Blockchain & Internet of Things Summer School 2016

IoT and Distributed Systems
Part I

Farzam Fanitabasi
PhD Candidate @ TU Delft
Faculty of TPM
Session Outline

- Internet of Things (IoT) and Distributed Systems
- Introduction to OpenIoT & GSN
  - Architecture
  - Scheduler
  - Service Delivery & Utility Manager
  - X-GSN
  - Mobile Sensor Management
    - CUPUS
    - QoS Manager
  - Optimization
  - Security
- Hands-on Configuration
  - GSN
  - IDE Core
  - Request Definition
  - Request Presentation
- Closing Notes
IoT and Distributed Systems

• Mutual Definitions:
  • Distributed / Hierarchical / Decentralized Systems
  • Internet of Things (IoT)

• Characteristics of an IoT infrastructure
  • Heterogeneous Devices
  • Resource Constraint
  • Spontaneous Interaction
  • Ultra Large-Scale Networks and Events
  • Dynamic Networks
  • Context Aware
  • Intelligence
  • Location-Aware
  • Distributed

Main Reference: [1]
IoT and Distributed Systems

• Mutual Definitions:
  • Hierarchical / Distributed / Decentralized Systems
  • Internet of Things (IoT)

• Characteristics of an IoT application
  • Diversity
  • Real Time
  • Everything as a Service
  • Increased Security Attack-Surface
  • Privacy Leakage

Main Reference: [1]
OpenIoT

• Why OpenIoT (according to OpenIoT team)
  • Features:
    • Complete (more or less) IoT Middleware
    • [One possible] blueprint for IoT solutions
    • Supports virtually any sensor type available
    • Blends IoT with semantic technologies
    • Support IoT/Cloud Integration and Convergence
    • Based on proven/available open source technologies
  • Awards:
    • Winner of “Rookie of the Year” as best Internet of Things Open Source Platform in 2013
    • Winner of “best semantic Interoperability” solution for sensor integration at the IoT Hackaton in 2014

Main Reference: [2-5]
OpenIoT Architecture

Main Reference: [2-5]
OpenIoT Utility-App Plane

• Utility-App Plane
  • Request Definition

  SERVICE SPECIFICATION

  REQUEST FORMULATION

  REQUEST SUBMISSION TO SCHEDULER

• Request Presentation
• Configuration and Management

Main Reference: [2-5]
OpenIoT Utility-App Plane

- Utility-App Plane
  - Request Definition
  - Request Presentation

- Configuration and Management

SELECT APPROPRIATE MASHUPS

RETRIEVE DATA FROM SDUM

REAL-TIME VISUALIZATION OF DATA

Main Reference: [2-5]
OpenIoT Utility-App Plane

• Utility-App Plane
  • Request Definition
  • Request Presentation
  • Configuration and Management

- MONITORING MODULE HEALTH
- CONFIG SENSOR FUNCTIONALITIES
- MANAGE DEPLOYED SERVICES

Main Reference: [2-5]
OpenIoT Virtualized Plane

- Virtualized Plane
  - Scheduler

  **PROCESS REQUESTS**

  **ENSURES PROPER ACCESS TO RESOURCES**

  **DISCOVER SENSOR AND ASSOCIATED DATA**

- Cloud Data Storage
- Service Delivery and Utility Management

Main Reference: [2-5]
OpenIoT Utility-App Plane

- Utility-App Plane
  - Scheduler
  - Cloud Data Storage

STORING DATA STREAMS AND METADATA

ADDED PUSH/PULL FUNCTIONALITY

Main Reference: [2-5]
OpenIoT Utility-App Plane

- Utility-App Plane
  - Scheduler
  - Cloud Data Storage
  - Service Delivery and Utility Management

- Combine Data Streams (Workflows)
- Service Metering Facility
- Performs Dual Role
- Using SPARQL from Scheduler

Main Reference: [2-5]
OpenIoT Physical Plane

- Physical Plane
  - Extension to Global Sensor Network (GSN)
  - X-GSN (used in OpenIoT)

Main Reference: [2-5],[6]
More on OpenIoT Scheduler

• Duties:
  • Sensor and ICO Detection
  • Virtual Sensor “Indirect” Activation
  • Request Storing and Activation
  • Service Status Update
  • Access Control

• Services:
  • Resource Discovery
  • Service User Management
    • Register / Unregister / Suspend / Enable / Enable from Suspension / Update
  • Registered Service Status
  • Service Update Resources
  • Get Service
  • Get Available Services
  • Get User

Main Reference: [2-5]
Indirect Dynamic Sensor Control

- Indirect Sensor Control
- Sensor Use Identification

Main Reference: [2-5]
More on OpenIoT Scheduler

Main Reference: [2-5]
More on OpenIoT Scheduler
Mobile Sensor Management

- CUPUS

```python
brokerName = testBroker
brokerPort = 10000
internalUDPPort = 10001

# max number of publications waiting to be sent, for each subscriber
queueCapacity = 1000

# GCM API key
apikey = AIzaSyADsn6N5Rq6an73KnSaEt3geI-BHy5nBdY

# Number of matcher processes, if elasticity is enabled this is the maximal number of Boolean Matchers
numberOfBooleanMatchers = 1
numberOfTopKWMatchers = 1

# Enable dynamic management of matchers
elasticity = False

# Maximal share of idle time that will trigger splitting event
splitThreshold = 0.1

# Minimal share of idle time that will trigger merging event
mergeThreshold = 0.8

# Number of messages after which elasticity check is performed
checkThreshold = 500

# If testing is set there will be extensive output to std.out
testing = True

# If logWriting is set a log file will be created and filled with detailed info of broker
logWriting = True
```

Main Reference: [2-5]
Security in OpenIoT

- Security Management Console
  - Defining/managing permissions for registered services
  - Defining/managing roles for registered services
  - Assigning roles to users or revoking roles from users
  - Adding/managing services
  - Registering/managing users

Main Reference: [2-5]
Hands-On Configuration of OpenIoT / GSN
Hands-on Configuration of OpenIoT / GSN

• Requirements:
  • Virtualization Software (preferably Oracle VMBox)
  • Java JRE/JDK version 6+ installed
  • GSN Installer (.jar file)
  • OpenIoT VMBox Image (latest release)
  • TextEdit software (such as Sublime Text)
  • JAVA IDE (such as Eclipse, IntelliJ, …)
Hands-on Configuration of OpenIoT / GSN

- **GSN**
  - **Sample Implementations**
    - [http://data.permasense.ch/index.html](http://data.permasense.ch/index.html)
    - [http://data.opensense.ethz.ch/](http://data.opensense.ethz.ch/)
    - [http://montblanc.slf.ch:22001/](http://montblanc.slf.ch:22001/)
  
- **Install GSN on workstation**
  - **Run GSN**
  - **In browser: localhost:22001**
  - **Map / Data / Output / Plot**
  - **Stop GSN**
  - **Go to Directory**

Main Reference: [6]
Hands-on Configuration of OpenIoT / GSN

• GSN
  1. Configuration
     1. GSN.xml
     2. realm.properties
     3. wrappers.properties
  2. Logs
     1. GSN.log
  3. Virtual Sensors
     1. Samples
     2. How to load / add / remove
     3. VS Structure

• Advance Topics
  • Security in GSN
  • Monitoring GSN performances
  • Continuous Queries in GSN
  • Configure GSN for performances
  • Configure GSN Storage
  • Configuring the SensorScope listener
  • Building and querying models in GSN
  • Client side processing

Main Reference: [6]
Hands-on Configuration of OpenIoT / GSN

• OpenIoT
  1. Open run the virtual machine / Build it yourself
  2. User/Pass: openiot/openiot
  3. Deploy the new code
  4. Start OpenIoT
     1. cd ~/Desktop/scripts
     2. ./virtuoso_start.sh
     3. ./jboss_start.sh
        1. if this throws errors / exceptions, something went wrong
     4. ./gsn_start.sh
  5. open on browser: localhost:22001
  6. stop server

Main Reference: [2-5]
Hands-on Configuration of OpenIoT / GSN
Hands-on Configuration of OpenIoT / GSN

- OpenIoT
  1. Jboss: OpenIoT.properties
     1. URLs
     2. Permissions
     3. Privacy / Security
     4. CAS Properties
     5. QoS Manager
  2. Jboss: security-config.ini
  3. OpenIoT code: Modules
     1. Security
        1. Security server: openiot.sql
     2. SDUM
     3. Scheduler
        1. rest-client.ini

Main Reference: [2-5]
Hands-on Configuration of OpenIoT / GSN

- OpenIoT
  - *OpenIoT code: Modules*
    1. X-GSN
       1. `gsn.xml`
       2. `data folder: weather stations`
       3. Virtual sensors
          1. `demo_weather.metadata`
          2. `demo_weather.xml`
          3. `samples`
          4. `LSM`
          5. create the data folder
    2. QoSManager
    3. LSM-light

Main Reference: [2-5]
Hands-on Configuration of OpenIoT / GSN

- OpenIoT
  - OpenIoT code: Modules
    1. CUPUS
      1. config
        1. mb1.config
        2. pub1.config
        3. sub1.config
        4. test_broker.config

Main Reference: [2-5]
Hands-on Configuration of OpenIoT / GSN

- OpenIoT
- Consoles:
  1. Security Console
     1. localhost:8080/security.management
     2. user/pass = admin/secret
     3. Manage roles
     4. manage users
     5. manage permissions
     6. manage services
     7. guest services

Main Reference: [2-5]
Hands-on Configuration of OpenIoT / GSN

- OpenIoT
  - Consoles:
    1. Request Definition
       1. localhost:8080/ui.requestDefinition
       2. user/pass = openiot/openiot
       3. load application (demo)
       4. add the filters / aggregators / presenters
       5. save the application
       6. validate your code
       7. add the data source

Main Reference: [2-5]
Hands-on Configuration of OpenIoT / GSN

• OpenIoT
  • Consoles:

1. Add sensors and data source
   A. `./lsm-register.sh <metadata_filename>`
   B. `lsm-register.bat <metadata_filename>`
   C. `curl -v http://localhost:22001/vs/vsensor/sens123/register -X POST --data-binary @virtual-sensors/LSM/opensense_1.metadata`

```
[main]
casOAuthClient=org.openiot.security.client.rest.CasOAuthWrapperClientRest
casOAuthClient.casOAuthUrl=https://localhost:8443/openiot-cas/oauth2.0
casOAuthClient.casOAuthRestUrl=https://localhost:8443/openiot-cas/openiot1/tickets
casOAuthClient.key=gsn
casOAuthClient.secret=gsn.secret

# Sets the callbackUrl for each client in the list
clients = org.pac4j.core.client.Clients
clients.callbackUrl = http://localhost:8080/sth/callback
clients.clientsList = $casOAuthClient

clientsReale = org.openiot.security.client.rest.CasOAuthClientRealmRest
clientsReale.permissionsURL = https://localhost:8443/openiot-cas/oauth2.0/permissions
clientsReale.defaultRoles = ROLE_USER
clientsReale.clients = $clients
```
Hands-on Configuration of OpenIoT / GSN

- OpenIoT
  - Consoles:
    1. Conductor
      1. *virtuoso (virtual database)*
      2. check to see if the data is correctly inserted
      3. *localhost:8080/conductor/*
      4. *user/pass = openiot/openiot*
      5. *SPARQL console*
      6. *select * from {?s ?p ?o}*
      7. *search for the metadata file/name/ID*

Main Reference: [2-5]
Hands-on Configuration of OpenIoT / GSN

- OpenIoT
- Consoles:

  1. Request Presentation
     1. localhost:8080/ui.requestPresentation
     2. user/pass = openiot/openiot
     3. load application
     4. see the data (loading takes too long usually)
     5. hence:
        1. go to localhost:22001
        2. see your data being pulled
        3. if so everything fine

Main Reference: [2-5]
JOB DONE!
Closing Notes

- Discussion:
  - Service-Oriented Design
  - Distribution
    - Sensor Discovery
    - Service Discovery
  - Limited to a Fixed Ontology
  - Ontology Matching
  - C-Approximate Neighbor detection
  - Fixed Database Schema
  - Dynamic Schema
  - Privacy
    - Data Ownership
    - Privacy-Preserving Aspects
      - Quasi Identifiers
References


