_dir = os.path.dirname(os.path.abspath(__file__))

    connection = sqlite3.connect(current_dir + "/system.db")

    cursor = connection.cursor()

    data_query = "UPDATE system_limit SET water_limit = {0} WHERE main_id = 'main'".format(limit)

    cursor.execute(data_query)

    connection.commit()
```

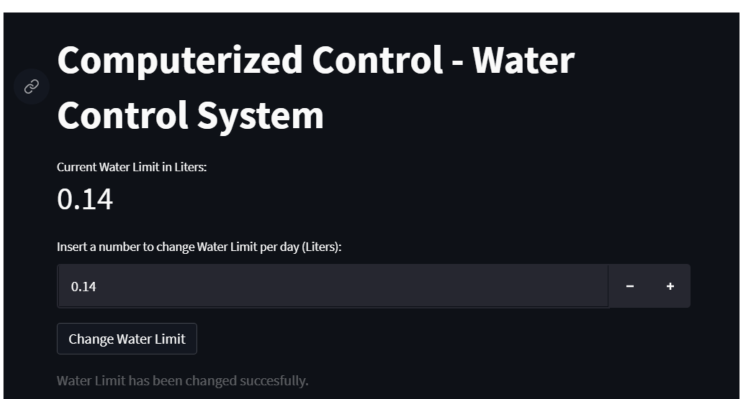
Tests & Results

Once we had performed all the assignments mentioned above, the following tests were made:

We launched and executed our SQLite Database and started both the REST API Web Server as well as the Streamlit App Server. The Streamlit App Server looked as follows:



Once we had the Streamlit App running, we tried to modify the Water Limit by changing it inside the Selection Bar to see how it behaved and the system worked correctly by changing the limit to 0.14:



```
...> ^C^C^Cjcjuarez@DESKTOP-ENLD3T8:~/TecDeMty/Computerized_Control_Final_Project$ ^C
jcjuarez@DESKTOP-ENLD3T8:~/TecDeMty/Computerized_Control_Final_Project$ sqlite3 system.db
SQLite version 3.31.1 2020-01-27 19:55:54
Enter ".help" for usage hints.
sqlite> DROP system_limit;
Error: near "system_limit": syntax error
sqlite> DROP TABLE system_limit;
sqlite> .read db.sql
sqlite> SELECT * FROM system_limit;
main|5.0
sqlite> SELECT * FROM system_limit;
main|0.04
sqlite>
```

The Matlab Server was also capable of detecting these changes by using the api, just as shown in the Matlab Console:

```
water_limit = 0;

% Control System predefined variables
% ####################################

% Main Loop
while true_variable > 1

    % Check Water Limit -----------------------
```

```
Command Window
New to MATLAB? See resources for Getting Started.
>> clear all
>> final_project
Water Limit has been changed from 0.000000 to 0.140000.
fx
```

And finally, we checked the ultrasonic detection on Matlab if the distance detected was less than 10 cm so it would be printed out to the console (in order to provide an output signal for the water valve):

```
    % Check Water Limit -----------------------
Command Window
New to MATLAB? See resources for Getting Started.
>> clear all
>> final_project
Water Limit has been changed from 0.000000 to 0.140000.
    9.1034

    9.7241

    9.6207

    9.5690

    8.4483

    6.6552
```

Bibliography

Streamlit • The fastest way to build and share data apps. (2022). Retrieved 25 April 2022, from https://streamlit.io/

SQLite Home Page. (2022). Retrieved 25 April 2022, from https://www.sqlite.org/index.html

Welcome to Python.org. (2022). Retrieved 25 April 2022, from https://www.python.org/

Welcome to Flask — Flask Documentation (2.1.x). (2022). Retrieved 25 April 2022, from https://flask.palletsprojects.com/en/2.1.x/