**WATER PIPELINE LEAKAGE DETECTION USING IOT**

**1. INTRODUCTION** :

Conservation of water is one of the major objectives for any country around the world. Water management plays a very important role in a society , as it is one of the basic needs for the mankind. Water supplied for human consumption must be highly efficient with minimum wastage. In a developing country like India, loss of water in domestic sector on account of leakage is approximately 30 to 40% of the total flow in the distribution. This leads to high risks in public health, money invested and on the valuable natural resource. India had an irrigation efficiency of ~36 percent in 1993-1994 and projected that efficiency would have to increase to 60 percent by 2050 to bring a balance in the demand and supply of water.

**a. Over view:**

Water distribution is generally installed through underground pipes. Monitoring the underground water pipelines is more difficult than monitoring the water pipelines located on the ground in open space. This situation will cause a permanent loss if there is a disturbance in the pipeline such as leakage. Leaks in pipes can be caused by several factors, such as the pipe's age, improper installation, and natural disasters. Therefore, a solution is required to detect and to determine the location of the damage when there is a leak. The detection of the leak location will use fluid mechanics and kinematics physics based on harness water flow rate data obtained using flow liquid meter sensor.

**b. Purpose :**

A water detector is an [electronic](https://en.wikipedia.org/wiki/Electronics) [device](https://en.wikipedia.org/wiki/Machine) that is designed to detect the presence of water for purposes such as to provide an alert in time to allow the prevention of [water leakage](https://en.wikipedia.org/wiki/Water_leakage). A common design is a small cable or device that lies flat on a floor and relies on the [electrical conductivity](https://en.wikipedia.org/wiki/Electrical_conductivity) of water to decrease the resistance across two contacts. The device then sounds an audible [alarm](https://en.wikipedia.org/wiki/Alarm) together with providing onward signalling in the presence of enough water to bridge the contacts. These are useful in a normally occupied area near any infrastructure that has the potential to leak water, such as [HVAC](https://en.wikipedia.org/wiki/HVAC), [water pipes](https://en.wikipedia.org/wiki/Water_pipes), [drain pipes](https://en.wikipedia.org/w/index.php?title=Drain_pipes&action=edit&redlink=1), [vending machines](https://en.wikipedia.org/wiki/Vending_machines), [dehumidifiers](https://en.wikipedia.org/wiki/Dehumidifier), or [water tanks](https://en.wikipedia.org/wiki/Water_tank).

# 2. LITERATURE SURVEY:

**a. Existing problem :**

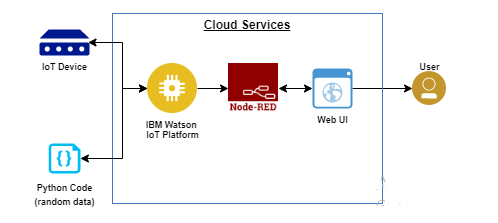
If your water pressure is excessively high, it could cause your pipes to leak and even burst. When the pressure in your pipes is distributed unevenly, it puts unnecessary force on the rest of your pipes and causes leaks. High water pressure can also cause drip leaks from faucets, showers, and toilets.

**b. Proposed solution:**

* Through this project, we can detect the pipeline leakage based on the flow rate of the water.
* The pipelines will be integrated with flow sensors for detecting flow rate and valves for controlling.
* There will be two sensor nodes connected to the main gateway.
* All the nodes flow rate will be sent to the main gateway, at the main gateway we will compare the flow rates.
* If the flow rate is less when compared to the flow rate of the main valve then the admins will be notified.
* When the admins get the notifications they can switch off the valves at the particular nodes through the web application.
* Every sensor node will also send the predefined location to recognize the location of the node.

# 3. THEORETICAL ANALYSIS:

**a. Block diagram:**



* Create Node-Red Application
* Create IBM Watson IoT Platform.
* Get the python code for running the project
* Configure the nodes and write a function to compare the flowrates
* if flow rate is less at any node that means there is a leakage
* if leakage is detected send a message notification

**b. Hardware/Software designing:**

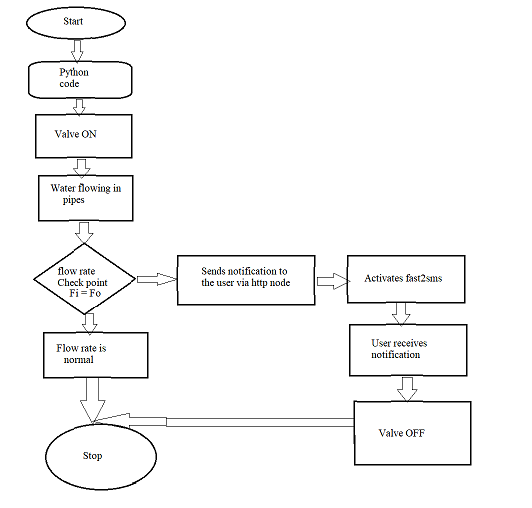
**SoftwareDesigned:**

* Installation of node red.
* Installation of IBM i/o nodes.
* Obtaining URLs for http nodes.
* Obtaining API Keys for their respective ibm nodes.
* Signing up fast2sms site.
* Obtaining API Keys for the sms communication.
* Storing the obtained data in the cloud

**4. EXPERIMENTAL INVESTIGATIONS:**

* For the experimental set up, initially we need to make the appropriate connects in the *Node Red app* provided with their service credentials and deploy the app.
* It Involves a designed test in which variables are changed, controlled, and measured in an effort to gather evidence to support or disprove a hypothesis.
* Give the proper connections with wires for connecting the nodes. The property and condition of each and every node bust be satisfied in order to get the proper output.
* Input node that can be used with Watson IoT Platform to receive events sent from devices, receive commands sent to devices, or receive status updates concerning devices or applications. It produces an object called msg and sets msg.payload to be a String containing the payload of the incoming message.
* Connect the function node for the representation of first flowrate sensor i.e, inlet sensor and also connect another function node for the outlet with the ibm iot node as an input.A JavaScript function to run against the messages being received by the node.
* Connect the guage nodes in order to measure and dispaly of flow rate of the pipes. Adds a gauge type widget to the user interface.
* Then connect rbe node. Report by Exception (RBE) node - only passes on data if the payload has changed. It can also block unless, or ignore if the value changes by a specified amount (Dead- and Narrowband mode)
* Then connect the switch node to the rbe node for the condition placement. It routes
* messages based on their property values or sequence position.
* Then connect the http response node to the switch node . It sends HTTP requests and returns the response.
* When it comes to debug node . It displays selected message properties in the debug sidebar tab and optionally the runtime log. By default it displays msg.payload, but can be configured to display any property, the full message or the result of a JSONata expression.
* At last when it comes to the cloudant node,It is a simple Cloudant output node. It stores msg in a chosen database.
* Ensure that the API keys and the URL's are properly located in their respective block of nodes. If the service credentials are misplaced from their respective nodes we may encounter a few errors and if the assistant credentials are given mistake, then it displays missing property. In this manner we can find theflawless nodes and connect them.

**5 . FLOW CHART**



**6.RESULT:**

* Using this flow , the water, when it gets leaked, can be determinesd by the sensor function node
* The Input flow rate is compared with the output flow rate to determine the water leakage to notify the user to avoid further ploblems.
* If the flow rates of both input and output sensors are equal, then the pipe is not leaking
* If the flow ratesof both sensors are not equal, then the nodes will activate and with the help of sms , the leakage message will be sent to the user.
* Then the valve can be turned off.

# **7**.ADVANTAGES & DISADVANTAGES:

**Advantages:**

* In order to prevent serious damage being done through faulty pipes
* It is essential to install a water leak system that will detect a problem before disaster hits.
* Water detection systems will alert you to even the smallest of leaks so that you can resolve them quickly.
* Money Saver
* Power saver

**Disadvantages:**

* The rust, foul and deteriorate
* Electronics are usually built separately
* More difficult installation
* No Warranty or Guarantee

**8. APPLICATIONS**

* Hospitals
* Hotels
* Near Water tanks
* Residential areas
* Under Ground pipelines
* Complex areas
* Stations

**9. CONCLUSION**

A leak detection system should be installed at the earliest possible point of entry in your household plumbing. All water-using appliances and water heaters should be downstream from the leak detector. The less water travels before reaching the detection system, the greater the likelihood of avoiding water damage. All water filtration systems, like a water softener or other whole-house water filtration system, should occur after the leak detection system. If the water filtration system becomes clogged or springs a leak, the monitor will spot the change in water flow and protect both your home and further damage to your equipment. Similarly, any water-using appliance, from your refrigerator to your showers, should occur after the leak detector to ensure maximum efficiency from the unit. Water heater tanks can burst if the volume of water within the tank causes the seams to burst or a malfunctioning thermostat causes the water to overheat.

Some leak detectors are point-of-use leak detectors. These are usually installed under a sink, targetting an under-sink water filtration system or a potentially leaky sink drain. These leak detectors usually use sensors to discern when water is leaking onto the cabinet’s floorboards. Some of these point-of-use detectors set off a loud alarm to alert the homeowner, while others have the ability to shut off the water supply to the specified faucet or water filter system.

**10. FUTURE SCOPE**

The early detection of leaks can prevent major gas spills, water seeping into the soil under highways resulting in sinkholes, minimizing infrastructure damage, preventing damages to the surrounding environment or personnel, and save money.Pipelines and thus also leak detection systems can be

found in a wide variety of areas for various products.Accordingly, the challenges that the leak detection system faces vary depending on the application. At the end of the day, the question is which leak detection system is the right one? There is no one right leak detection system.Modern leak detection systems function in a wide variety of environments and allow for individual adaptation to customer surroundings, guaranteeing optimal performance under all normal operating conditions.

**11. BIBILOGRAPHY**

* https://internetofthings.ibmcloud.com/
* www.fast2sms.com
* https://node-red-hggtz-2021-05-15.eu-gb.mybluemix.net/
* docs.fast2sms.com
* https://cloud.ibm.com/
* https://github.com/

**12. APPENDIX**

1. **Source code:**

import time

import sys

import ibmiotf.application

import ibmiotf.device

import random

import json

#Provide your IBM Watson Device Credentials

organization = "hm0hyb"

deviceType = "iotdevice"

deviceId = "1001"

authMethod = "token"

authToken = "1234567890"

# Initialize the device client.

I=0

O=0

def myCommandCallback(cmd):

print("Command received: %s" % cmd.data['command'])

if cmd.data['command']=='valveon':

print("VALVE ON IS RECEIVED")

elif cmd.data['command']=='valveoff':

print("VALVE OFF IS RECEIVED")

if cmd.command == "setInterval":

if 'interval' not in cmd.data:

print("Error - command is missing required information: 'interval'")

else:

interval = cmd.data['interval']

elif cmd.command == "print":

if 'message' not in cmd.data:

print("Error - command is missing required information: 'message'")

else:

print(cmd.data['message'])

try:

deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": authMethod, "auth-token": authToken}

deviceCli = ibmiotf.device.Client(deviceOptions)

#..............................................

except Exception as e:

print("Caught exception connecting device: %s" % str(e))

sys.exit()

# Connect and send a datapoint "hello" with value "world" into the cloud as an event of type "greeting" 10 times

deviceCli.connect()

while True:

I=88

O=22

#Send Temperature & Humidity to IBM Watson

data = {"d":{ 'flowratei' : I, 'flowrateo' : O,}}

print (data)

def myOnPublishCallback():

print ("Published flowratei = %s C" % i, "flowrateo = %s %%" % o, "to IBM Watson")

success = deviceCli.publishEvent("Data", "json", data, qos=0, on\_publish=myOnPublishCallback)

if not success:

print("Not connected to IoTF")

time.sleep(1)

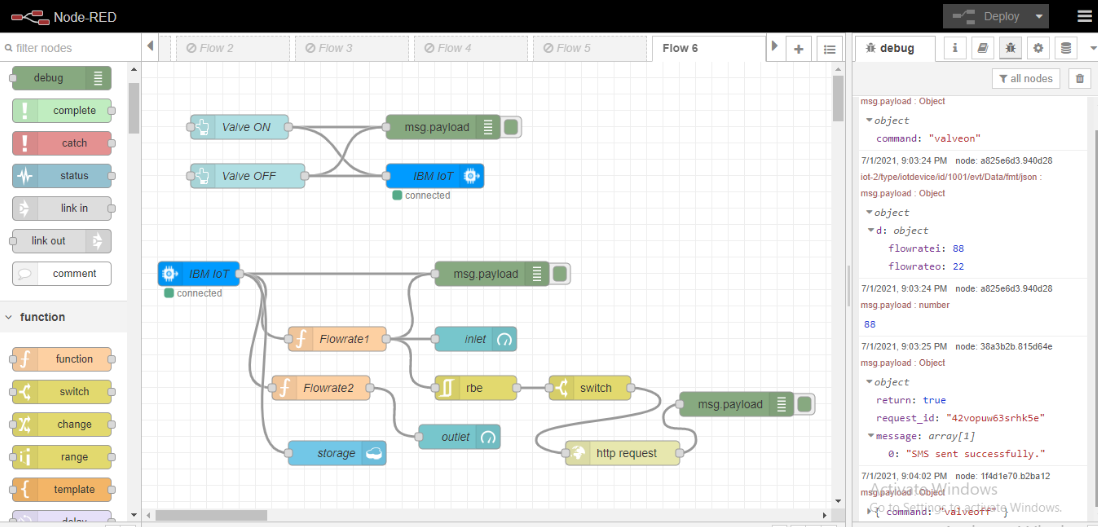
deviceCli.commandCallback = myCommandCallback

# Disconnect the device and application from the cloud

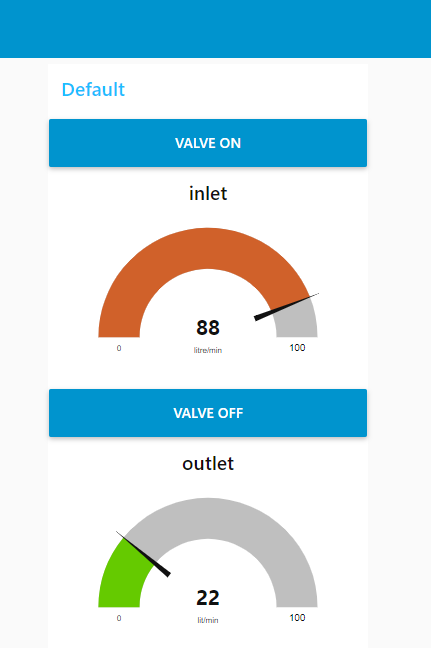
deviceCli.disconnect()

**b. UI output screenshots:**

Node red:



**UI :**



**Code output: USER'S ALERT MESSAGE**

