Aalto University
School of Science
and Technology

CS-E5360 Systems of Systems

Practical Session

Asad Javed
PhD Researcher
Computer Science

Avleen Malhi
Postdoctoral Researcher
Computer Science

Agenda

- Introduction
- O-MI/O-DF Reference Implementation
- How to run O-MI node
- Live demos
- (Scripts and Slides are available at: https://github.com/AaltoAsia/SoS-Scripts)

Introduction

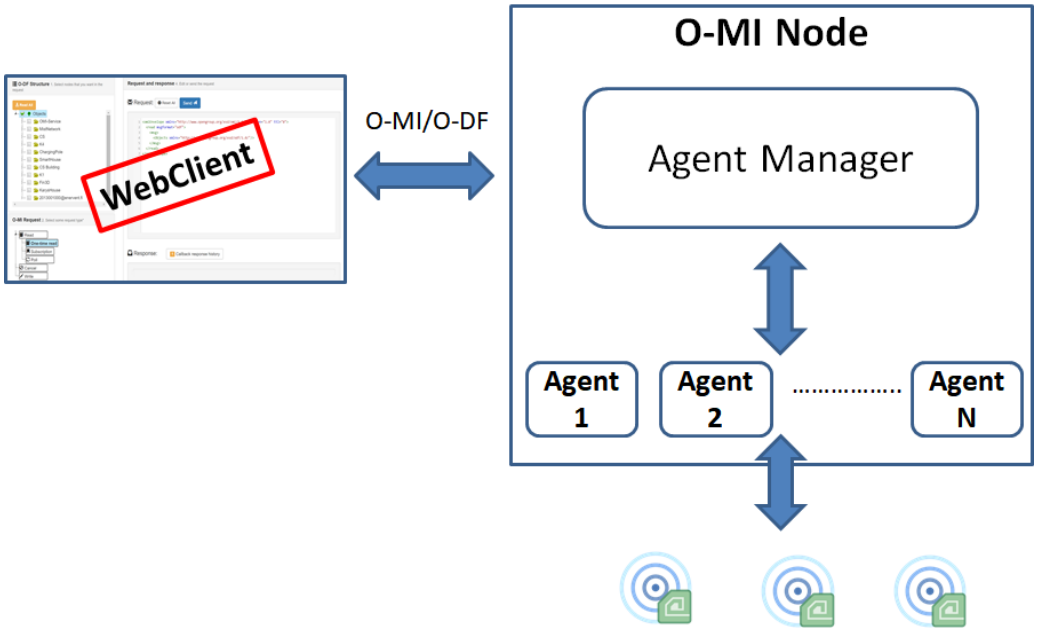
Aalto University
School of Science
and Technology

- This session will address the practical aspects of using IoT technology to deliver smart city solutions.
- Goal of the session:
  - Install O-MI Node reference implementation
  - Use an IoT device to send sensor values to your Node
  - Make a data subscription to send value from your Node to our Node
- We will have **live visualization** of all the data coming to our O-MI Node

O-MI/O-DF Reference Implementation

Aalto University
School of Science
and Technology

- Open source implementation developed at Aalto University
  - Available at: https://github.com/AaltoAsia/O-MI
- Enable fast deployment of IoT node

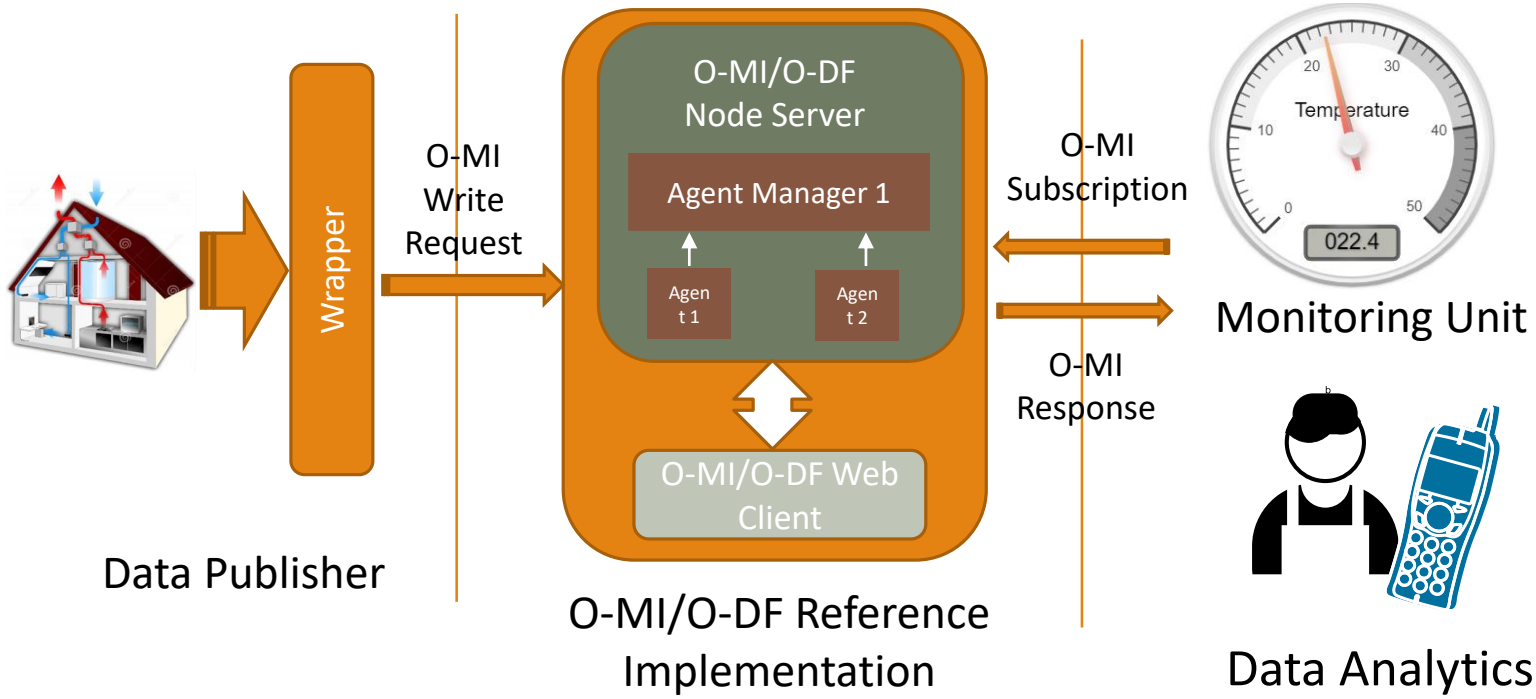


How to run O-MI node?

Aalto University
School of Science
and Technology

- Dependency Java JDK 1.8
- Run pre-compiled node:
  1. Download the latest release version and unpack it. Available at: https://github.com/AaltoAsia/O-MI/releases
     - For Windows: Use “o-mi-node-1.0.9.zip” (Don’t use deep directory structure)
     - For MAC/Linux: Use “o-mi-node-1.0.9.tgz”
  2. To allow **write** requests from other machines you need to modify file “<Extracted folder>/conf/application.conf”.
     - Change the setting “**allowRequestTypesForAll”**, add “**write**” to the list, such that the result looks like this:
     - allowRequestTypesForAll = ["read", "cancel", "call", "**write**"]
  3. **For Windows:** Go to the <Extracted folder>/bin and click on o-mi-node.bat
     For MAC/Linux: cd <Extracted folder>/bin and run “./ o-mi-node”
  4. The server can now be accessed with URL http://localhost:8080/

Case Study: Smart Home



Smart Home Scenario Description

Aalto University
School of Science
and Technology

- Smart Home equipped with
  - Temperature and Humidity sensor (SHT-20)
  - CO2 sensor (S-100)
- Wrapper to publish smart home data to O-MI node
  - Read sensor value
  - Translate it to O-DF
  - Put it in O-MI Write request
  - Send with HTTP POST request
- Developer can test the system using web-client
- Data consumer (Monitoring unit) might be interested in these sensor values for analyzing the required object values. It can subscribe to the objects of interest

Hardware and Sensors Provided

- ESP8266 WiFi development modules
- SHT20 Temperature and humidity sensor
- S-100 CO2 sensor

Live Demo

Implementation Using ESP8266

Download Scripts

Aalto University
School of Science
and Technology

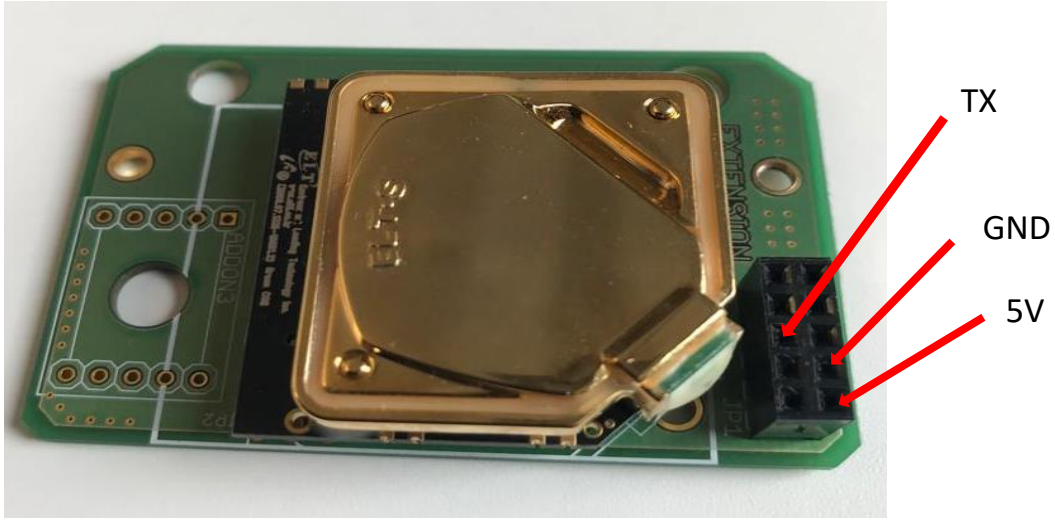
- **Connect to our WiFi “TP-LINK_C5E618”**
- Open new termial and download the provided scripts from https://github.com/AaltoAsia/SoS-Scripts

ESP8266 Arduino Configuration

Aalto University
School of Science
and Technology

- The ESP8266 is a very low-cost Wi-Fi microchip with full TCP/IP stack and microcontroller capability.
- Arduino is an open-source electronics platform based on easy-to-use hardware and software.
  - Install latest Arduino IDE from https://www.arduino.cc/en/main/software
  - Start Arduino and open Preferences window.
  - Enter https://arduino.esp8266.com/stable/package_esp8266com_index.json into Additional Board Manager URLs field.
  - Open Boards Manager from Tools > Board menu and install esp8266 platform
  - **For MAC machine**, download and install the driver ”CH34x_Install_V1.5.pkg” from https://github.com/adrianmihalko/ch340g-ch34g-ch34x-mac-os-x-driver
  - Restart the Arduino IDE

CO_2 Sensor (Model: S-100)



- Connect jumper cables:
  - Sensor TX -> ESP RX
  - Sensor GND -> ESP GND
  - Sensor 5V -> ESP 5V

Interfacing CO2 Sensor with ESP8266

Aalto University
School of Science
and Technology

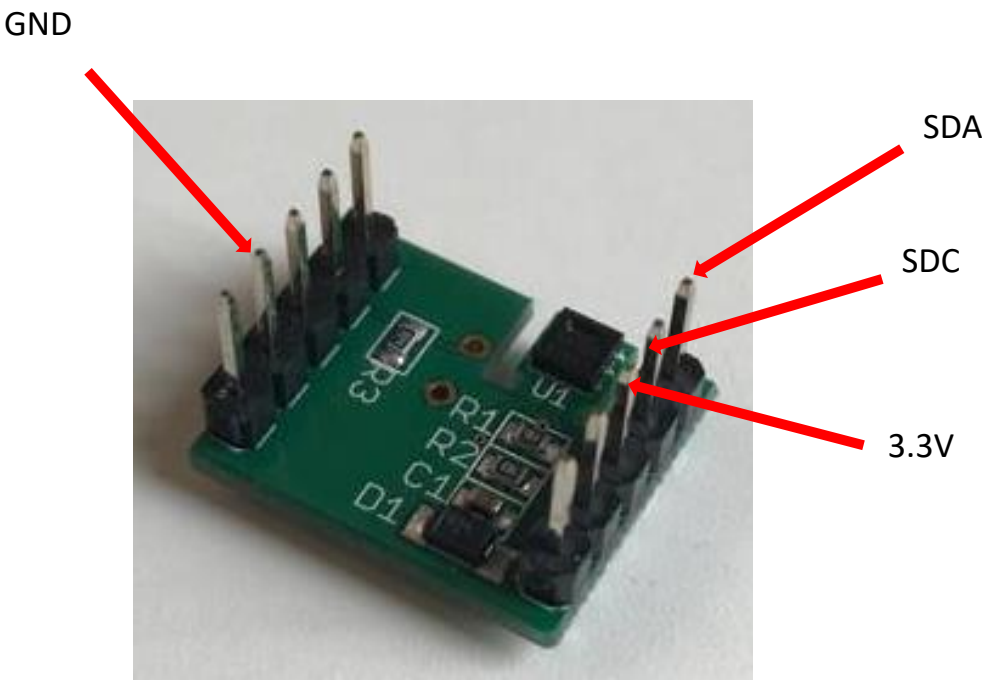
1. Connect ESP chip with your laptop USB port.
2. Start Arduino IDE and select the following board from Tools -> Board
   - LOLIN (WEMOS) D1 R2 and Mini
3. Select port using Tools -> Port
4. cd <....>/SoS-Scripts/CO2_test (for opening test scripts for CO2 sensor)
5. Open create_odf tab in Arduino and change <id> of the Object to something unique and change the url to your O-MI Node (http://yourIP:8080/)
6. **Note:** Disconnect the RX of ESP8266 while uploading the code.
7. Press Upload button 

8. Connect RX of ESP8266 again after uploading is done.
9. If it fails you might have wrong port or too high upload speed
10. View the output through Tools -> Serial Monitor
    - Select baud rate as 38400

Temperature and Humidity Sensor (Model: SHT-20)

Aalto University
School of Science
and Technology

# ?	* ?
+ GND	* ?
*?	* 3v3
*?	* SDC
*?	* SDA



- Connect jumper cables:
  - Sensor SDA -> ESP SDA (D2)
  - Sensor SDC -> ESP SDL (D1)
  - Sensor GND -> ESP GND
  - Sensor 3.3V -> ESP 3V3

SHT20 Sensor

Aalto University
School of Science
and Technology

- Download the library for SHT20 from https://github.com/DFRobot/DFRobot_SHT20 into the 'libraries' folder of your Arduino installation directory (<....>/Documents/Arduino/libraries).
- cd <....>/SoS-Scripts/temp_humi_test (go to our scripts directory)
- Open create_odf tab in Arduino and change <id> of the Object to something unique and change the url to your O-MI Node (http://yourIP:8080/)
- Press Upload button
- Tools -> Serial Monitor
  - Select baud rate as 115200

Create a subscription for our visualization Node

Aalto University
School of Science
and Technology

1. Go to http://localhost:8080/ and go to ”O-MI Test Client WebApp”
2. Check that your data have been received correctly with a Read request, after that, continue with subscription:
3. Select ”Objects” from the O-DF tree
4. Select ”Subscription” request
5. Open ”Optional parameters”
6. Put address of the visualization node (http://192.168.1.20:8080/) to ”callback” field
7. Press ”Send” and see if your value shows on the visualization (after it changes next time)


Thank you