SMART DRAINAGE MANAGEMENT SYSTEM

A Project report submitted in the partial fulfillment of the requirements
For the award of the degree of

BACHELOR OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY

By

G. KEERTHI (16HP1A1215)

V. PHANI SONIKA (16HP1A1224)

T. HEPSIBA SUSAN (16HP1A1213)

Under the esteemed Guidance of

Mrs. G.DURVASI M.Tech
Assistant Professor
Department of IT

DEPARTMENT OF INFORMATION TECHNOLOGY

ANDHRA LOYOLA INSTITUTE OF ENGINEERING & TECHNOLOGY
(Approved by AICTE, Affiliated to JNTU KAKINADA)
VIJAYAWADA-520 008

2016-2020

ANDHRA LOYOLA INSTITUTE OF ENGINEERING & TECHNOLOGY

(Approved By AICTE, Affiliated to JNTU KAKINADA)

Vijayawada, Krishna District (A.P)

CERTIFICATE

This is to certify that this project work entitled **"Smart Drainage Management System"** is the bonafied work of **G.KEERTHI (16HP1A1215), V.PHANI SONIKA (16HP1A1224) and T.HEPSIBA SUSAN (16HP1A1213)** of final year B.Tech which they have submitted in partial fulfillment of the requirements for the award of Degree of Bachelor of Technology in **Information Technology** to JNTUK during the academic year 2019-2020.

Mrs. G. DURVASI
Project Guide
Assistant Professor
Department of IT

Mr. S. KISHORE BABU
Head of The Department
Associate Professor
Department of IT

EXTERNAL EXAMINER

DECLARATION

We hereby declare that this project work entitled **"SMART DRAINAGE MANAGEMENT SYSTEM"** ha**s** been carried out by us and contents have been presented in the form for the award of the Degree of Bachelor of Technology in INFORMATION TECHNOLOGY.

We further declare that this dissertation has not been submitted elsewhere for any degree.

G. KEERTHI (16HP1A1215)
V. PHANI SONIKA (16HP1A1224)
T.HEPSIBA SUSAN (16HP1A1213)

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without mentioning the people who made it possible and whose constant guidance and encouragement crown all the efforts with success.

I would like to take this opportunity to express my profound sense of gratitude to **Dr. Fr. XAVIER S. J ,** Director of Andhra Loyola Institute of Engineering &Technology, Vijayawada, for allowing me to utilize the college resources there by facilitating the successful completion of my project.

I feel elated to extend my sincere gratitude to **Mr. S. KISHORE BABU**, Head of the Department, Information Technology, for his encouragement all the way during analysis of the project. His annotations, insinuations and criticisms are the key behind the successful completion of the project and for providing me all the required facilities.

In particular, I am very grateful to my Project Guide **Mrs. G. DURVASI**, Assistant Professor, Department of IT for his technical guidance and support throughout the project. I am deeply indebted for his support and cooperation.

I gratefully acknowledge the support, encouragement and patience of my parents. I would like to thank all the teaching and non-teaching staff members of Department of IT, Andhra Loyola Institute of Engineering and Technology, Vijayawada, who have kindly cooperated with me during my project and helped me in making this effort a significant work, without whom I could never get the pleasure in bringing forward my first real venture in practical computing in the form of this project entitled **"SMART DRAINAGE MANAGEMENT SYSTEM"** allotted to me during the final year.

G.KEERTHI (16HP1A1215)
V.PHANI SONIKA (16HP1A1224)
T.HEPSIBA SUSAN (16HP1A1213)

Abstract

As we are developing cities into smart cities hence there is a need for designing proper underground infrastructure which includes underground water pipelines, drainage monitoring for the purpose of keeping the city clean. If the drainage system is not monitored properly it may lead to blockage and sewage overflow which in turn leads to contamination of pure water resulting in spreading of infectious diseases. So different kind of work has been done to detect, maintain and manage these underground systems. Also, leaks and bursts are unavoidable aspects of water distribution system management and can account for significant water loss within a distribution network if left undetected for long period. This project represents the implementation and design functions for monitoring and managing underground drainage system with different approaches. It gives a description of water flow system by giving water flow rate and detecting overflow and also it uses detection method to detect blockage defects in sewer pipeline based on Waterflow.

Keywords : IOT, WSN, Drainage System, Sensors.

INDEX

FIGURE INDEX

TABLE INDEX

SCREEN INDEX

1. INTRODUCTION

1.1 Introduction

Sewage framework assumes a significant job in huge urban communities where a huge number of individuals live. Sewage framework is known as the base for land dryness from the overabundance and unused water. Downpour water and wastewater. Sewage conditions ought to be observed so as to keep up its appropriate capacity. Actually, not all zones have a sewage observing group. It prompts inconstant observing of the waste condition. The inconstant checking has added to the obstructing of the waste that suggests the welcome which triggers flooding in the area. Manual checking is additionally hard. It needs a ton of committed people who are just ready to record constrained reports with low exactness. The issue emerges in such waste lines can make difficult issues in the everyday schedule of the city. Issues, for example, blockage because of waste material, abrupt increment in the water level if the best possible cleaning moves are not made every once in a while. The present sewage framework isn't electronic because of which it is difficult to know whether blockage is happening specifically area. Likewise we don't get early alarms of the blockage or the expansion in water level. Henceforth identification and fixing of the blockage becomes tedious and furious. NodeMCU combines features of WIFI access point and station + microcontroller. These features make the NodeMCU extremely powerful tool for Wifi networking. It can be used as access point and/or station, host a webserver or connect to internet to fetch or upload data. By using this types of wireless sensor network we can notified of sensor data through internet.

1.2 Problem Statement

Overflow of sewage on roads is been a major problem in many developed and under developed cities as well. Existing drainage system is manual monitoring and all areas doesn’t have proper monitoring teams. Manual monitoring is difficult and ineffective.

One of the major problems on a rainy day is overflow of manholes and drainage on a road. which is caused due to no information on manhole level of filling. Another problem in drainage system is difficulty of finding blockage in underground drainage pipes. Most of time drainage management team of municipality detects the problem in the drainage system when it is causes the trouble. The main reason for this is manual monitoring because it's difficult and some time it takes lots of time.

1.3 Project Overview

The objective of this work is to implement a smart drainage management system which will reduces the risk of drainage issues in the future and gives a better monitoring system for the drainage which will eliminate the manual monitoring of sewage. A smart drainage management system that provides low cost method in locating the blockage in the underground drainage pipes and the overflow under the manhole by using sensors and internet of things.

1.4 Objectives of the Project

1. This project is to keep the city clean, safety and healthy. And replace the manual work of drainage monitoring for the safety of sewer workers, human and city.
2. Predictive system: The intelligence of sensors and predictive system identifies the drain is clogged and also give us information of level of manhole filled so, action can be taken by the authorities.
3. Completely connected: The sensors are communicated through Wi-Fi communication modules to share information.
4. This information can be viewed by authorized users at any time and any place in the internet.
5. The information of the assets will be stored on the cloud for 30 days and it can be downloaded and viewed any time by the users.

2. LITERATURE SURVEY

2.1 INTRODUCTION

A system of sewer pipes collectively called sewers, collects the sewage and takes it for further treatment or disposal. Properly functioning septic tanks require emptying every 2–5 years depending on the load of the system. An underground sewage system was designed where all the waste from the home and industrial waste water are collected together and driven together. Sewage overflow on roads is a major problem in cities, where large numbers of complaints are launched and no actions are being taken.

Most of the cities adopted the underground drainage system and it is the duty of Municipal Corporation to maintain cleanliness, health and safety of cities. If the drainage system is not properly managed then pure water gets contaminate with drainage water and infectious diseases may get spread. The drainage gets blocked during rainy season and it will create the problems to routine life like traffic may get jammed, environment will become dirty and totally it will upsets the public. In many cases blocked drains can cause sewage and waste water to back up and potentially come up onto your property. Suppose if there is a facility that officials or concerned persons come to know immediately the blockage or clogging inside the drainage channels in which area and exact place where it gets blocked. So our main focus is to monitor the manholes using sensors. If drainage gets blocked or water overflows, the sensor senses the activity and sends the information via transmitter to the concern persons. Manhole maintenance by human is very difficult because environment is very poor and it is difficult to go inside of manhole for inspecting the states of manholes all the time. Immediately it is not possible to confirm if the person intrudes the manhole or an accident happens inside of the manhole. The drainage system is essential for the people who live in urban areas as this system reduces flood effect by carrying water away (a facility to dispose liquid waste).Improper maintenance of existing drainage system leaving many people suffer. The major areas effecting in many urban areas because of faulty, improper drainage monitoring system are roads. Roads are built up to

support human and vehicular traffic. Currently urbanization has negative impact on drainage system in many cities and towns across the globe. Irregular monitoring of drainage system leads to contamination of water and leads to water borne diseases. Stagnation of water on roads will causes roads to damage. More importantly flooding of roads lead to traffic jams and causes loss of valuable human hours, loss of revenue and employment. Ground water contamination is also possible, if once it is contaminated it's very difficult to clean up. A good and efficient drainage system is badly required for the developing countries like India. In creation of many smart cities the architecture of drainage system plays an important role. To maintain a good and proper drainage system it takes more human resource

2.2 Existing System

Existing drainage monitoring system is manual monitoring by municipal sewage workers who usually monitors the drainage system in certain interval of time.

2.3 Disadvantages of Existing System

- Manual work.
- Less efficient.
- Manual monitoring may NOT cover large area.
- Blockage detection is difficult.
- Late detection of problem.
- Unable to monitor different location drainages at a time.

2.4 Proposed System

In this project, we design smart real-time drainage monitoring system using various sensors such as water flow sensor and ultrasonic sensor. The water level will determine the extent of the flood as low, medium or high. This will enhance early flood detection by which drainage workers can take precautions. The water flow sensors will detect the rate of water flow which helps in

detecting the blockage in the drain lines and provides the early notofications so that we can clean it as early as possible.

If there is any abrupt increasing change in any of the parameters, the system will informs about the problem along with the location.

2.5 Advantages of Proposed System

- It eliminates the manual work.
- Indicates the problem before it occurs.
- It has very few interconnections.
- The embedded system is small in size.
- The system is less expensive.
- It has fast operation.
- It has improved product quality.
- It optimizes use of system resources.
- It has low power operation.

2.6 Requirement Specification

Software Requirements

- Operating system: Arduino
- Coding language: C/C++.

Hardware Requirements

- Arduino UNO
- Water flow Sensor
- Ultra Sonic Sensor
- Node MCU
- Power Supply

3. ANALYSIS

3.1 Introduction

This will cover the problem analysis, requirement analysis and conclusion of the analysis. From the analysis, the criteria and problem of the current system will be analyzed and characterized. The analysis phase defines the requirements of the system, independent of how these requirements will be accomplished. This phase defines the problem that the customer is trying to solve.

The goal of analysis is to determine where the problem is in an attempt to fix the system. This step involves breaking down the system in different pieces to analyze the situation, analyzing project goals, breaking down what needs to be created and attempting to engage users so that definite requirements can be defined. The deliverable result at the end of this phase is a requirement document.

3.2 Arduino Software

Ever since computers first entered the world, programming has always been seen as a rather esoteric process. With all its codes and symbols, programming has never been very beginner friendly. It usually takes years and years of studying to get even the most basic concepts down and it's especially difficult to apply these codes to real work devices. Nowadays, however, knowing how to code and program is a very useful skill to have. Arduino IDE is a coding software that makes the programming world more accessible to beginners with its simple interface and community-driven system.

3.2.1 Simple Interface

As mentioned in the beginning, programming is very daunting for people who have never had a background for them. There's no doubt that when thinking of programming, people mostly visualize the raining green code of the Matrix. However, Arduino IDE makes coding so much

simpler for beginners. It is, in essence, a text processor with programming-specific functions.

One such function is auto-format. Auto-formatting is very helpful for people who don't really know how to format code. With one click of your mouse, the code that you've written will be arranged in an easy-to-understand format. In addition, there are dozens of templates that people can choose from if they want to follow more complex Arduino sketches.

3.2.2 Variety of Templates

The templates provided in the Arduino IDE are extremely helpful. Of course, not all of them will be relevant to what you're doing, but they are great for starting out. They have some very basic codes such as Blink and Keyboard Logout that users can utilize to create more complex codes.

You can either use those codes to create other codes with similar functions, or you can add those codes to other codes to create multi-functional codes. Once the sketches have been created, users can easily upload their sketches to their chose Arduino boards. The uploading does take some time but the process of uploading itself is not complicated at all.

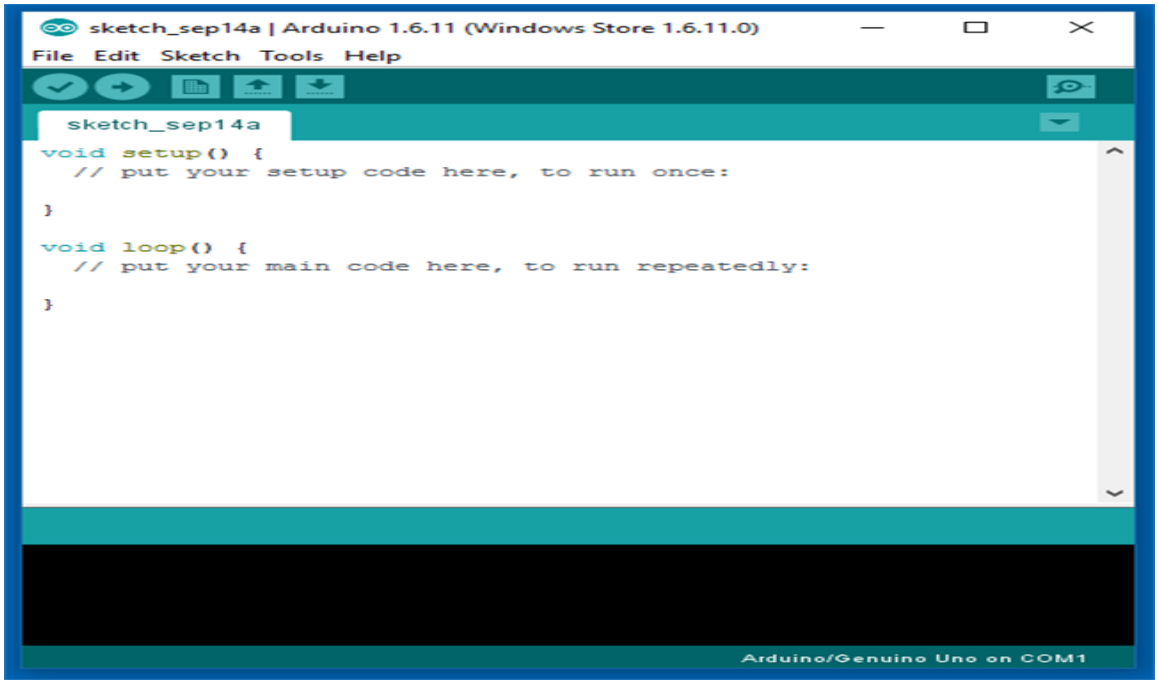


Fig 3.2.2.1 Sample Template

3.2.3 Community-Driven System

One of the best things about the Arduino IDE is the fact that it's community-driven. Arduino has a very active forum where users can share their creations with other programmers and get feedback and troubleshooting tips. More advanced users can provide their own tips as well. What's great about the Arduino IDE is that this focus on community is accommodated by the software itself. Users have the option of uploading their code directly to the forums which is extremely convenient.

3.2.4 Join the Programming World with Arduino

There is no doubt that if you are a beginner that's just starting out with programming, Arduino IDE is one of the best programs there is. It's easy to use and has a lot of templates that are helpful for beginners. Not only that, users can take advantage of the community of Arduino users to make their creations better. For anyone who thinks that programming is something that they can never get into, Arduino IDE will definitely change your mind.

The templates provided in the Arduino IDE are extremely helpful. Of course, not all of them will be relevant to what you're doing, but they are great for starting out. They have some very basic codes such as Blink and Keyboard Logout that users can utilize to create more complex codes. You can either use those codes to create other codes with similar functions, or you can add those codes to other codes to create multi-functional codes. Once the sketches have been created, users can easily upload their sketches to their chose Arduino boards. The uploading does take some time but the process of uploading itself is not complicated at all.

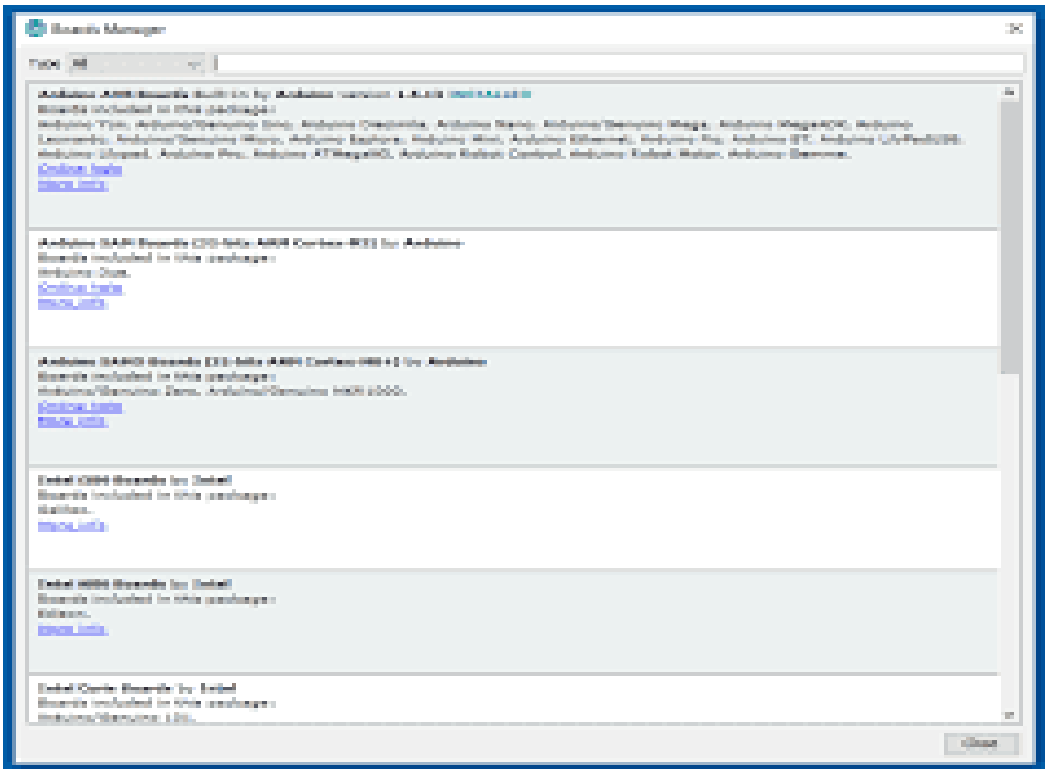


Fig 3.2.4.1 Arduino IDE

3.2.5 Pros

- Sleek interface design
- Active community
- Lots of templates
- Cross-platform

3.2.6 Cons

- Uploading takes time
- Cannot change font style
- Quick buttons cannot be added

3.2.7 Download and Install the Arduino Software

3.2.7.1 Download

Go to the Arduino website and click the download link to go to the download page. On the download page, click the Windows link to download the Arduino software for Windows as shown below.

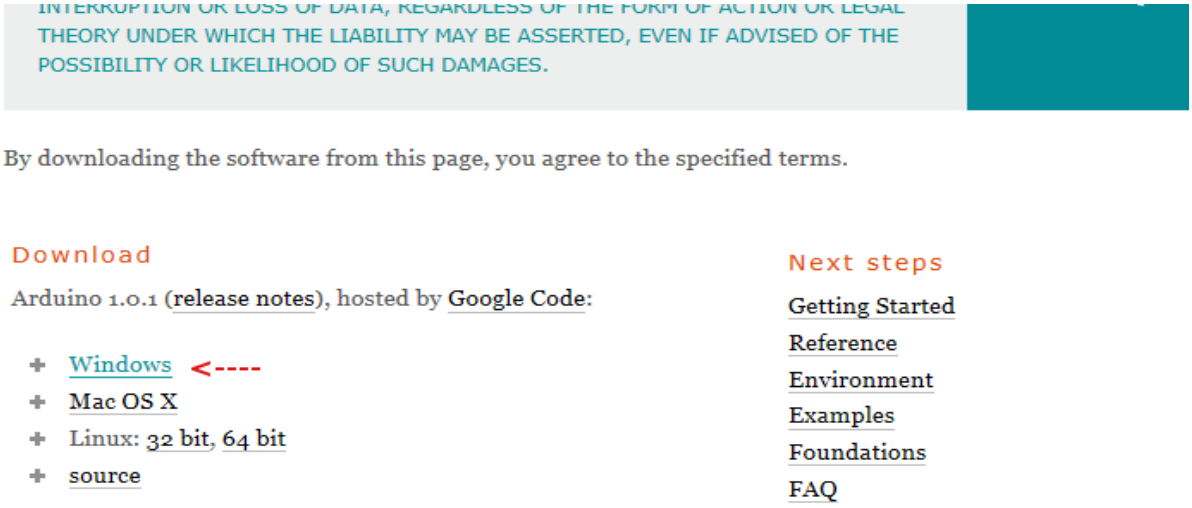


Fig 3.2.7.1.1 Arduino downloading page

3.2.7.2 Install:

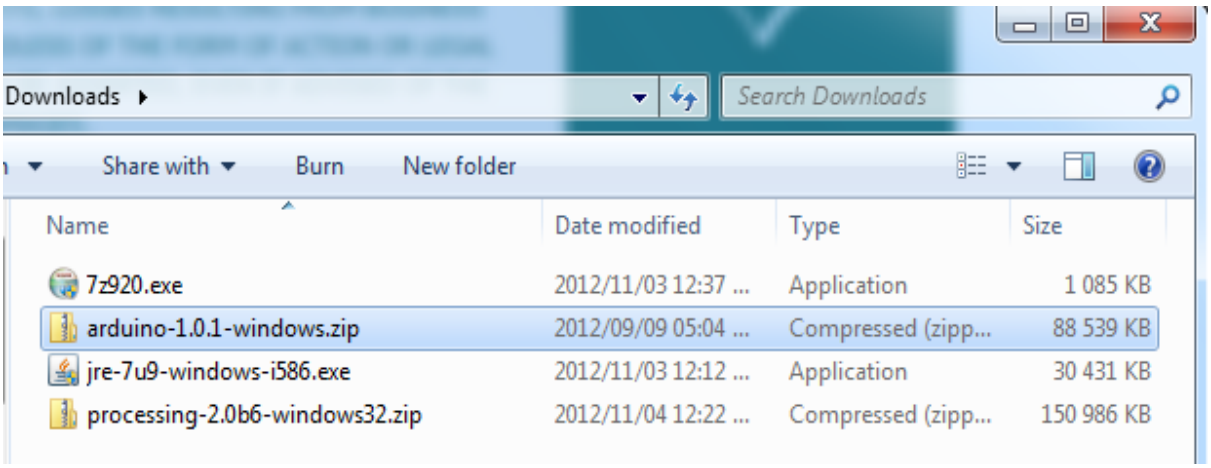


Fig 3.2.7.2.1 Arduino file screen shot

Install the Arduino Windows Drivers:

1. Plug the Arduino into the PC

Plug the Arduino board into the PC. Windows will try to install drivers, but will fail.

2. Start the Windows Device Manager

Click the Windows Start menu button.

Right-click Computer on the menu and then click Properties from the pop-up menu:

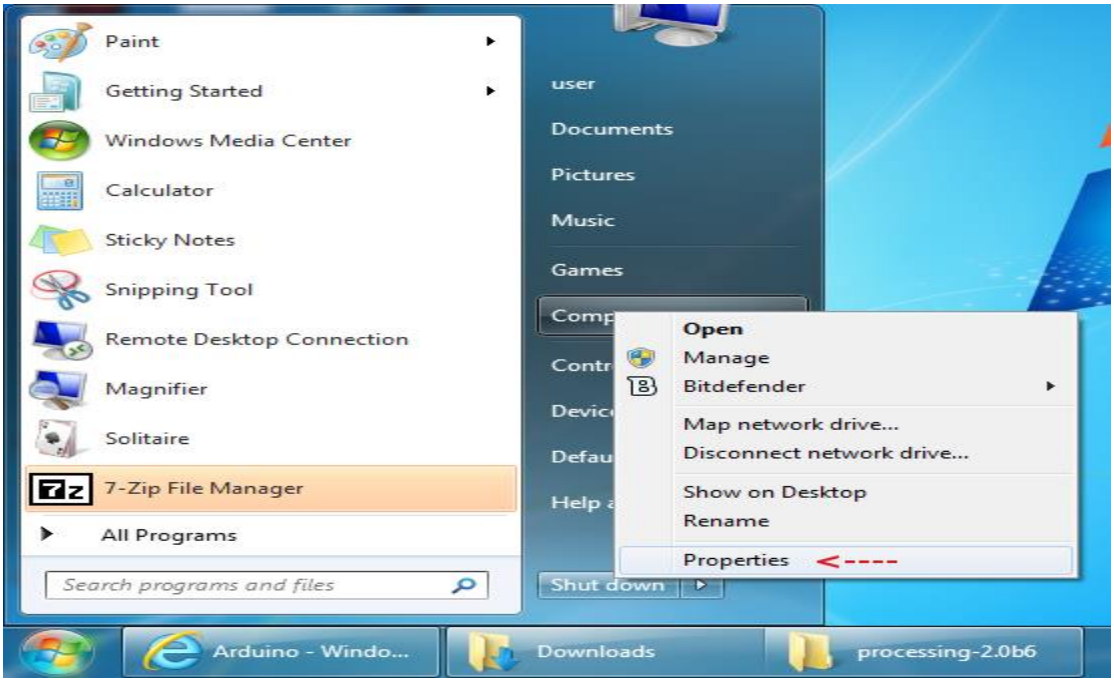


Fig 3.2.7.2.2 Windows menu

Click the Device Manager link to start the device manager:

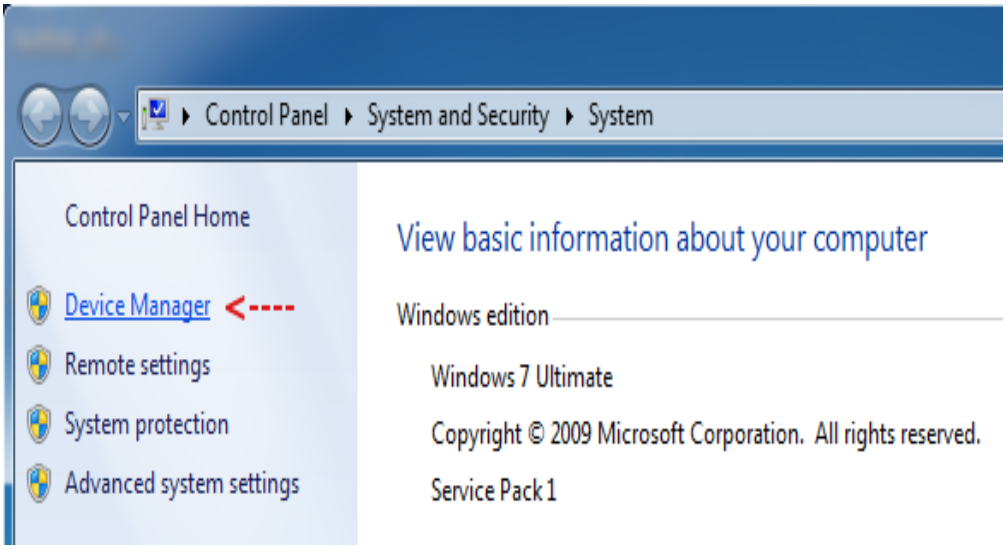


Fig 3.2.7.2.3 Device Manager link

The Device Manager will open and display the Arduino Uno:

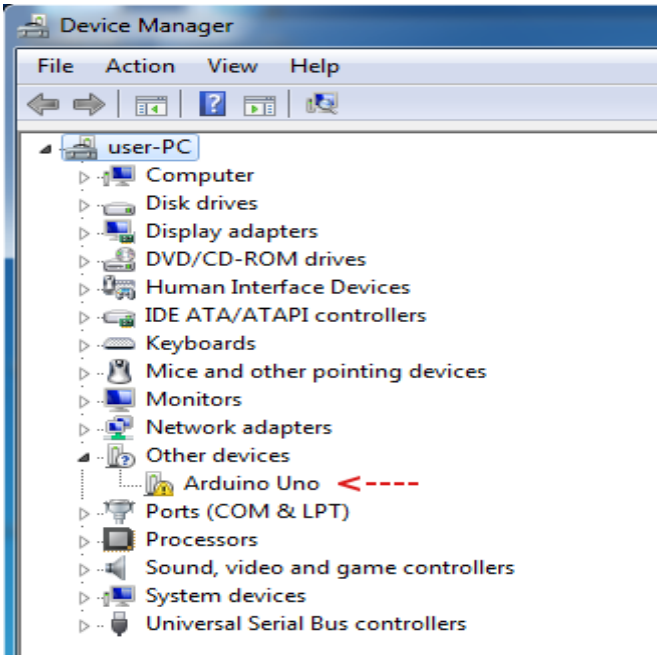


Fig 3.2.7.2.4 Arduino uno on device manager list

3.2.7.3 Install the Device Driver:

In the Device Manager Window, right-click the Arduino board and then click Update Driver Software... on the pop-up menu:

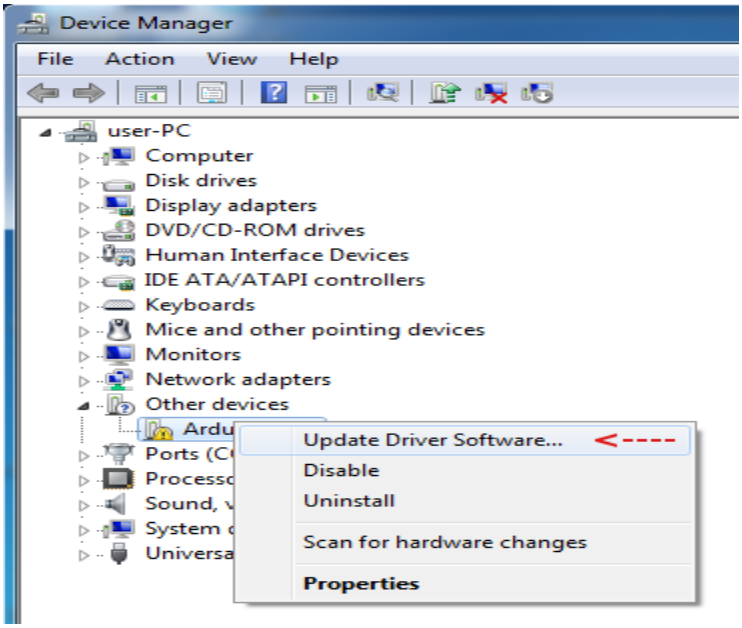


Fig 3.2.7.3.1 Arduino board and then click Update Driver Software

The Update Driver Software dialog box will pop up. Click Browse my computer for driver software:

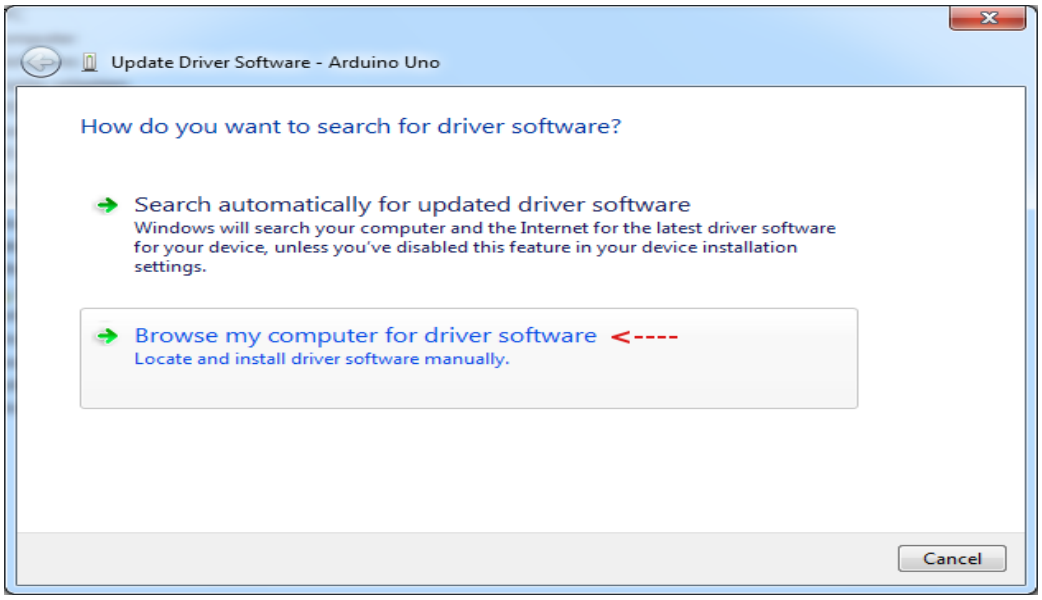


Fig 3.2.7.3.2 Update Driver Software –Arduino Uno

Next, click the Browse**...** button:

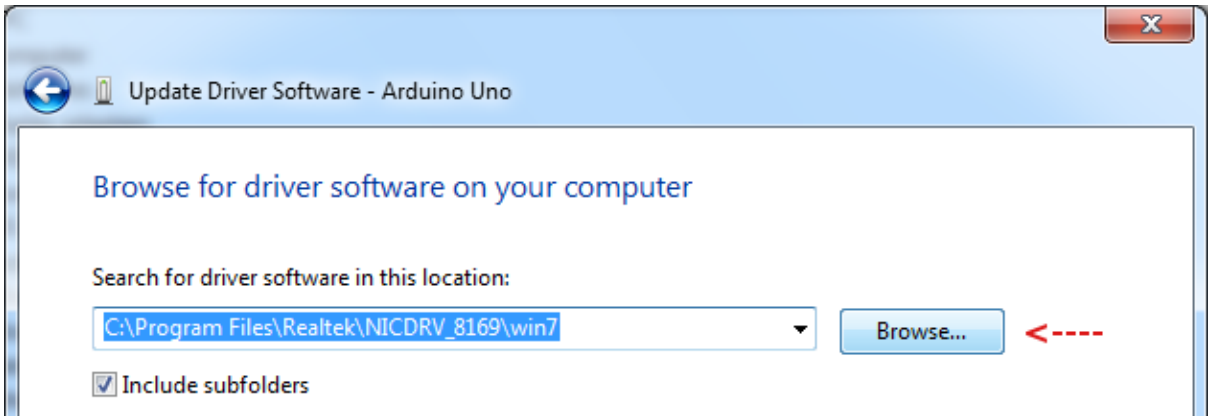


Fig 3.2.7.3.3 arduino location

Navigate to the driver's folder in the Arduino folder that you downloaded:

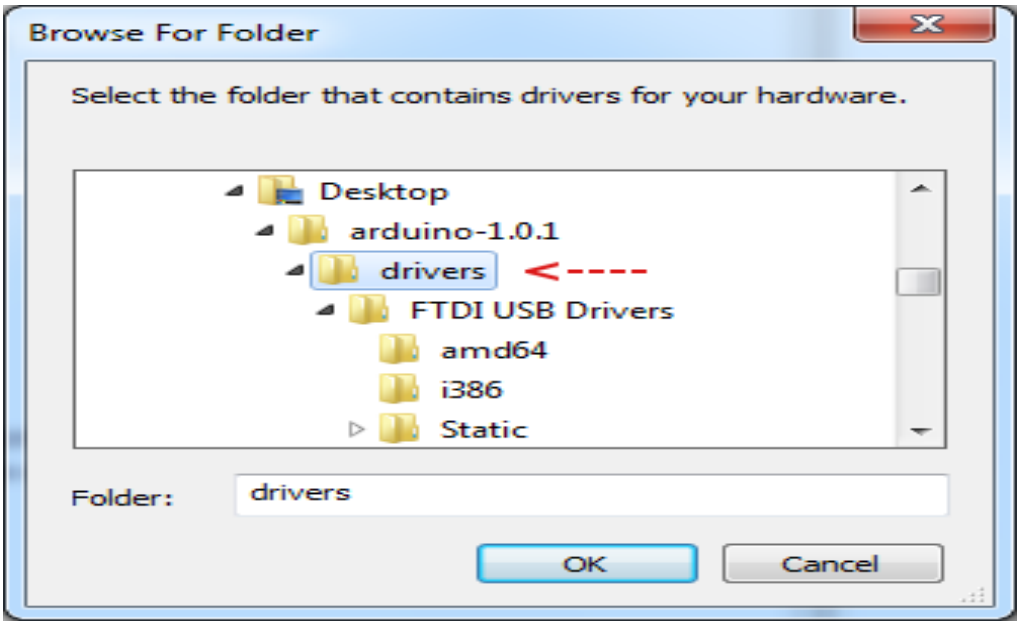


Fig 3.2.7.3.4 software browses for folder

After selecting the driver folder, click the Next button:

In the dialog box that pops up, click Install this driver software anyway:

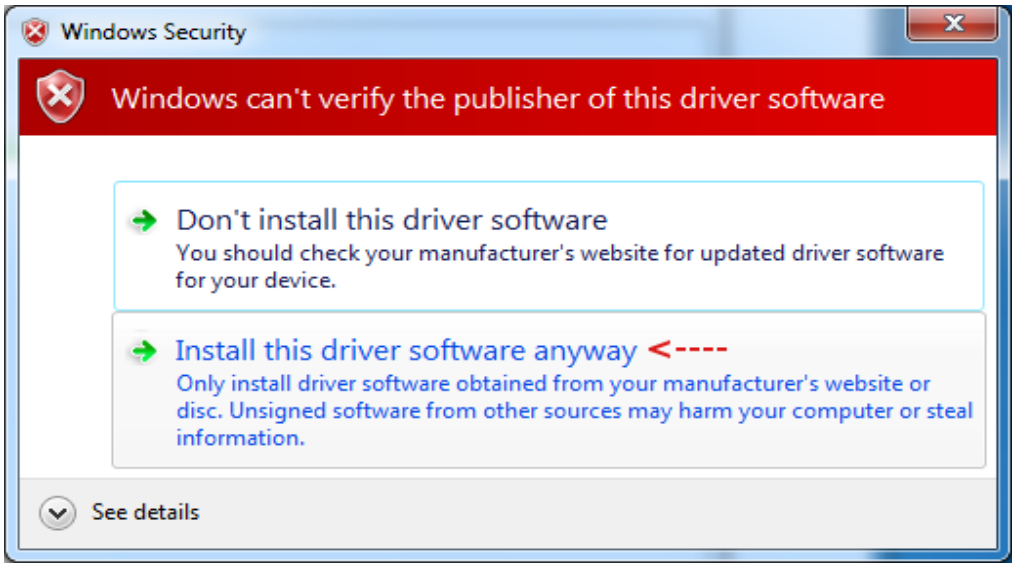


Fig 3.2.7.3.5 pop up of installing software

After some time, the driver installation will finish and you will see the following dialog box. Take note of the port that the Arduino was configured as. In this case it was COM3

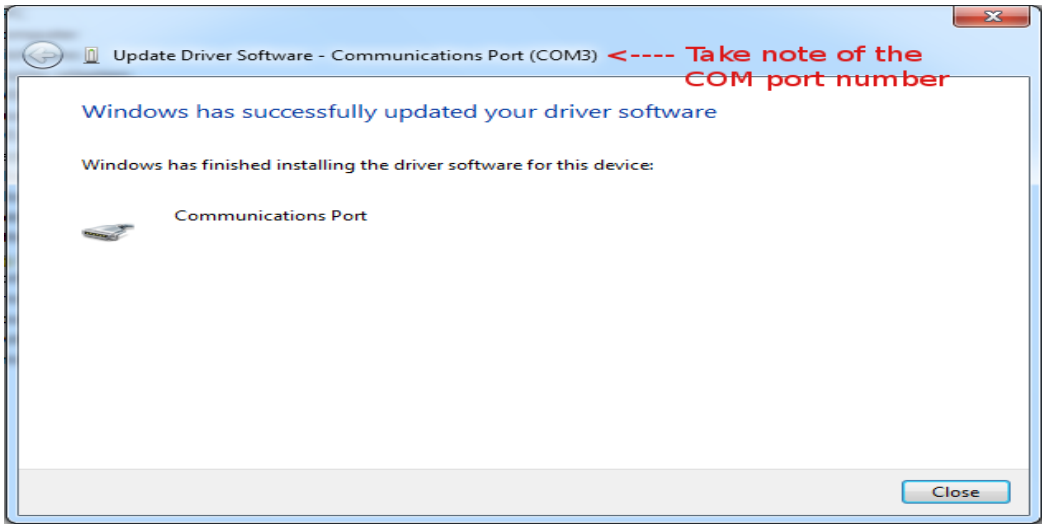


Fig 3.2.7.3.6 COM3 port number window

3.3 AllThingsTalk:

3.3.1 Introduction:

AllThingsTalk Platform gives you the tools to deliver data, connectivity and application services to your IoT audience.

3.3.2 How AllThingsTalk works:

Maker is the web interface of the AllThingsTalk IoT Cloud. It allows you to explore the possibilities of the AllThingsTalk Cloud, connect your devices and build your own IoT prototypes.

AllThingsTalk Maker (ATT Maker) is free and easy to use platform. It provides almost everything to create real Internet of Things solution:

- Full GUI administration

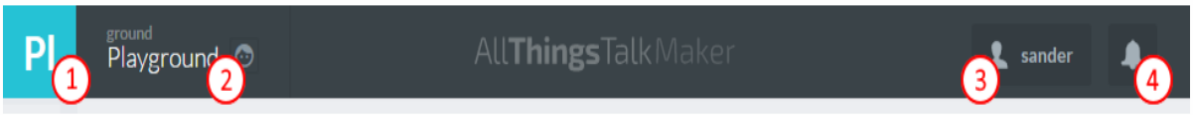
- Device management
- Charts and other visual controls for visual presentation
- Server-side rules engine for notification and actuation
- MQTT and HTTP Rest protocols.

3.3.3 Get Started with Maker

Maker is the web interface of the AllThingsTalk Cloud. It allows you to explore the possibilities of the AllThingsTalk Cloud, connect your devices and build your own IoT prototypes.

3.3.3.1 Onboarding

After you have created an account and logged in, the landing page will give you an overview of all the Grounds you own or are a member of. Once you enter a certain ground, you have several menu's available to you to navigate and interact with all the devices in that ground.



1	Ground menu	This menu contains the actions (delete, view activity, toggle notifications) as well as general info about this ground.
2	Grounds overview	A dropdown menu containing all ground you own or are a member of.
3	Account menu	This menu is always available on the main page as it contains all general info (non ground specific). You can get an overview of *all* your • rules • gateways • devices across all grounds in your account. Furthermore it contains info on your user profile as well as several links to documentation.
4	Notifications	Your personal notifications will appear here.

Fig 3.3.3.1.1 AllThingsTalk Playground options

3.3.3.2 About grounds

Grounds are distinct areas of an AllThingsTalk Space that groups relevant IoT resources together, allowing for easier access management, delegation and separation of concerns.

A ground allows people with common interest to build and share their IoT devices, assets and data either internally within a team, externally to a selected audience or publicly with everyone. This enables better collaboration between people who are building and maintaining common IoT resources.

3.3.3.3 Your environment

You can organize your IoT product and applications within multiple usage contexts, for example projects, business units, customer segments, facilities, apartments, rooms, homes, common spaces, locations and so on…

3.3.3.4 Own and joined grounds

You are granted full access to manage all resources of a ground that you own, while you have limited access to the resources of grounds that you have joined as member.

3.3.3.5 Creating new ground

As a user, you can add new grounds to your environment, in order to clearly distinct one usage area from another. In order to create new ground, all you need is a valid name (title). The system will then create a public but unlisted ground.

3.3.4 Manage your ground

3.3.4.1 Changing ground title

You can modify the title of a ground that you own, so it better tells what's behind it. Once you change it, the new title becomes available to everyone who is visiting your ground.

3.3.4.2 Allowing members to add and manage ground resources

As a ground owner, you can allow people in your ground to co-manage devices and rules together with you.

3.3.4.3 Deleting ground

You can delete any ground that you own, given you have previously removed all native devices from that ground.

Once the ground is deleted, none of the previous members will be able to access or revive it.

3.3.4.4 Sharing a ground link

When you create a ground, it's by default made public. Once you share the URL of the ground with another user that you trust, they will be able to view the ground name and pulse.

3.3.5 Manage members

People that collaborate with you in a ground are called members. A ground can have unlimited number of members.

3.3.5.1 Viewing members

Being a ground owner or member, you can view basic info (such as usernames) of all members of that ground.

3.3.5.2 Adding members

As a ground owner, you can add ground members by providing their username or email address.

- If such email address is not associated with an existing account, system will send a personal invitation link by email. Recipient will therefore be able to complete their account credentials and become a regular user.
- If user account with such email address already exist, system will add that user as ground member without sending any notification.

3.3.5.3 Removing members

As the ground owner, you can always decide to remove a member from the ground, so she doesn't have the access to the ground resources.

3.3.6 Ground tokens

Ground tokens allow client applications to access multiple resources within a ground. Best example is a client that wants to access all devices and their assets within a single ground.

3.3.6.1 Creating new ground token

When you create a new ground, a new ground token will be generated. You can create unlimited number of ground tokens for each ground. By having multiple ground tokens, you can delegate and manage ground access for multiple client applications.

3.3.6.2 Viewing ground tokens

For each ground, you can view all the generated ground tokens.

3.3.6.3 Revoking ground token

After you revoke a ground token, system will prevent any access to ground resources to the clients that continue to present the revoked token.

3.3.6.4 Join ground as member

By default, no one can join ground if not explicitly invited (or added) by the ground owner. Specifically, when you create a ground that anyone can join, and share the ground's URL with an audience, anyone with an account can join your ground and therefore become it's member.

3.3.7 Resource management permissions

As a group of people, you can do more when you're working in team. With resource management permissions, you allow people in your ground to manage devices and rules just like you do.

3.3.7.1 Setting resource management permissions

As a ground owner, you can allow people in your ground to co-manage devices and rules together with you by going to ground *Settings*, and changing *Resource management permissions* to members. After that, all ground members will be able to manage devices and rules in your ground.

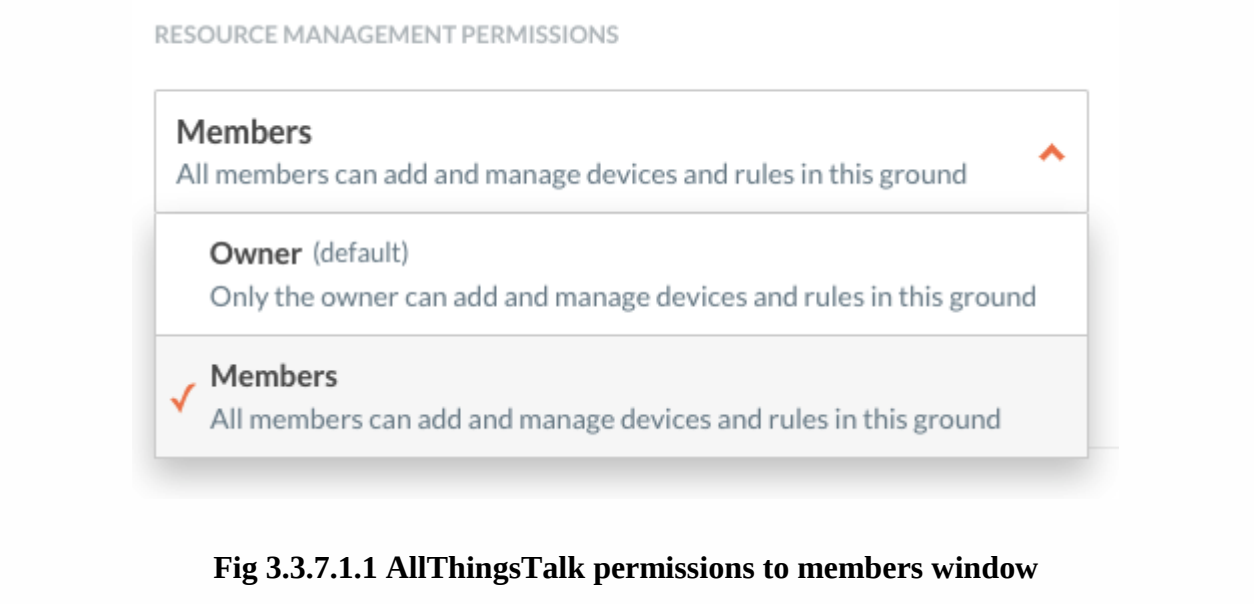


Fig 3.3.7.1.1 AllThingsTalk permissions to members window

3.3.7.2 Manage devices

Given *Resource management permissions* are set to Members, all members can add new devices to the ground, so it's easier to delegate that task between people in the ground.

Besides adding devices, after changing these permissions everyone in the ground are able to manage any existing device in the ground. More specifically members will be able to change device title and description,

- delete device,
- create new, update or delete an asset,
- publish asset state or actuate an asset,
- view device settings and
- view authentication details
- generate new or delete existing token
- view connectivity details
- connect to or disconnect device from network
- view payload format settings

- request and receive an export of historical data
- delete historical data

3.3.7.3 Sharing devices

As a ground member, you will be able to help out your team in sharing valuable data and therefore let other people leverage on data being collected in your common ground. This means that as a member you can

- share a device to another ground
- view native device's remote grounds
- remove native device from remote ground
- remove a device shared to your ground

3.3.7.4 Manage rules

Given resource management permissions are set to Members, as a ground member, you will be able to

- create rules
- update a rule
- view a rule
- list rules
- update a rule meta
- delete a rule
- pause a rule
- resume a rule

3.3.8 Devices

A device is logical container of physical measurements. It includes hardware and software that directly interacts with the world. They connect to a network to communicate with each other or to cloud. Devices might be ***directly*** or ***indirectly*** (through a gateway) connected to the cloud.

Sensors and **Actuators** are connected to the device (or integrated in its hardware). These sensors and actuators are called the Assets of the device and are graphically represented in Maker using Controls on a Pinboard.

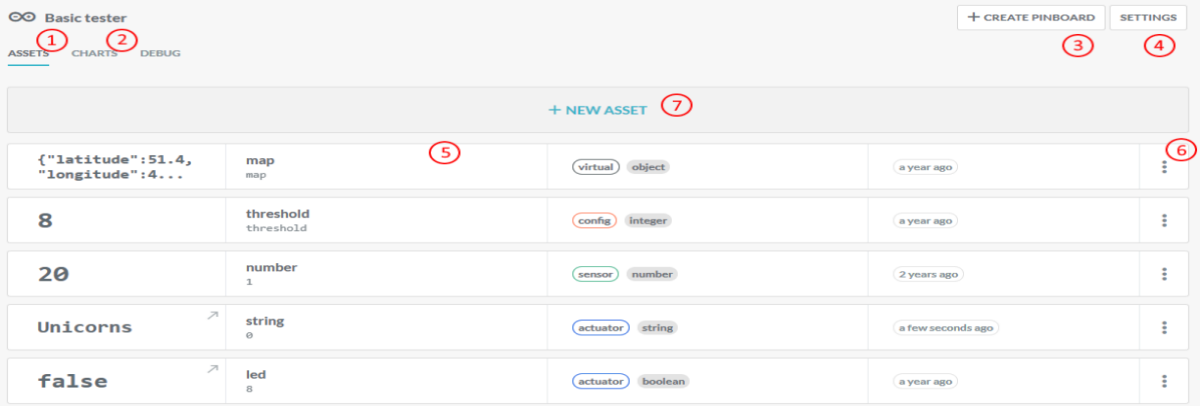


Fig 3.3.8.1 AllThingsTalk Basic tester window

1	Asset overview	List of all assets for this device.
2	Charts	Graph showing historical data of one or more assets.
3	Create pinboard	Creates a default pinboard for this device, pinning one default control for each asset.
4	Device settings	View device setup and settings.
5	Asset	One asset block, showing this assets parameters and value.
6	Asset actions	Delete, rename or view the asset details. For actuators, you can also send a command here.
7	New asset	Create a new asset.

Fig 3.3.8.2 list of basic tester in AllThingsTalk

3.3.8.1 Create device

Lora WAN

To create a device

- Go to the ground in which you want to create it and select the **Devices** tab on the left
- Click the large *New device* button

- Select the type of device you want to create
- Select your LoRaWAN provider
- Enter a name for your device
- Enter the **DEVEUI**

Under **Add Keys**, enter the

- DevAddr
- Network Session Key
- App Session key

They can be found under the Lora WAN Server provider account you have selected. The keys are not mandatory but it is useful to enter them in the screen below. This will allow us to auto-generate code for you and it might come in handy for your own reference.

- Hit CONNECT

Wi-Fi / LAN

AllThingsTalk Kits
To create a device

- Go to the ground in which you want to create it and select the **Devices** tab on the left
- Click the large *New device* button
- Select the type of device you want to create

Your own hardware

To create a device

- Go to the ground in which you want to create it and select the **Devices** tab on the left
- Click the large *New device* button
- Select the type of device you want to create
- Enter a name for your device

Your device is now created and you are provided with its **Device id**, on top of your **Client Id** and **Client key**.

3.3.8.2 Delete device

When you delete a device, all its assets will be deleted and all rules connected to the device become invalid! You might want to delete any rules linked to the device as well.

3.3.9 About pinboards

Pinboards let you visualize collected data by displaying set of visual controls that show values from either a single device, group of devices, an asset or group of assets. You can visualize data coming from any device in ground, no matter if you own it or not.

3.3.9.1 Controls and pins

Controls are UI elements that either visually displays an asset's current state or 'last known' data (e.g. label, map), or allows you to actuate a device (e.g. button, slider). It usually displays the timestamp for the last known state

Every control is wrapped into a pin, which determines its name, position and size. There are number of predefined sizes that depend on the control that it wraps.

3.3.9.2 Listing pinboards

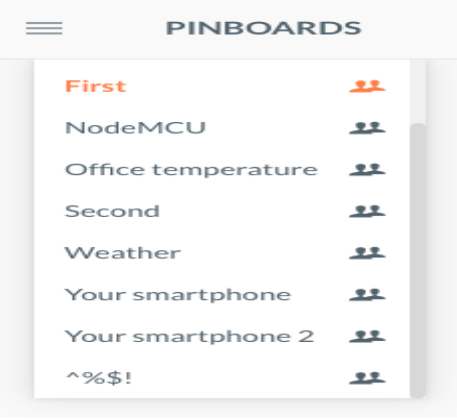


Fig 3.3.9.2.1 Listing pinboards

As a ground owner you are able to list

- your pinboards
- pinboards of all ground members

As a ground member, you are able to list

- your pinboards
- pinboards of the ground owner

3.3.9.3 Viewing pinboard

As a user you can view any pinboard that is listed in your ground. Viewing pinboard allows you to view any data that has been made visible through the pinned controls. Most of the controls update data in real-time, so whenever a device updates an asset state, you will be able to see the change.

3.3.9.4 Creating new pinboard

You can create as many pinboards as you like, and arrange them up to your needs. Any pinboard created by ground owner is accessible by all the members in that ground.

To create a pinboard, while in ground:

- Go to Pinboards
- Choose `+ NEW PINBOARD` to open the *New pinboard* modal
- Enter *Pinboard name* and choose `CREATE`

3.3.9.5 Arranging pinboard

For the pinboard that you have created, you can change its name, pin title, position or size. To arrange a pinboard, while in a ground:

- Choose the pinboard you want to arrange
- Choose `ARRANGE`
- Either pin new control, or change name, position of pins or size

- After you finish arranging it, SAVE the pinboard and it will display the data as you have arranged it.

3.3.9.6 Pinning controls

If you want to add new data visualizations to an existing pinboard, you can pin a new control at any free space in the Pinboard. To allow full customization of what you want to visualize you can pin the same asset multiple times on the same pinboard using different controls, or you can also pin the same asset on multiple pinboards.

To pin a new control:

- While in Arrange mode of a pinboard
- Choose + NEW PIN

Follow the steps, they are different depending the control you want to pin

3.3.9.7 Deleting pinboard

You can delete any pinboard that you have created. Once deleted, no one will be able to access it or revoke it.

3.4 Arduino UNO

Arduino is a standard term for a software company, project, and user community that designs and produces computer open-source hardware, open-source software, and microcontroller based kits for producing digital devices and interactive objects that can sense and monitor the physical devices.

The project is based on microcontroller board designs, produced by many vendors, using various microcontrollers. These systems provide sets of digital and analog I/O pins that can interconnect to various expansion boards (termed shields) and other systems. The board's features are serial communication interfaces, including universal serial bus (USB) on some designs, for loading programs from personal computers. For programming the microcontrollers, the Arduino project generates an integrated development environment (IDE) based on a programming language named processing, which also assists the languages C and C++.

The first Arduino was introduced in 2005, aiming to issue a low cost, easy way for professionals to generate devices that interact with their environment using sensors and actuators. Common examples of such devices are intended for beginner hobbyists including simple robots, thermostats, and motion detectors.

Arduino Uno boards are available commercially in preassembled form, or as do-it-yourself kits. The hardware design specifications are openly obtainable, allowing the Arduino boards to be produced by anyone. Ad fruit industries are estimated in mid-2011 that over 300,000 official Arduino had been commercially produced, and in2013 that 700,000 official boards were in user's hands.

Fig 3.4.1 Arduino UNO Board

Table 3.4.1 Pin Description

Pin Category	Pin Name	Details
Power	Vin, 3.3V, 5V, GND	Vin: Input voltage to Arduino when using an external power source. 5V: Regulated power supply used to power

		microcontroller and other components on the board. 3.3V: 3.3V supply generated by on-board voltage regulator. Maximum current draw is 50mA. GND: ground pins.
Reset	Reset	Resets the microcontroller.
Analog Pins	A0 – A5	Used to provide analog input in the range of 0-5V
Input/output Pins	Digital Pins 0 - 13	Can be used as input or output pins.
Serial	0(Rx), 1(Tx)	Used to receive and transmit TTL serial data.
External Interrupts	2, 3	To trigger an interrupt.
PWM	3, 5, 6, 9, 11	Provides 8-bit PWM output.
SPI	10 (SS), 11 (MOSI), 12 (MISO) and 13 (SCK)	Used for SPI communication.
Inbuilt LED	13	To turn on the inbuilt LED.
TWI	A4 (SDA), A5 (SCA)	Used for TWI communication.
AREF	AREF	To provide reference voltage for input voltage

Table 3.4.2 Arduino Uno Technical Specifications

Microcontroller	ATmega328P – 8 bit AVR family microcontroller
Operating Voltage	5V
Recommended Input Voltage	7-12V
Input Voltage Limits	6-20V
Analog Input Pins	6 (A0 – A5)
Digital I/O Pins	14 (Out of which 6 provide PWM output)
DC Current on I/O Pins	40 mA
DC Current on 3.3V Pin	50 mA
Flash Memory	32 KB (0.5 KB is used for Bootloader)
SRAM	2 KB
EEPROM	1 KB
Frequency (Clock Speed)	16 Hz

3.5 Water Flow Sensor

Water flow sensor is an easy-to-use, cost-effective high level water flow rate detecting sensor. Water flow sensor consists of a plastic valve from which water can pass. A water rotor along with a hall effect sensor is present the sense and measures the water flow.

When water flows through the valve it rotates the rotor. By this, the change can be observed in the speed of the motor. This change is calculated as output as a pulse signal by the hall effect sensor. Thus, the rate of flow of water can be measured.

The main working principle behind the working of this sensor is the Hall effect. According to this principle, in this sensor, a voltage difference is induced in the conductor due to the rotation of the rotor. This induced voltage difference is transverse to the electric current. The water flow sensor can be used with hot waters, cold waters, warm waters, clean water, and dirty water also. These sensors are available in different diameters, with different flow rate ranges.

Table 3.5.1 Water flow sensor specifications

Cumulative flow conversion	1L(H2O)=516 pulse, accuracy 10%
Flow characteristic	F=10Q, F:HZ, Q:L/min, positive or negative 10%
water quality requirement	drinking water health standards, 0~60 degree celsius
Size	1/2Inch Plastic
flow range	1~30L/MIN
Hydraulic pressure	less than or equal to 1.75 MPa

Fig 3.5.1 Water flow sensor

3.6 Ultra Sonic Sensor

The HC-SR04 Ultrasonic distance sensor consists of two ultrasonic transducers. The one acts as a transmitter which converts electrical signal into 40 KHz ultrasonic sound pulses. The receiver listens for the transmitted pulses. If it receives them it produces an output pulse whose width can be used to determine the distance the pulse travelled.

The sensor is small, easy to use in any robotics project and offers excellent non-contact range detection between 2 cm to 400 cm (that's about an inch to 13 feet) with an accuracy of 3mm. Since it operates on 5 volts, it can be hooked directly to an Arduino

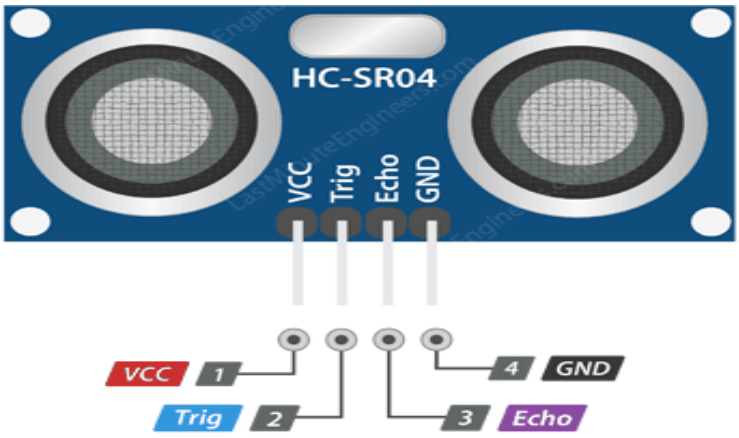


Fig 3.6.1 Ultra sonic sensor

Table 3.6.1 Ultrasonic Sensor Pin Configuration

Pin Number	Pin Name	Description
1	Vcc	The Vcc pin powers the sensor, typically with +5V
2	Trigger	Trigger pin is an Input pin. This pin has to be kept high for 10us to initialize measurement by sending US wave.
3	Echo	Echo pin is an Output pin. This pin goes high for a period of time which will be equal to the time taken for the US wave to return back to the sensor.
4	Ground	This pin is connected to the Ground of the system.

Ultrasonic sensors work by emitting sound waves with a frequency that is too high for a human to hear. These sound waves travel through the air with the speed of sound, roughly 343 m/s. If there is an object in front of the sensor, the sound waves get reflected back and the receiver of the ultrasonic sensor detects them. By measuring how much time passed between sending and receiving the sound waves, the distance between the sensor and the object can be calculated.

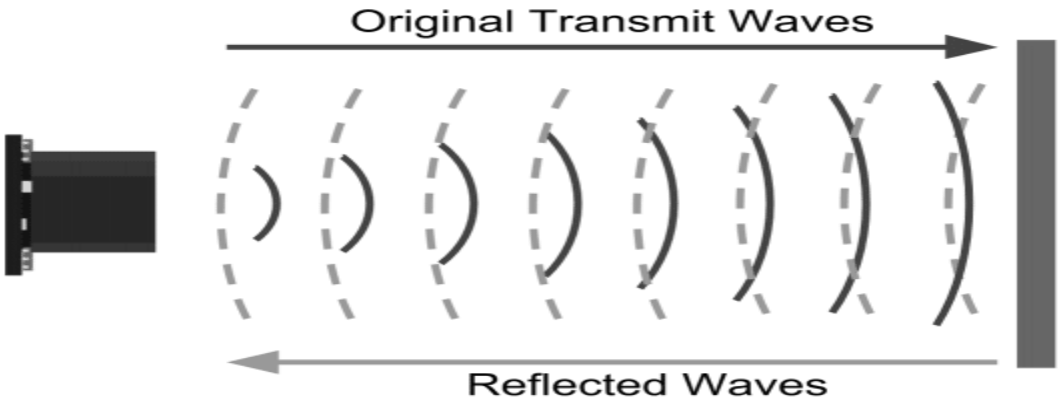


Fig 3.6.2 Working of ultrasonic

At 20°C the speed of sound is roughly 343 m/s or 0.034 cm/µs. Let's say that the time between sending and receiving the sound waves is 2000 microseconds. If you multiply the speed of sound by the time the sound waves traveled, you get the distance that the sound waves traveled.

Distance = Speed x Time

But that is not the result we are looking for. The distance between the sensor and the object is actually only half this distance because the sound waves traveled from the sensor to the object and back from the object to the sensor. So you need to divide the result by two.

Distance (cm) = Speed of sound (cm/µs) × Time (µs) / 2

And so for the example this becomes:

Distance (cm) = 0.0343 (cm/µs) × 2000 (µs) / 2 = 34.3 cm

3.6.1 HC-SR04 Sensor Features

- Operating voltage: +5V
- Theoretical Measuring Distance: 2cm to 450cm
- Practical Measuring Distance: 2cm to 80cm
- Accuracy: 3mm
- Measuring angle covered: <15°
- Operating Current: <15mA
- Operating Frequency: 40Hz

3.7 Node MCU

The All new NodeMCU ESP8266 V3 Lau CH340 Wife Dev. Board is a fast leading edge low-cost Wife technology. Modern high-level mature LUA based technology. It is an integrated unit with all available resources on board. It is super simple to complement your existing Arduino projects or any development board that has I/O pins available.

Modern Internet development tools such as Node.js can take advantage the NodeMCU with the built-in API to put your idea on the fast track immediately. NodeMCU is

built based on the mature ESP8266 technology to take advantage of the abundant resources available on the web.

NodeMCU has ESP-12 based serial WiFi integrated on board to provide GPIO, PWM, ADC, I2C and 1-WIRE resources at your fingertips, built-in USB-TTL serial with super reliable industrial strength CH340 for superior stability on all supported platforms. This module is one of the cheapest available wifi-modules in the market. V3 or Version3 is the latest version of this module. This tutorial, however, will facilitate you to connect all the versions of ESP8266 NodeMcu, i.e V1, V2 or V3.

1. Arduino-like hardware IO – Advanced API for hardware IO, which can dramatically reduce the redundant work for configuring and manipulating hardware. Code like Arduino, but interactively in Lua script.
2. Nodejs style network API Event-driven API for network applications, which facilitates developers writing code running on a 5mm5mm sized MCU in Nodejs style. Greatly speed up your IOT application developing process.
3. Development Kit The Development Kit based on ESP8266, integrates GPIO, PWM, IIC, 1-Wire and ADC all on one board. Power your development in the fastest way combination with NodeMCU Firmware!

Fig 3.7.1 Node MCU

3.8 Power Supply

A power supply is an electrical device that supplies electric power to an electrical load. The primary function of a power supply is to convert electric current from a source to the correct voltage, current, and frequency to power the load. As a result, power supplies are sometimes referred to as electric power converters. Some power supplies are separate standalone pieces of equipment, while others are built into the load appliances that they power. Examples of the latter include power supplies.

All power supplies have a power input connection, which receives energy in the form of electric current from a source, and one or more power output connections that deliver current to the load. The source power may come from the electric power grid, such as an electrical outlet, energy storage devices such as batteries or fuel cells, generators or alternators, solar power converters, or another power supply. Adjustable power supplies allow the output voltage or current to be programmed by mechanical controls A power supply is a component that supplies power to at least one electric load. Typically, it converts one type of electrical power to another, but it may also convert a different form of energy such as solar, mechanical, or chemical - into electrical energy.

A power supply provides components with electric power. The term usually pertains to devices integrated within the component being powered. For example, computer power supplies convert AC current to DC current and are generally locate.

Fig 3.8.1 Power supply

4. DESIGN

4. 1 Introduction

The Design phase is when you build the plan for how you will take your project through the rest of the SDL process from implementation, to verification, to release. During the Design phase you establish best practices to follow for this phase by way of functional and design specifications, and you perform risk analysis to identify threats and vulnerabilities in your software.

4.2 Feasibility Study

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential. Three key considerations involved in the feasibility analysis are

- Economic Feasibility
- Social Feasibility
- Technical Feasibility

4.2.1 Economical Feasibility

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus, the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

4.2.2 Technical Feasibility

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

4.2.3 Social Feasibility

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

4.3 Data Flow Diagram

A data flow diagram is graphical tool used to describe and analyze movement of data through a system. These are the central tool and the basis from which the other components are developed. The transformation of data from input to output, through processed, may be described logically and independently of physical components associated with the system. These are known as the logical data flow diagrams.

The physical data flow diagrams show the actual implements and movement of data between people, departments and workstations. A full description of a system actually consists of a set of data flow diagrams. Using two familiar notations Yourdon, Gane and Sarson notation develops the data flow diagrams. Each component in a DFD is labeled with a descriptive name. Process is further identified with a number that will be used for identification purpose. The development of DFD'S is done in several levels. Each process in lower level diagrams can be broken down into a more detailed DFD in the next level. The lop-level diagram is often called context diagram. It consists a single process bit, which plays vital role in studying the current

system. The process in the context level diagram is exploded into other process at the first level DFD.

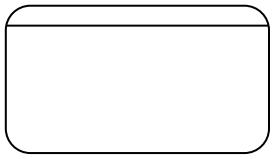
The idea behind the explosion of a process into more process is that understanding at one level of detail is exploded into greater detail at the next level. This is done until further explosion is necessary and an adequate amount of detail is described for analyst to understand the process. Larry Constantine first developed the DFD as a way of expressing system requirements in a graphical from, this lead to the modular design. A DFD is also known as a "bubble Chart" has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design. So it is the starting point of the design to the lowest level of detail. A DFD consists of a series of bubbles joined by data flows in the system.

4.3.1 DFD Symbols

In the DFD, there are four symbols

1. A square defines a source (originator) or destination of system data.
2. An arrow identifies data flow. It is the pipeline through which the information Flows.
3. A circle or a bubble represents a process that transforms incoming data flow into Outgoing data flows.
4. An open rectangle is a data store, data at rest or a temporary repository of data.

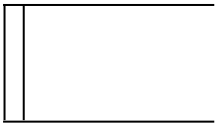
- Process that transforms data flow.



- Source or Destination of data

- Data flow

- Data Store



4.3.2 Constructing A DFD

Several rules of thumb are used in drawing DFD'S:

1. Process should be named and numbered for an easy reference. Each name should be representative of the process.
2. The direction of flow is from top to bottom and from left to right. Data traditionally flow from source to the destination although they may flow back to the source. One way to indicate this is to draw long flow line back to a source. An alternative way is to repeat the source symbol as a destination. Since it is used more than once in the DFD it is marked with a short diagonal.
3. When a process is exploded into lower level details, they are numbered.The names of data stores and destinations are written in capital letters. Process and dataflow names have the first letter of each work capitalized.

A DFD typically shows the minimum contents of data store. Each data store should contain all the data elements that flow in and out. Questionnaires should contain all the data elements that flow in and out. Missing interfaces redundancies and like is then accounted for often through interviews.

4.3.3 Silent Features of DFD'S:

1. The DFD shows flow of data, not of control loops and decision are controlled considerations do not appear on a DFD.
2. The DFD does not indicate the time factor involved in any process whether the dataflow take place daily, weekly, monthly or yearly.
3. The sequence of events is not brought out on the DFD.

4.4 Types of Dataflow Diagrams

1. Current Physical
2. Current Logical
3. New Logical
4. New Physical

1. Current Physical

In Current Physical DFD process label include the name of people or their positions or the names of computer systems that might provide some of the overall system processing label includes an identification of the technology used to process the data.

2. Current Logical

The physical aspects at the system are removed as much as possible so that the current system is reduced to its essence to the data and the processors that transforms them regardless of actual physical form.

3. New Logical

This is exactly like a current logical model if the user were completely happy with the user were completely happy with the functionality of the current system but had problems with how it was implemented typically through the new logical model will differ from current logical model while having additional functions, absolute function removal and inefficient flows recognized.

4. New Physical

The new physical represents only the physical implementation of the new system.

4.4.1 Data Flow Diagram

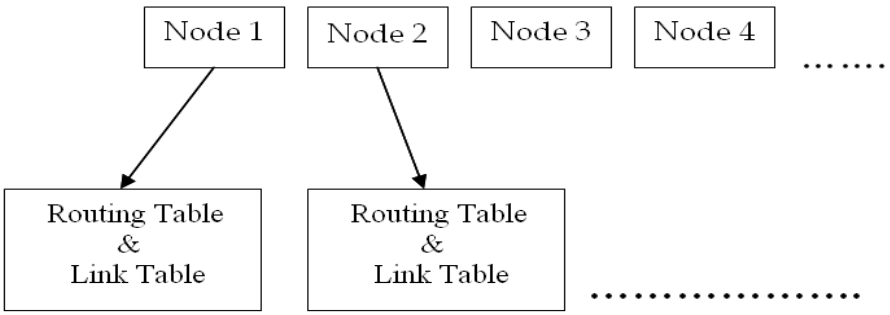


Fig 4.4.1.1 Updating the Node Details

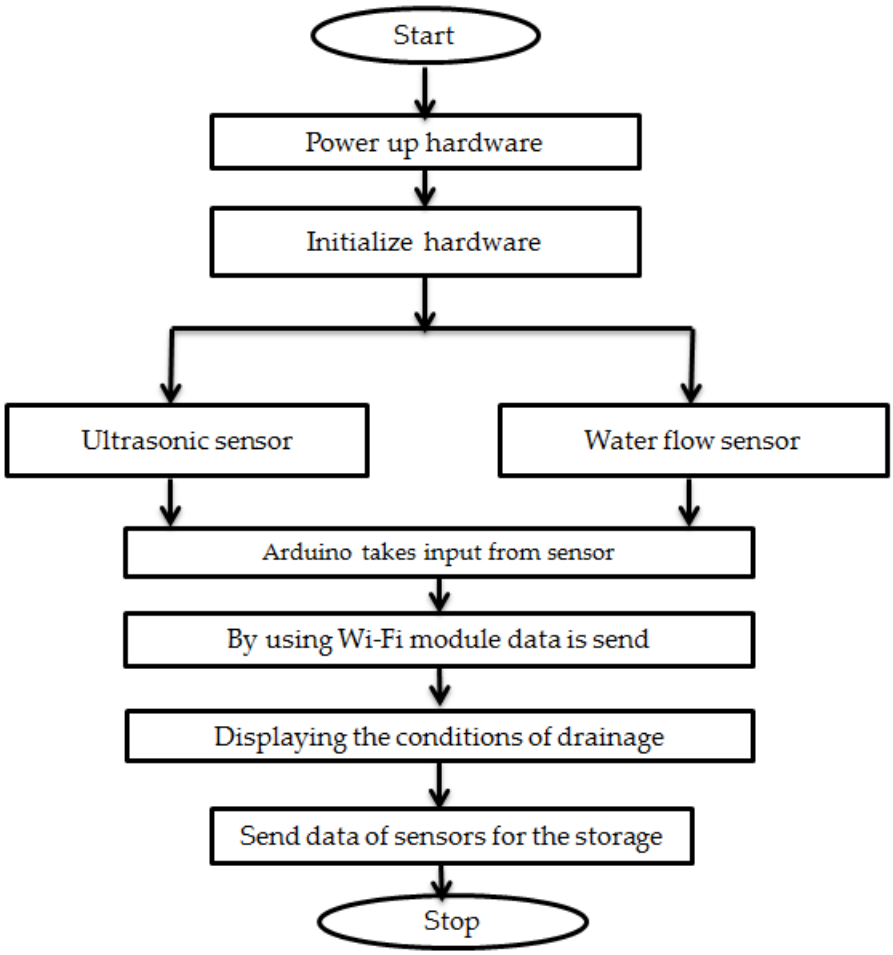


Fig 4.4.1.2 Data flow diagram

4.5 UML Diagrams

4.5.1 Use Case Diagram

In many design processes, the use case diagram is the first that designers will work with when starting a project. This diagram allows for the specification of high-level user goals that the system must carry out. More formally, a use case is made up of a set of scenarios. Each scenario is a sequence of steps that encompass an interaction between a user and a system.

The use case brings scenarios together that accomplish a specific goal of the user. A use case can be specified by textually describing the steps required and any alternative actions at each step. The use case diagram allows a designer to graphically show these use cases and the actors that use them. An actor is a role that a user plays in the system.

It is important to distinguish between a user and an actor (better thought of as a role). A user of the system may play different roles through the course of his, her or its job (since an actor may be another system).

A use case diagram at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams as well.

4.5.1.1 Application

It is a very well-known adage that "A picture is worth a thousand words". With regards to use case diagrams, that is exactly what they are meant to do. While a use case itself might drill into a lot of detail about every possibility, a use case diagram can help provide a higher-level view of the system. It has been said before that "Use case diagrams are the blueprints for your system". They provide the simplified and graphical representation of what the system must actually do.

Due to their simplistic nature, use case diagrams can be a good communication tool for Stakeholders. The drawings attempt to mimic the real world and provide a view for the stakeholder to understand how the system is going to be designed. Siau and Lee conducted research to determine if there was a valid situation for use case diagrams at all or if they were unnecessary. What was found was that the use case diagrams conveyed the intent of the system in a more simplified manner to stakeholders and that they were "interpreted more completely than class diagrams". The purpose of the use case diagrams is simply to provide the high level view of the system and convey the requirements in layman's terms for the stakeholders. Additional diagrams and documentation can be used to provide a complete functional and technical view of the system.

4.5.1.2 Limitations

- As stated above, a use case diagram is not a standalone model, but one that can be used in conjunction with other models as well.
- Since the main purpose is to outline general behavior, the use case diagram should "focus on business goals rather than system goals".
- A use case diagram is not meant to technically outline the functionality of the system, but to provide the business reasoning and outcomes of the system.

4.5.1.3 Actors

A use case defines the interactions between external actors and the system under consideration to accomplish a goal. Actors must be able to make decisions, but need not be human: "An actor might be a person, a company or organization, a computer program, or a computer system hardware, software, or both. Actors are always stakeholders, but many stakeholders are not actors, since they "never interact directly with the system, even though they have the right to care how the system behaves.

For example, "the owners of the system, the company's board of directors, and regulatory bodies such as the Internal Revenue Service and the Department of Insurance" could all be stakeholders but are unlikely to be actors.

Similarly, a person using a system may be represented as different actors because he is playing different roles. For example, user "Joe" could be playing the role of a Customer when using an Automated Teller Machine to withdraw cash from his own account, or playing the role of a Bank Teller when using the system to restock the cash drawer on behalf of the bank.

Actors are often working on behalf of someone else. Cockburn writes that "These days I write 'sales rep for the customer' or 'clerk for the marketing department' to capture that the user of the system is acting for someone else." This tells the project that the "user interface and security clearances" should be designed for the sales rep and clerk, but that the customer and marketing department are the roles concerned about the results.

A stakeholder may play both an active and an inactive role: for example, a Consumer is both a "mass-market purchaser" and a User. In turn, a User is both a "normal operator" and a

"functional beneficiary". For example, when user "Joe" withdraws cash from his account, he is operating the Automated Teller Machine and obtaining a result on his own behalf.

4.5.1.4 Use Cases

When your first outline of the actors is complete, the next step is to look for the system's use cases. The first use cases are very preliminary, and you will doubtless have to change them a few times until they are stable. If the system's vision or requirements are deficient, or if the system analysis is vague, the system's functionality will be unclear.

Therefore, you must constantly ask yourself if you have found the right use cases. Furthermore, you should be prepared to add, remove, combine, and divide the use cases before you arrive at a final version. You will get a better understanding of the use cases once you have described them in detail.

The best way to find use cases is to consider what each actor requires of the system. Remember the system exists only for its users, and should therefore be based on the users' needs. You will recognize many of the actors' needs through the functional requirements made on the system.

For each actor, human or not, ask yourself the following questions:

- What are the primary tasks the actor wants the system to perform?
- Will the actor create, store, change, remove, or read data in the system?
- Will the actor need to inform the system about sudden, external changes?
- Does the actor need to be informed about certain occurrences in the system?
- Will the actor perform a system start-up or shutdown?

The answers to these questions represent the flows of events that identify candidate use cases. Not all constitute separate use cases; some may be modeled as variants of the same use case. It is not always easy to tell what a variant is and what a separate and distinct use case is. However, it will become clearer when you describe the flows of events in detail.

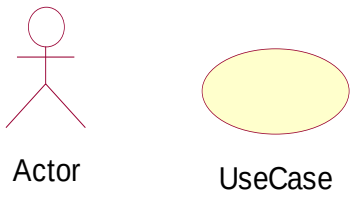
A system can have several possible use-case models. The best way to find the "optimal" model is to develop two or three models, choose the one you prefer, and then develop it further. Developing several alternative models also helps you to understand the system better.

When you have outlined your first use-case model, you should verify that the use-case model addresses all functional requirements. Scrutinize the requirements carefully to ensure that all the use cases meet all the requirements.

4.5.1.5 Outline the Flow of Events

At this point, you should also write a first draft of the flow of events of the use case. Describe each use case's flow of events as brief instants of performance, but do not go into detail. The person who will later specify the use case even if it is you will need this step-by-step description. Start by outlining the basic flow of events, and once you have agreed on that, add alternative flows.

A use case is a set of scenarios that describing an interaction between a user and a system. A use case diagram displays the relationship among actors and use cases. The two main components of a use case diagram are use cases and actors.



An actor is representing a user or another system that will interact with the system you are modeling. A use case is an external view of the system that represents some action the user might perform in order to complete a task.

Contents

- Use cases
- Actors
- Dependency, Generalization, and association relationships

4.5.1.6 Flow of Events

A flow of events is a sequence of transactions (or events) performed by the system. They typically contain very detailed information, written in terms of what the system should do, not how the system accomplishes the task. Flow of events are created as separate files or documents in your favorite text editor and then attached or linked to a use case using the Files tab of a model element.

A flow of events should include:

- When and how the use case starts and ends
- Use case/actor interactions
- Data needed by the use case
- Normal sequence of events for the use case
- Alternate or exceptional flows

4.5.1.7 Construction of use case diagrams

Use-case diagrams graphically depict system behavior (use cases). These diagrams present a high-level view of how the system is used as viewed from an outsider's (actor's) perspective

A use-case diagram can contain:

- Actors ("things" outside the system)
- use cases (system boundaries identifying what the system should do)
- Interactions or relationships between actors and use cases in the system including the associations, dependencies, and generalizations.

4.5.1.8 Relationships in use cases

- **Communication:**

  The communication relationship of an actor in a use case is shown by connecting the actor symbol to the use case symbol with a solid path. The actor is said to communicate with the use case.

- **Uses:**

  A Uses relationship between the use cases is shown by generalization arrow from the use

case.

- **Extends:**

The relationship extends is used when we have one use case that is similar to another use case but does a bit more. In essence it is like subclass

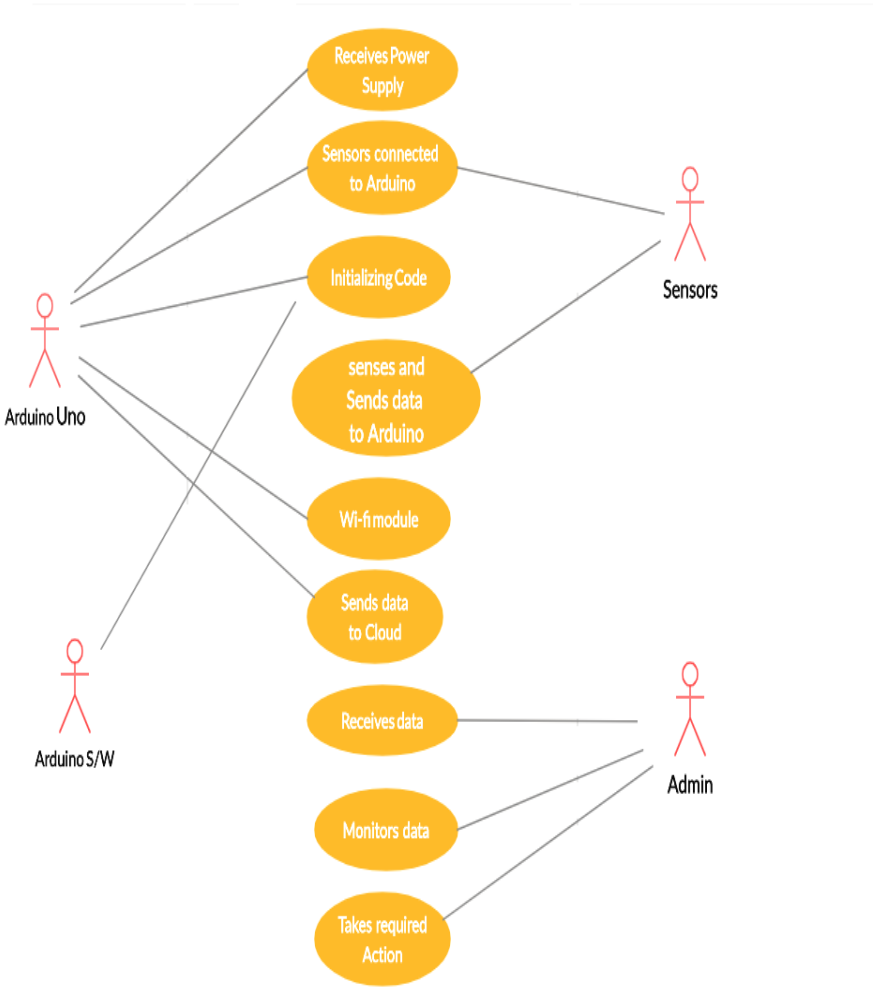


Fig 4.5.1.6.1 Use Case Diagram

4.5.2 Sequence Diagram

Sequence diagrams emphasize the order in which things happen, while collaboration diagrams give more flexibility in their layout. You can use whichever you prefer when drawing

interactions, as both shows the same information. A sequence diagram has two dimensions: The vertical dimension represents the “Time”, while the horizontal dimension represents “different objects”. The vertical line is called “Lifeline”.

The lifeline represents the objects existence during the interaction. Once the use cases are specified, and some of the core objects in the system are prototyped on class diagrams, we can start designing the dynamic behavior of the system. Typically, an Interaction diagram captures the behavior of a single case by showing the collaboration of the objects in the system to accomplish the task. These diagrams show objects in the system and the messages that are passed between them.

There are two types of Interaction Diagrams Sequence and Collaboration. Sequence diagrams emphasize the order in which things happen, while collaboration diagrams give more flexibility in their layout. You can use whichever you prefer when drawing interactions, as both shows the same information. A sequence diagram is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart.

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

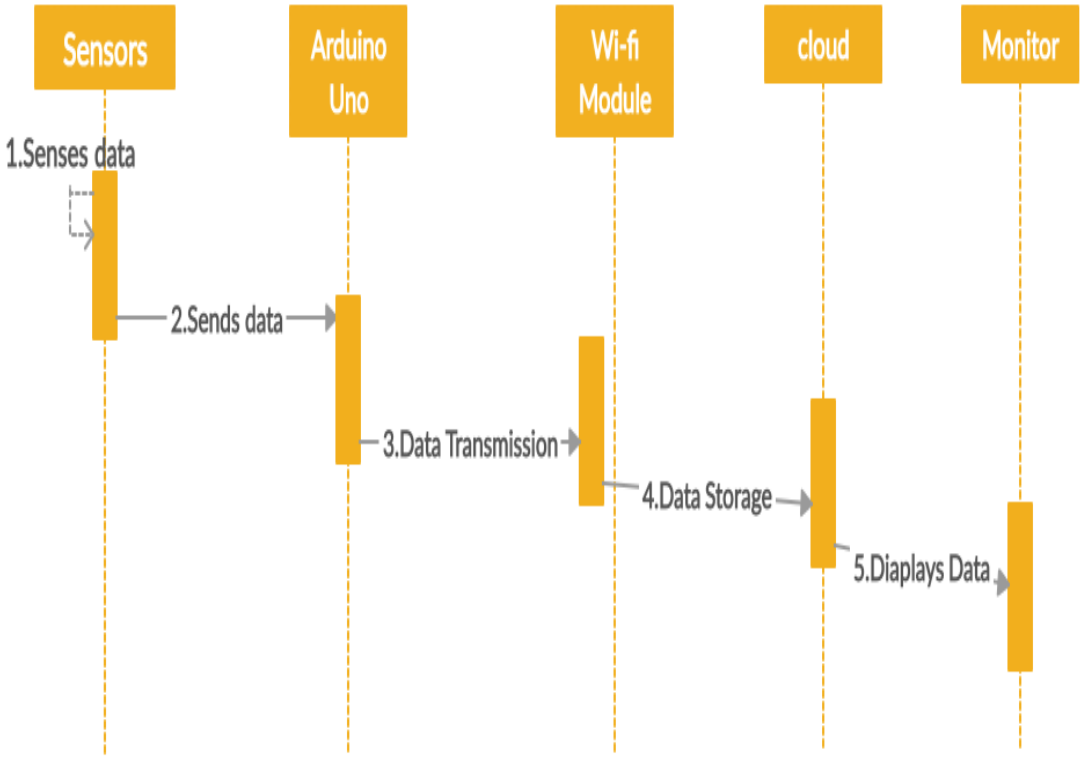


Fig 3.5.2.1 Sequence Diagram

4.5.3 Collaboration Diagram

A role is a slot for an object within a collaboration that describes the type of object that may play the role and its relationships to other roles. However, a sequence diagram does not show the relationships among the roles or the association among the objects. An object role is shown as a vertical dashed line.

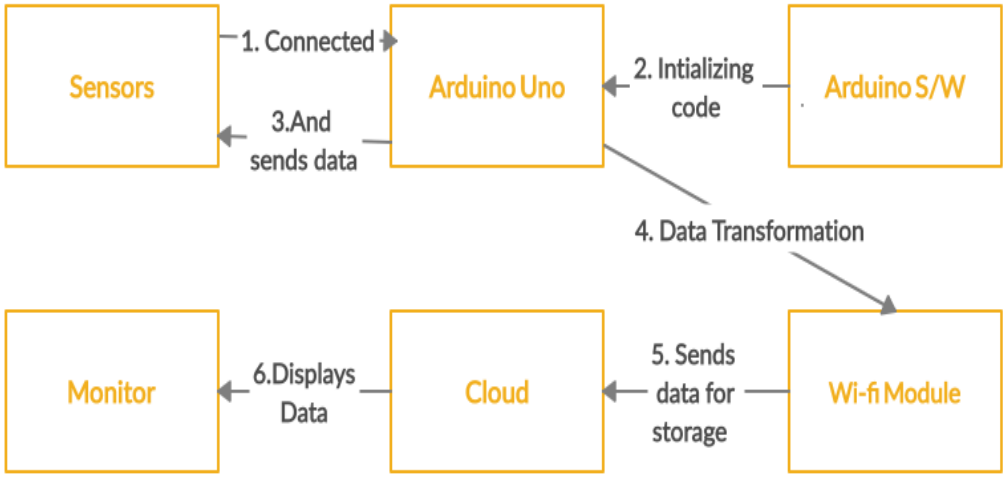


Fig 4.5.2.2 Collaboration Diagram

4.5.4 Class Diagram

Class diagrams are widely used to describe the types of objects in a system and their relationships. Class diagrams model class structure and contents using design elements such as classes, packages and objects. Class diagrams describe three different perspectives when designing a system, conceptual, specification, and implementation. These perspectives become evident as the diagram is created and help solidify the design.

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes.

The class diagram is the main building block of object oriented modelling. It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for modeling. The classes in a class diagram represent both the main objects, interactions in the application and the classes to be programmed.

A class with three sections:

In the diagram, classes are represented with boxes which contain three parts:

- The upper part holds the name of the class
- The middle part contains the attributes of the class
- The bottom part gives the methods or operations the class can take or undertake

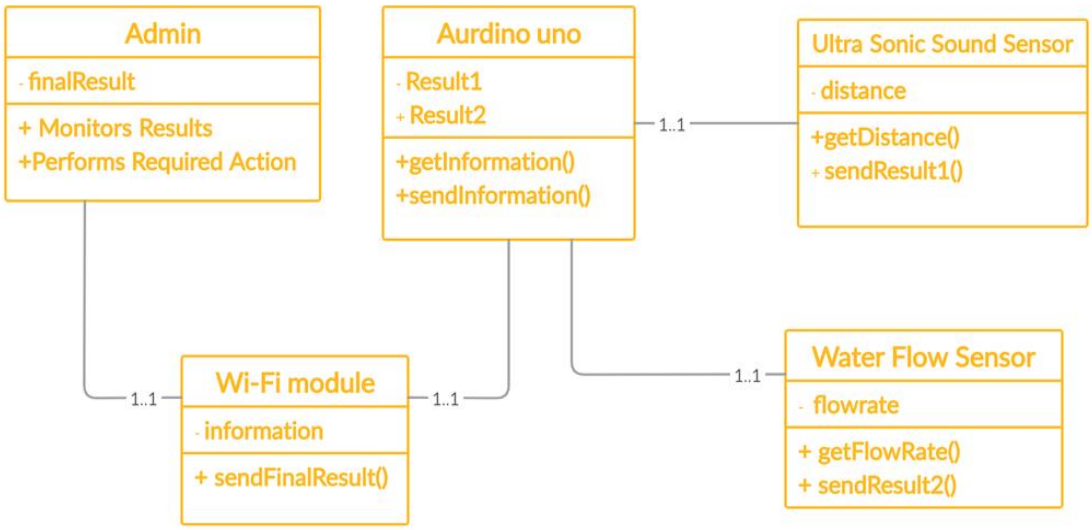


Fig 4.5.4.1 Class Diagram

4.5.5 Component Diagram

Component diagrams are different in terms of nature and behavior. Component diagrams are used to model physical aspects of a system. Now the question is what are these physical aspects? Physical aspects are the elements like executables, libraries, files, documents etc which resides in a node. So, component diagrams are used to visualize the organization and relationships among components in a system. These diagrams are also used to make executable systems.

4.5.5.1 Purpose

Component diagram is a special kind of diagram in UML. The purpose is also different from all other diagrams discussed so far. It does not describe the functionality of the system but it describes the components used to make those functionalities. So, from that point component diagrams are used to visualize the physical components in a system. These components are libraries, packages, files etc. A single component diagram cannot represent the entire system but a collection of diagrams are used to represent the whole.

So, the purpose of the component diagram can be summarized as:

- Visualize the components of a system.
- Construct executables by using forward and reverse engineering.
- Describe the organization and relationships of the components.

Component diagrams are used to describe the physical artifacts of a system. This artifact includes files, executables, libraries etc. So the purpose of this diagram is different, Component diagrams are used during the implementation phase of an application.

This diagram is very important because without it the application cannot be implemented efficiently. A well-prepared component diagram is also important for other aspects like application performance, maintenance etc.

So before drawing a component diagram the following artifacts are to be identified clearly:

- Files used in the system.
- Libraries and other artifacts relevant to the application.
- Relationships among the artifacts.

Now after identifying the artifacts the following points needs to be followed:

- Use a meaningful name to identify the component for which the diagram is to be drawn.
- Prepare a mental layout before producing using tools.
- Use notes for clarifying important points.

The following is a component diagram for Student placement system. Here the artifacts are files. So, the diagram shows the files in the application and their relationships.

So, the following component diagram has been drawn considering all the points mentioned above

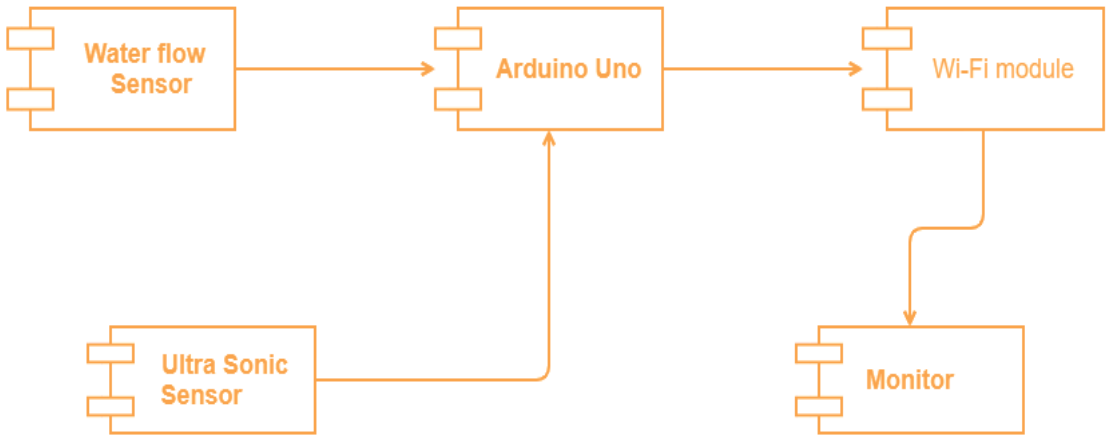


Fig 4.5.5.1.1 Component Diagram

4.5.6 Deployment Diagram

Deployment diagrams are used to visualize the topology of the physical components of a system where the software components are deployed. So, deployment diagrams are used to describe the static deployment view of a system. Deployment diagrams consist of nodes and their relationships.

4.5.6.1 Purpose

The name Deployment itself describes the purpose of the diagram. Deployment diagrams are used for describing the hardware components where software components are deployed. Component diagrams and deployment diagrams are closely related. Component diagrams are used to describe the components and deployment diagrams shows how they are deployed in hardware is mainly designed to focus on software artifacts of a system. But these two diagrams are special diagrams used to focus on software components and hardware components. So most

of the UML diagrams are used to handle logical components but deployment diagrams are made to focus on hardware topology of a system. Deployment diagrams are used by the system engineers.

The purpose of deployment diagrams can be described as:

- Visualize hardware topology of a system.
- Describe the hardware components used to deploy software components.
- Describe runtime processing nodes.

Deployment diagrams are useful for system engineers. An efficient deployment diagram is very important because it controls the following parameters

- Performance
- Scalability
- Maintainability
- Portability

So before drawing a deployment diagram the following artifacts should be identified:

- Nodes
- Relationships among nodes

So the following deployment diagram has been drawn considering all the points mentioned above:

Fig 4.5.6.1.1 Deployment Diagram

5. CODING

5.1 Introduction

Coding is the process of designing, writing, testing, debugging, and maintaining the source code of computer programs. This source code maybe written in one or more programming languages (such as C++, C#, Java, Python, Smalltalk, etc.). The purpose of programming is to create a set of instructions that computers use to perform specific operations or to exhibit desired behaviors. The process of writing source code often requires expertise in many different subjects, including knowledge of the application domain, specialized algorithms and formal logic.

5.2 Sample Code

Arduino sensor code

```
volatile int flow_frequency;
unsigned int l_hour;
unsigned char flowsensor = ;//pinnumber;
unsigned long currentTime;
unsigned long cloopTime;
void flow () // Interrupt function
{
  flow_frequency++;
}
void setup()
{
  pinMode(flowsensor, INPUT);
  pinMode(,OUTPUT);//(echo pinnumber,OUTPUT)
  pinMode(,INPUT);//(trigger pinnumber,INPUT)
  Serial.begin(9600);
```

```
  attachInterrupt(0, flow, RISING);
  sei(); // Enable interrupts
  currentTime = millis();
  cloopTime = currentTime;
}
void loop ()
{
    waterflow();
    ultra();
          delay(100);
}
void waterflow()
{

  {
 // (Pulse frequency x 60 min) / 7.5Q = flowrate in L/hour
    flow_frequency = 0; // Reset Count
  }
  if(){
  Serial.println(""); //sending an number or an char for node MCU code
  delay(100);
  }
  else
  {
   Serial.println("");//sending an number or an char for node MCU code
   delay(1000);
  }
}
void ultra()
```

```
{
  digitalWrite(,LOW);      //sending a pulse //(echo pinnumber,LOW)
  delayMicroseconds(2);
  digitalWrite(,HIGH);//(echo pinnumber ,HIGH)
  delayMicroseconds(10);
  digitalWrite(,LOW);//(echo pinnumber,LOW)
  float t=pulseIn(,HIGH);//(trigger pinnumber,HIGH)
  t=t/2;
  int d=0.0343*t;       //After converting m into cms and s into ms
 // Serial.println("distance:");
  //Serial.println(d);
  if(d>20)
  {
    Serial.println("");//sending an number or an char for node MCU code
  }
  else if(d>10 && d<20)
  {
     Serial.println("");//sending an number or an char for node MCU code
  }
  else if(d<10)
  {
     Serial.println("");//sending an number or an char for node MCU code
  }
  delay(3000);
}
```

NODE MCU code

```
#define CBOR
```

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
#include <ArduinoJson.h>

// Define http and mqtt endpoints
#define http "api.allthingstalk.io"  // API endpoint
#define mqtt "api.allthingstalk.io"  // broker

#include <ATT_IOT.h>
#include <SPI.h>  // required to have support for signed/unsigned long type.

void callback(char* topic, byte* payload, unsigned int length);
WiFiClient espClient;
PubSubClient pubSub(mqtt, 1883, callback, espClient);

char deviceId[ ] = "";//give device ID given by the allthingstalk for your account
char token[] = "";// give maker given by the allthingstalk for your account
ATTDevice device(deviceId, token);
// ATTDevice device;

#ifdef CBOR
  #include <CborBuilder.h>
  CborBuilder payload(device);
#endif

void setup()
{
  pinMode(LED_BUILTIN, OUTPUT);
  digitalWrite(LED_BUILTIN, HIGH);  // Turn the onboard LED off
```

```
  Serial.begin(9600);  // Init serial link for debugging

  // Enter your WiFi credentials here!
  setupWiFi("", "");//give parameters username of WIFI, Password of WIFI
  //
  while(!device.connect(&espClient, http))  // Connect to AllThingsTalk
    Serial.println("retrying");

  // Create device assets
  device.addAsset("", "", "counting up", "sensor", "{\"type\": \"string\"}");//give asset name
  device.addAsset("", "", "counting up", "sensor", "{\"type\": \"string\"}");//give asset name

  while(!device.subscribe(pubSub))  // Subscribe to mqtt
    Serial.println("retrying");
}

void setupWiFi(const char* ssid, const char* password)
{
  delay(10);
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED)
  {
    delay(500);
    Serial.print(".");
  }
  Serial.println();
```

```
  Serial.println("WiFi connected");
}
char j;
int counter = 0;
void loop()
{
  if(Serial.available()>0)
  {
    while(Serial.available()==0);
    j=Serial.read();
    check();
    }
  device.process();
}
void check()
{
  if(j==")//give number and char that is passed in Arduino sensor code
  device.send("","");//print statement with asset name
  else if(j==") //give number and char that is passed in Arduino sensor code
  device.send("","");//print statement with asset name
  else if(j==") ) //give number and char that is passed in Arduino sensor code
  device.send("","");//print statement with asset name

  if(j==") )//give number and char that is passed in Arduino sensor code

  device.send("","");//print statement with asset name
  else if(j==") //give number and char that is passed in Arduino sensor code
  device.send("","");//print statement with asset name
}
```

```
/****
 * Callback function
 * Handle messages that were sent from the AllThingsTalk cloud to this device
 */
void callback(char* topic, byte* payload, unsigned int length)
{ }
```

6. IMPLEMENTATION

6.1 Introduction

Implementation is the stage of the project when the theoretical design is turned out into a working system. Thus it can be considered to be the most critical stage in achieving a successful new system and in giving the user, confidence that the new system will work and be effective.

The implementation stage involves careful planning, investigation of the existing system and it's constraints on implementation, designing of methods to achieve changeover and evaluation of changeover methods.

6.2 Project Implementation

6.2.1 Project modules

- Setup the arduino board.
- Upload code to arduino and NodeMCU.
- Create AllThingsTalk account.
- Create device and asset in AllThingsTalk.
- Create Pin boards in AllThingsTalk.

6.2.2 Module Description

1. Setup the arduino board:

In this setup we connect ultrasonic sensor and water flow sensor with arduino board. As water flow sensor is analog sensor it connected to the analog pins on the arduino board. And ultrasonic sensor is digital sensor it is connected to the digital pins on the arduino board. And we need to connect the connectivity IOT WIFI module we connected the Node MCU ESP8266 to the arduino.

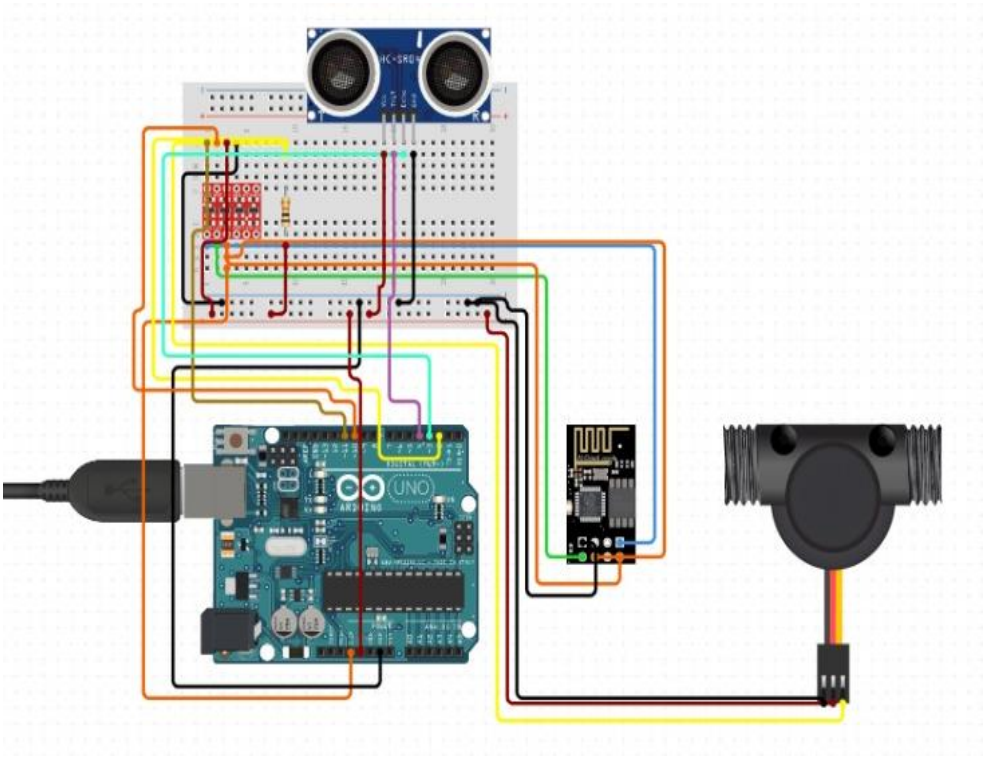


Fig 5.2.2.1 Circuit diagram of the setup

2. Upload code to arduino and NodeMCU:

The code written in the arduino software is to be uploaded in the respective boards in order to make them work. The code can be uploaded by using data cables. The Node MCU code should be uploaded separately and arduino code should be uploaded separately.

After uploading code into their respective boards, they will be ready to work according our code and give outputs.

3. Create AllThingsTalk account:

Creating an AllThingsTalk account is easy process which required a mail Id and we need to create a password. And Using this credentials the user will login into the AllThingsTalk account. The Admin need to create the ground which will hold devices used in that project. The admin who created the ground only have the access to share the ground, add devices, assets and add the members.

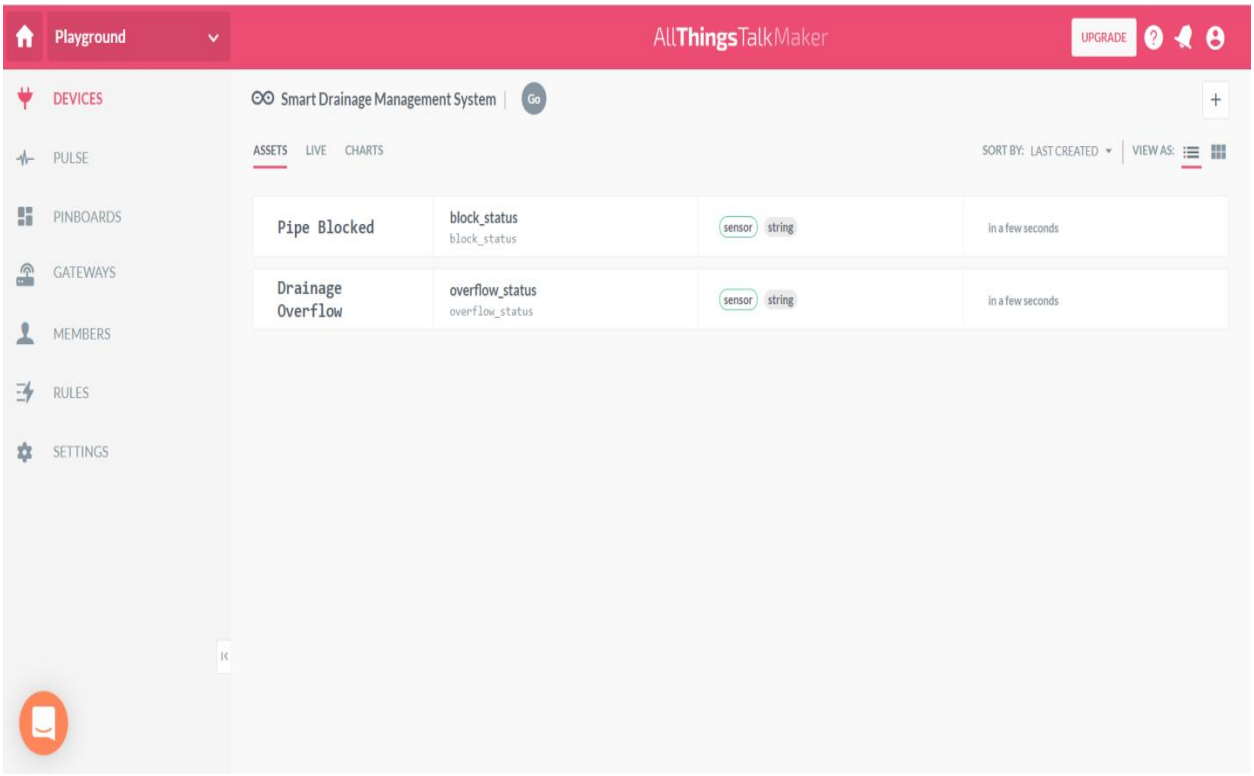
4. Create device and asset in AllThingsTalk:

After creating ground the admin will create a device that will represents the arduino board here. Then a maker id and device Id will created their in that account that Id's will be used in the coding part. After creating the devices, the assets should be created then the AllThingsTalk platform will be ready to display the outputs given by the assets(sensors).

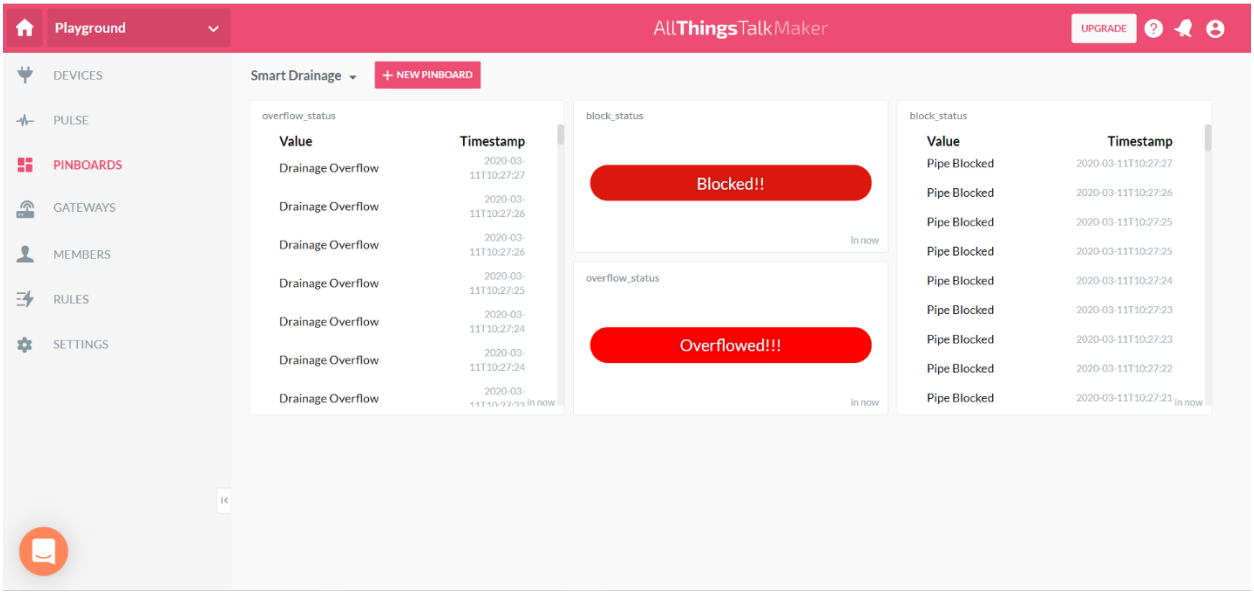
5. Create Pin boards in AllThingsTalk:

Pin boards are for visualizing part of the outputs. By creating Pin boards graphically represents the assets state or data. There are different controls available like bar charts or Line progress, charts, labels etc.

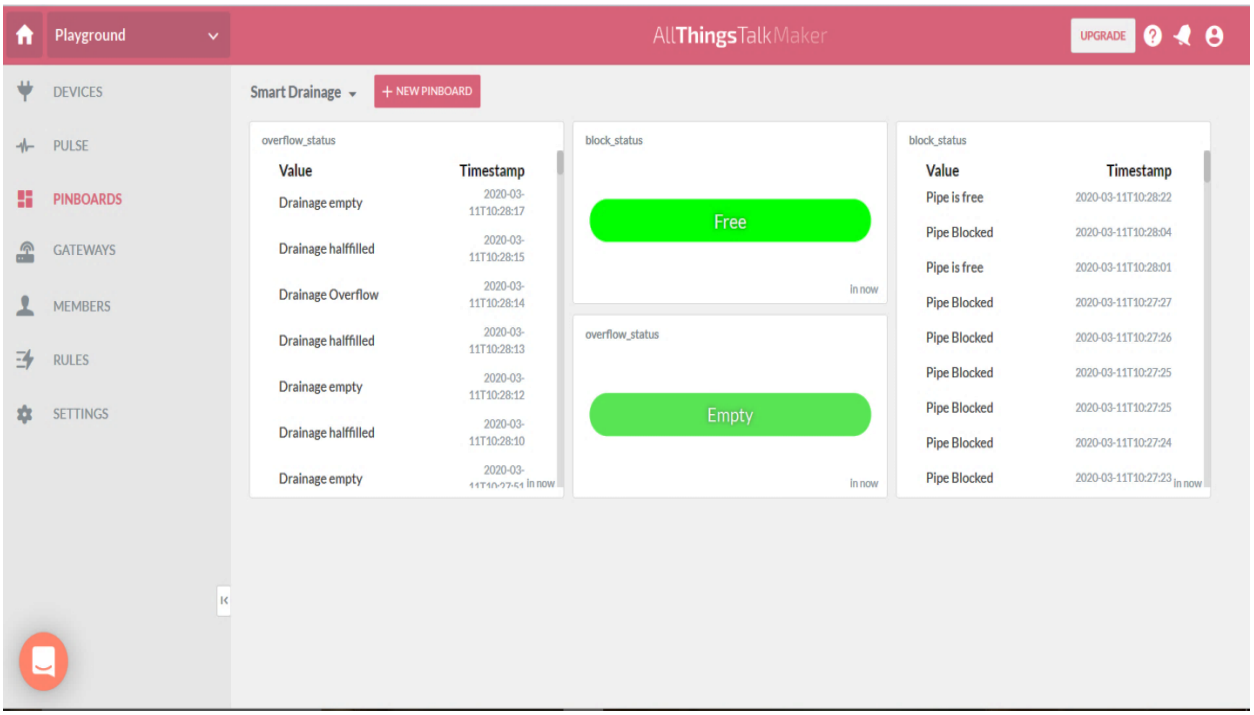
5.3 Output Screenshots



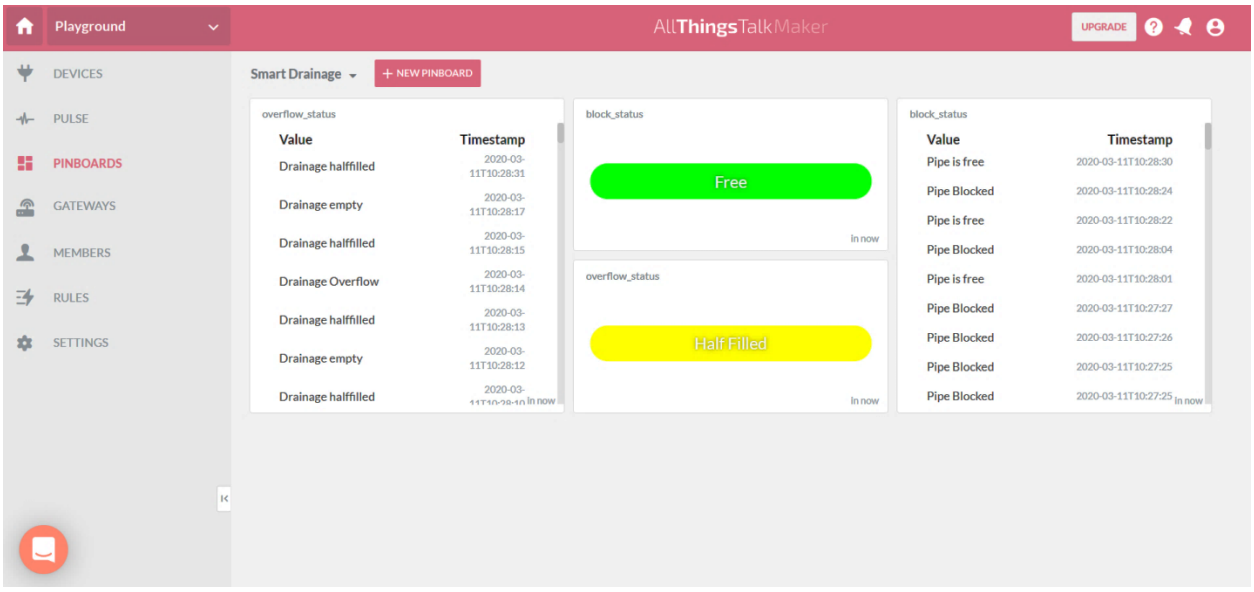
Screen #1 Output screen when pipe is blocked and drainage is overflowed



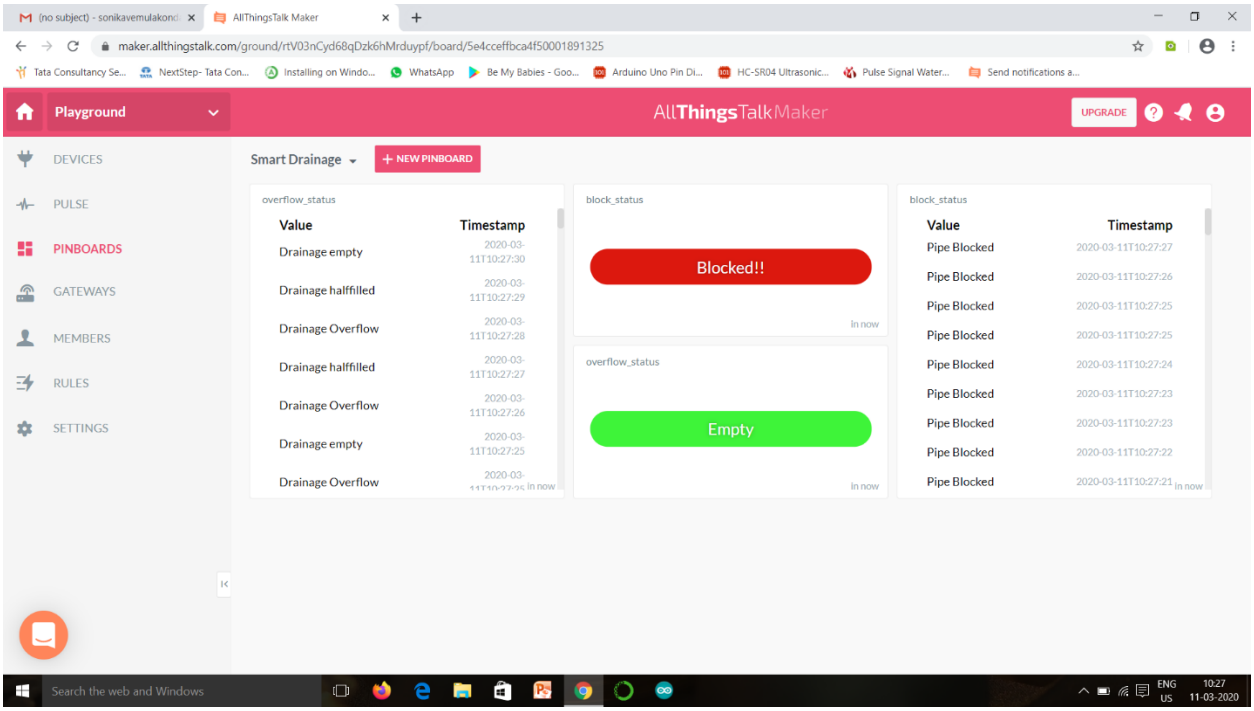
Screen #2 Output screen on pinboards when pipe is blocked and drainage is overflowed



Screen #3 Output screen on pinboards when pipe is free and drainage is empty



Screen #4 Output screen on pinboards when pipe is free and drainage is half filled



Screen #5 Output screen on pinboards when pipe is blocked and drainage is empty

7. Testing

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

IoT testing is a type of testing to check IOT devices. Today there is increasing need to deliver better and faster services. There is a huge demand to access, create, use and share data from any device. The thrust is to provide greater insight and control, over various interconnected IOT devices. Hence, IoT testing framework is important.

7.1 Types of Testing in IoT:

Usability Testing

- Compatibility Testing
- Reliability and Scalability Testing
- Data Integrity Testing
- Security testing
- Performance Testing

Usability Testing:

We need to make sure the usability of each of the device used here. Usability in terms of displaying data, processing data, pushing job tasks from the devices should be tested thoroughly.

There are so many devices of different shape and form factors are used by the users. Moreover, the perception also varies from one user to other. That's why checking usability of the system is very important in IoT testing.

Compatibility Testing:

There are lots of devices which can be connected though IOT system. These devices have varied software and hardware configuration. Therefore, the possible combination are huge. As a result, checking the compatibility in IOT system is important.

Testing items such as, multiple operating system versions, browser types and respective versions, generations of devices, communication modes [For e.g. Bluetooth 2.0, 3.0] is necessary for IoT compatibility testing.

Reliability and Scalability Testing:

Reliability and Scalability is important for building an IOT test environment which involves simulation of sensors by utilizing virtualization tools and technologies.

Data Integrity Testing:

It's important to check the Data integrity in IOT testing as it involves large amount of data and its application.

Security testing:

In the IOT environment, there are many users are accessing a massive amount of data. Thus, it is important to validate user via authentication, have data privacy controls as part of security testing.

IoT Security challenges: IoT is data centric where all the devices/system connected operate based on the data that is available. When it comes to the data flow between devices, there is always a chance that the data can be accessed or read when getting transferred. From a testing standpoint, we need to check if the data is protected/encrypted when getting transferred from one device to the other.

Wherever, there is an UI, we need to make sure there is a password protection on it.

Performance Testing:

Performance testing is important to create strategic approach for developing and implementing an IOT testing plan.

As testers, we need to make sure the system performs the same even though the added data is propagated. We should also test the monitoring utility to display the system usage, power usage, temperature etc.

7.2 Test Cases

Test Cases for Pipe Blockage

Test case description	Test case notation	Requirements	Expected Output (Message on the web browser monitoring page)	Actual Output (Message on the web browser monitoring page)	Test case status
Sends an alert message and displays on the web browser monitoring page as pipe found to be 'FREE'	T_1	Water should be flowing freely in pipe without any other materials in pipe so water flows without any blockage.	Pipe is free	Pipe is free	Pass
Sends an alert message and displays on the web browser monitoring page as pipe found to be 'BLOCKED'	T_2	Water flow should be decreased or stopped due to other waste materials blocking in the pipe.	Pipe Blocked	Pipe Blocked	Pass

Test Cases for Drainage Manhole filling

Test case description	Test case notation	Requirements	Expected Output (Message on the web browser monitoring page)	Actual Output (Message on the web browser monitoring page)	Test case status
Sends an alert message and displays on the web browser monitoring page as bin is found to be 'EMPTY'	T_3	There should be No or less then few water present in the bin.	Drainage empty	Drainage empty	Pass
Sends an alert message and displays on the web browser monitoring page as bin is found to be 'HALF FILLED'	T_4	Water should be half-filled in the bin.	Drainage half filled	Drainage half filled	Pass
Sends an alert message and displays on the web browser monitoring page as bin is found to be 'HALF FILLED'	T_5	Water should be Filled more than half in the bin.	Drainage Overflow	Drainage Overflow	Pass

8. CONCLUSION

Sensor networks are considered as the key enablers for the IoT paradigm [. This paper addresses all about smart and real-time Drainage monitoring system through IoT applications for metropolitan cities. By using various sensors such as gas detection, water level as well as blockage detection we can monitor the real time scenario of drainage system by for detecting the problems in drainage system. By doing this we can able to take particular action on the problems as we will receive the early alerts of blockage as well as increase. This paper can be used to design the smart and real time drainage system for monitoring as well as troubleshooting purpose.

REFERENCES

[1] Brown, Eric (13 September 2016).”Who need the Internet of things”.Linux.com. Retrieved 23 October 2016.

[2] Wemer-Allen, G., Johnson, J., Ruize, M., Less, J., and Welsh, Matt “Monitoring Volcanic Eruptions with a Wireless sensor Network. (ISSN: 2321 – 5658) Volume 01– Issue 04, December 2013 Asian Online Journals

[3] Basha, D. and Rus, D. “Design of Early Warning Flood Detection System for developing countries. Proceeding of the conference on ICTD, Bonsalove, India. Pp 1- 10, 2007.

[4] Yuwat, C. and Kilaso, S. “ A Wireless Sensor Network for Weather and Disaster Alarm System” , IPCSIT Vol. 6, Singapore. Pp 1 – 5, 2011

[5] Morias, R., Valente, A., Serodo, C. “A Wireless Sensor Network for Smart Irrigation and Environmental Monitoring. EFTA/WCCA Joint Congress on IT in Agriculture, Portugal, pp 845 – 850. 2005

[6] Windarto, J.” Flood Early Warning System develop at Garang River Semarang using Information Technology base on SMS and Web”. International Journal of Geomatics and Geosciences Vol. 1 No. 1, 2010

[7] Wirawam, S., Pratoma, I., and Mita, Nagahisa. “Design of Low Cost Wireless Sensor Network-Based Environmental Monitoring System for Developing Country”. Proceedings of APCC 2008.

[8] Retno Tri Wahyuni1* Yusmar Palapa Wijaya2 Dini Nurmalasari “Design of Wireless Sensor Network for Drainage Monitoring System” Vol.5, No.5, 2014

International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 03 | Mar 2020 www.irjet.net p-ISSN: 2395-0072

SMART DRAINAGE MANAGEMENT SYSTEM

G. Durvasi[1], G. Keerthi[2], V. Phani Sonika[3], T. Hepsiba Susan[4]

[1]Assistant Professor, Department of Information Technology and Engineering, Andhra Loyola Institute of Engineering and Technology, Vijayawada-08, Andhra Pradesh, India
[2,3,4]Under Graduate Student, Department of Information Technology and Engineering, Andhra Loyola Institute of Engineering and Technology, Vijayawada-08, Andhra Pradesh, India

---

Abstract - *As we are forming urban communities into brilliant urban communities subsequently there is a need for designing proper underground infrastructure which includes underground water pipelines, drainage monitoring for the purpose of keeping the city clean. If the system isn't monitored properly it's going to cause blockage and sewage overflow which successively results in contamination of pure water leading to spreading of infectious diseases. So different quite work has been done to detect, maintain and manage these underground systems. In like manner, leaks and bursts which are unavoidable aspects of water distribution system management and may account for significant water loss within a distribution network if left undetected for long period. This project represents the implementation for monitoring and managing underground system with different approaches. It gives a description of water flow system by giving water flow rate and detecting overflow and also it uses detection method to detect blockage defects in sewer pipeline based on Water flow.*

Key Words: IOT, WSN, Drainage System, Sensors.

1. INTRODUCTION

Sewage framework assumes a significant job in huge urban communities where a huge number of individuals live. Sewage framework is known as the base for land dryness from the overabundance and unused water. Downpour water and wastewater. Sewage conditions ought to be observed so as to keep up its appropriate capacity. Actually, not all zones have a sewage observing group. It prompts inconstant observing of the waste condition. The inconstant checking has added to the obstructing of the waste that suggests the welcome which triggers flooding in the area. Manual checking is additionally hard. It needs a ton of committed people who are just ready to record constrained reports with low exactness. The issue emerges in such waste lines can make difficult issues in the everyday schedule of the city. Issues, for example, blockage because of waste material, abrupt increment in the water level if the best possible cleaning moves are not made every once in a while. The present sewage framework isn't electronic because of which it is difficult to know whether blockage is happening specifically area. Likewise, we don't get early alarms of the blockage or the expansion in water level. Henceforth identification and fixing of the blockage becomes tedious and furious. NodeMCU combines features of WIFI access point and station + microcontroller. These are the features which makes the NodeMCU extremely powerful tool for Wi-Fi networking. It tends to be used as access point and/or station, host a webserver or connect to internet to fetch or upload data. By using this type of wireless sensor network, we can be notified the sensor data through internet.

1.1 PROBLEM STATEMENT

Overflow of sewage on roads is been a major problem in many developed and under developed cities as well. Existing drainage system is manual monitoring and all areas doesn't have proper monitoring teams. Manual monitoring is difficult and ineffective.

One of the major problems on the rainy days are overflow of manholes and drainage on a road. which is caused due to no information on manhole level of filling. Another problem in drainage system is difficulty of finding blockage in underground drainage pipes. Most of time drainage management team of municipality detects the problem in the drainage system when it is causes the trouble.

1.2 OBJECTIVE

This project is to keep the city clean, safety and healthy. And to replace the manual work of drainage monitoring for the safety of sewage workers.

The intelligence of sensors and predictive system identifies the drain is clogged and also give us information of level of manhole filled so, action can be taken by the authorities.

The sensors are communicated through Wi-Fi communication modules to share information.
This information can be viewed by authorized users at any time and any place in the internet.

2. LITERATURE SURVEY

1. Towards the implementation of IoT for environmental condition monitoring in homes

Authors: Sean Dieter Tebje Kelly, Nagender Kumar Suryadevara, and Subhas Chandra Mukhopadhyay,

Fellow, IEEE.

- In paper discusses an effective implementation for Internet of Things used for monitoring regular domestic conditions by means of low-cost global sensing system.
- The systems basic operations include remote management and control of domestic devices such as electric lamp; water heater etc.
- The framework of the monitoring system is based on a combination of widespread distributed sensing units, information system for the data collection, and reasoning and situation awareness.

2. On real-time performance evaluation of volcano-monitoring systems with wireless sensor networks

Authors: Roman Lara, Member, Diego Benitez, Senior Member, IEEE, Antonio Caamano, Marco Zennaro and Jose Luis Rojo- Alvarez

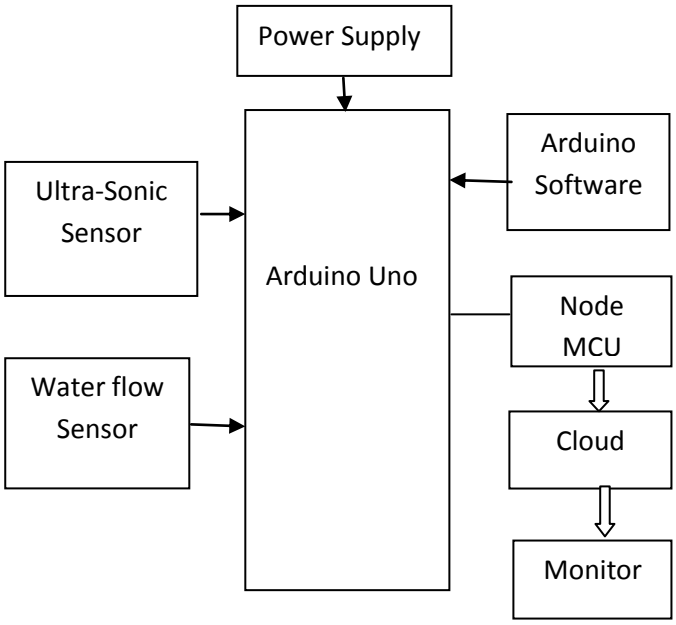
- This journal paper discusses about early warning of a volcanic eruption using real-time (RT) systems.
- In this wireless sensor networks (WSNs) may play a key role.
- To test the system in a real scenario, 10 sensors were deployed in a strategic area at Cotopaxi Volcano, and information was collected during three days of continuous monitoring.
- This information was sent to a remote surveillance laboratory located 45 km away from the station.

3. METHODOLOGY

The smart drainage system will have:

- Sensors to detect blockage, flood.
- The knowledge of sensors and framework will recognize the stopping up inside the seepage framework and will give the subtleties of the area and other data for additional activities.
- The Node MCU Wi-Fi module will be Wi-Fi organizing (can be utilized as passageway or potentially station, have a web server), interface with web to bring or transfer information.
- This whole information will be altogether sent by the Wi-Fi module hub to the server will be put away at the cloud every one of this information will be available progressively situation for consistent observing.

3.1 BLOCK DIAGRAM



4. SYSTEM SPECIFICATIONS

4.1 Arduino UNO

Arduino is a standard term for a software company, project, and user community that designs and produces computer open-source hardware, open-source software, and microcontroller-based kits for producing digital devices and interactive objects that can sense and monitor the physical devices.

The project is based on microcontroller board designs, produced by many vendors, using various microcontrollers. These systems provide sets of digital and analog I/O pins that can interconnect to various expansion boards (termed shields) and other systems. The board's features are serial communication interfaces, including universal serial bus (USB) on some designs, for loading programs from personal computers**.**

4.2 ULTRA AONIC SENSOR

The HC-SR04 Ultrasonic separation sensor comprises of two ultrasonic transducers. The one goes about as a transmitter which changes over electrical sign into 40 KHz ultrasonic sound heartbeats. The recipient tunes in for the transmitted heartbeats. On the off chance that it gets them it delivers a yield beat whose width can be utilized to decide the separation the beat voyaged.

4.3 WATER FLOW SENSOR

Water stream sensor comprises of a plastic valve from which water can pass. A water rotor along with a hall effect sensor is available to sense and measures the water stream.

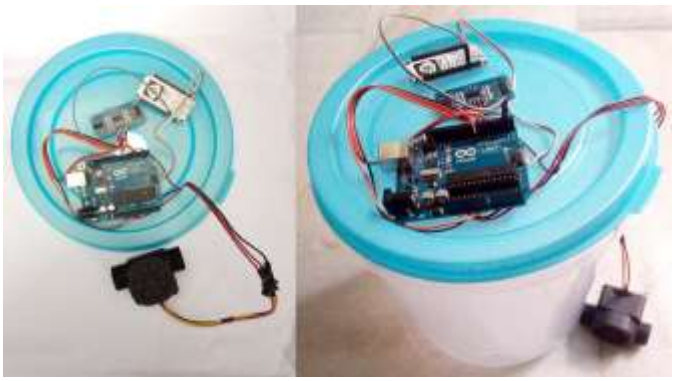
At the point when water moves through the valve it pivots the rotor. By this, the change can be seen in the speed of the engine. This change is determined as yield as a heartbeat signal by the corridor impact sensor. Consequently, the pace of stream of water can be estimated.

4.4 Node MCU

The All new NodeMCU ESP8266 V3 Lau CH340 Wi-Fi Dev. Board is a fast-leading edge low-cost Wife technology. Modern high-level mature LUA based technology. It is an integrated unit with all available resources on board. It is super simple to complement your existing Arduino projects or any development board that has I/O pins available.

Modern Internet development tools such as Node.js can take advantage the NodeMCU with the built-in API to put your idea on the fast track immediately. NodeMCU is built based on the mature ESP8266 technology to take advantage of the abundant resources available on the web.

5. Experiment Setup



Experimental Result

CONCLUSION

Sensor networks are considered as the key enablers for the IOT paradigm. This paper tends to about savvy and ongoing Drainage observing framework through IOT applications for metropolitan urban areas. By utilizing different sensors, for example, gas identification, water level just as blockage recognition we can screen the constant situation of waste framework by for recognizing the issues in sewage framework. By doing this we can able to take particular action on the problems as we will receive the early alerts of blockage as well as increase. This paper can be utilized to plan the savvy and constant sewage framework for checking just as investigating reason.

REFERENCES

[1] Brown, Eric (13 September 2016).” Who need the Internet of things”? Linux.com? Retrieved 23 October 2016.
[2] Wemer-Allen, G., Johnson, J., Ruize, M., Less, J., and Welsh, Matt “Monitoring Volcanic Eruptions with a Wireless sensor Network. (ISSN: 2321 – 5658) Volume 01– Issue 04, December 2013 Asian Online Journals
[3] Basha, D. and Rus, D. “Design of Early Warning Flood Detection System for developing countries. Proceeding of the conference on ICTD, Bonsalove, India. Pp 1- 10, 2007.
[4] Yuwat, C. and Kilaso, S. “A Wireless Sensor Network for Weather and Disaster Alarm System”, IPCSIT Vol. 6, Singapore. Pp 1 – 5, 2011
[5] Morias, R., Valente, A., Serodo, C. “A Wireless Sensor Network for Smart Irrigation and Environmental Monitoring. EFTA/WCCA Joint Congress on IT in Agriculture, Portugal, pp 845 – 850. 2005
[6] Windarto, J.” Flood Early Warning System develop at Garang River Semarang using Information Technology base on SMS and Web”. International Journal of Geomatics and Geosciences Vol. 1 No. 1, 2010