Multi-Thing IoT Applications over OpenDaylight Project Proposal

Vision and Goals of the Project

This project will aim at providing two smart city mobile applications that will help optimize city operations and/or reduce expenditure by integrating IoT technology. The first smart city app is going to be an informational parking app that will tell users the number of available parking spaces in a given parking lot. Our minimum goal for this app is to provide this information for at least two parking lots at Boston University. Depending on how much will be accomplished within the time constraint of this project, more parkings lots will be added to the app. The second smart city app is going to be a Boston University maintenance app that will allow users to submit a photo of something that is broken or requires inspection within BU property. The photo will then be filtered and maintenance faculty users will be notified on the app and be able to look at the photo to see what the problem is. Our minimum goal for this app is for a successful interaction between students and the faculty maintenance team through the app, where the student posts a photo with the problem and the maintenance team receives the photo to inspect the problem. When the problem is fixed, the app will update to notify the users.

Users and Personas of the Project

For the parking app, users of the app will be drivers who are trying to find ideal places to park their cars.This means that the user base will consist of anyone who owns a car and drive within the vicinity of the app's supported parking lots.

Scope and Features of the Project

a. Inside Project's Scope

   This project will uses sensors to collect data, apply logic via a reasoner for analysis, and display the data through a mobile application for its end-users. In addition to the minimum acceptance criteria, this project will provide limited statistical information. For the parking application: capacity percentages, times when capacity has been reached, and average usage. For the maintenance application: total reports, average turnaround time, and throughput.

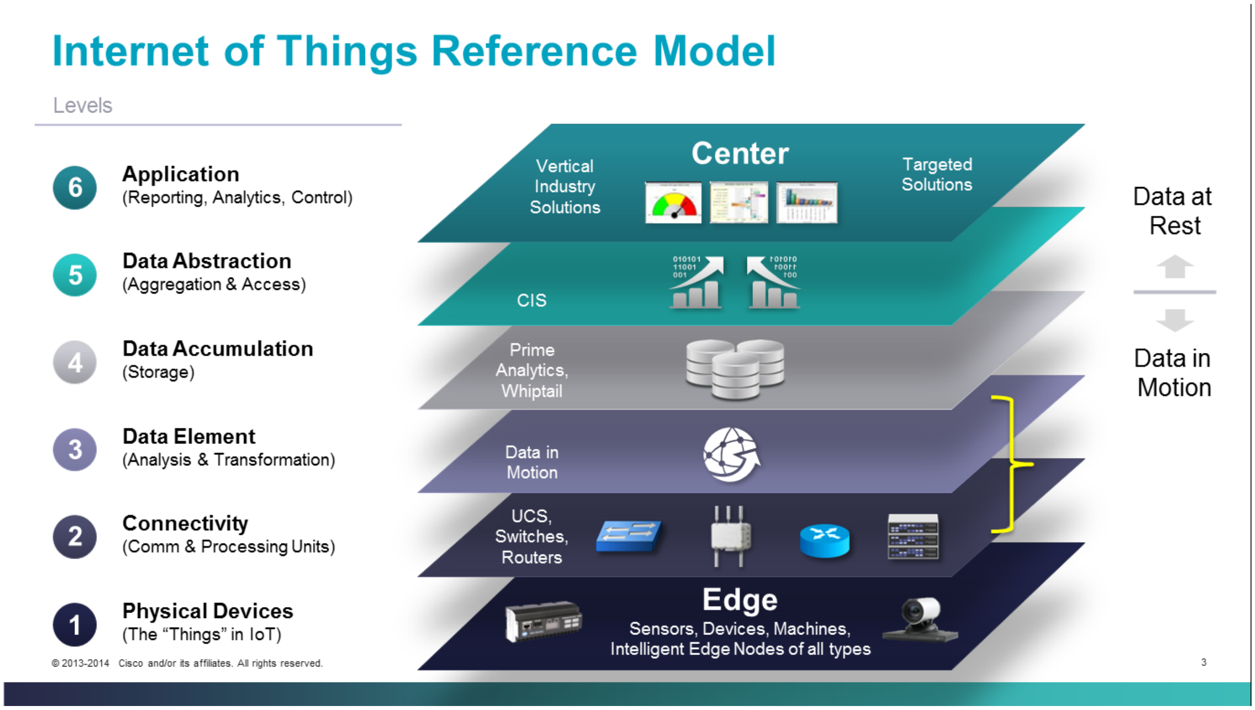
b. Out of Scope

   This project is also limited to use cases within Boston University. The parking application would not be applicable for street/metered parking spaces or parking for vehicles other than cars. The parking application will initially only be tested at

BU parking lots. The maintenance application will be especially targeted for BU students to report problems (not emergencies) around public areas of the campus (not including within dorms/apartments) that directly pertain to BU Facilities Management and Planning.

Solution Concept

The IoT can be divided into layers as shown below:



Internet of Things Reference Model

Levels

6 **Application**
(Reporting, Analytics, Control)

5 **Data Abstraction**
(Aggregation & Access)

4 **Data Accumulation**
(Storage)

3 **Data Element**
(Analysis & Transformation)

2 **Connectivity**
(Comm & Processing Units)

1 **Physical Devices**
(The "Things" in IoT)

© 2013-2014 Cisco and/or its affiliates. All rights reserved.

Center
Vertical Industry Solutions
Targeted Solutions
CIS
010101 11001 001
Prime Analytics, Whiptail
Data in Motion
UCS, Switches, Routers
Edge
Sensors, Devices, Machines, Intelligent Edge Nodes of all types
Data at Rest
Data in Motion
3

Figure 1: Different Layers of IoT Reference Model

Our project will aim at mostly all levels of IoT Reference Model from Physical to Application.
A normal scenario as to how we map IoT Reference Model in our project can be seen as below.

- Layer 1:

Context-aware sensors to enable easy communication and data transfer. Incorporating sensors is targeted towards making unreliable tasks reliable. e.g. an unreliable schedule makes it harder for transit, sensor enables real time data transfer to make it better and reliable and thus making human life easier.

- Layer 2:

Data capture and Data Storage, for our model ODI Data Store, needs a communication mechanism to enable sending data to the storage in real time. This is achieved by adding a layer of connectivity. Sensors can be mounted by networking components on top of sensors.

- Layers 3 & 4:

Collecting Real Time data collection needs a storage module. A data storage with computing capabilities enables easy processing and analysis of data, which is done at these layers.

- Layers 5 & 6:

The analysed/processed data is then presented in the form of a high level application for users in a way that it is understandable by the end users. An access layer's implementation can be achieved for data abstraction with ease but not a prerequisite on implementing the other aspects of the model.

The IoT Reference Model integrated with the technologies used within this project looks something like Figure 2 below.

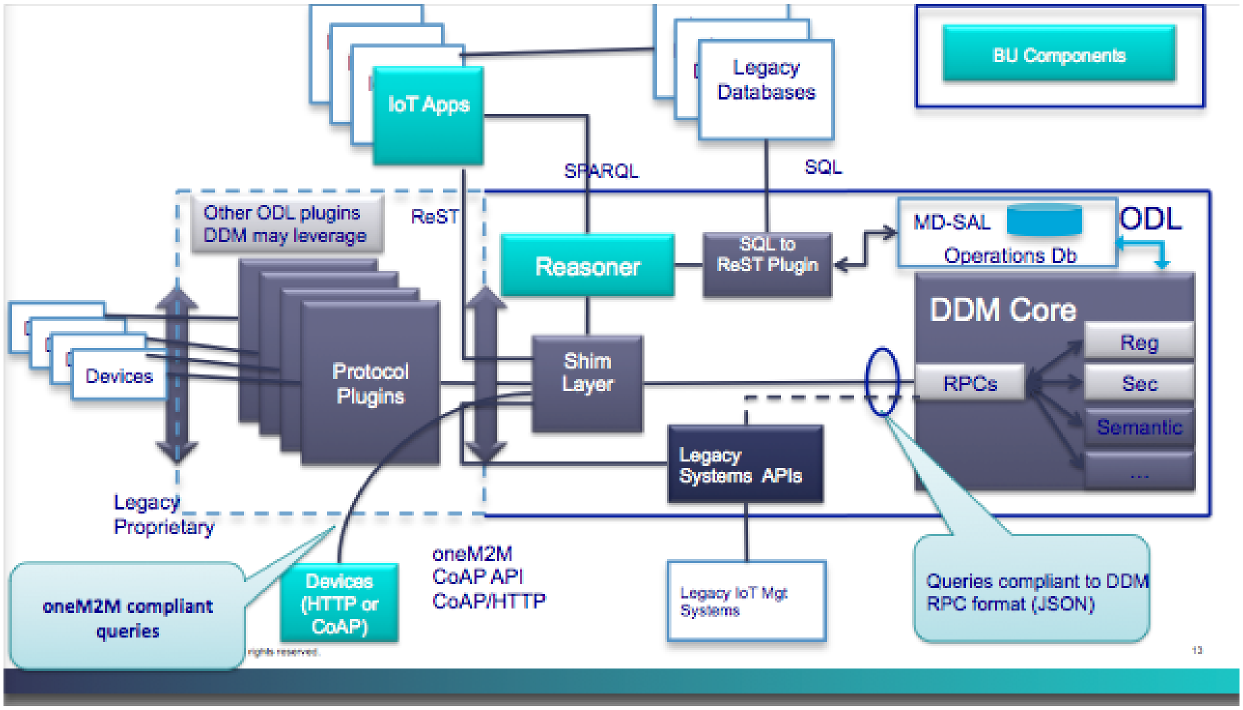
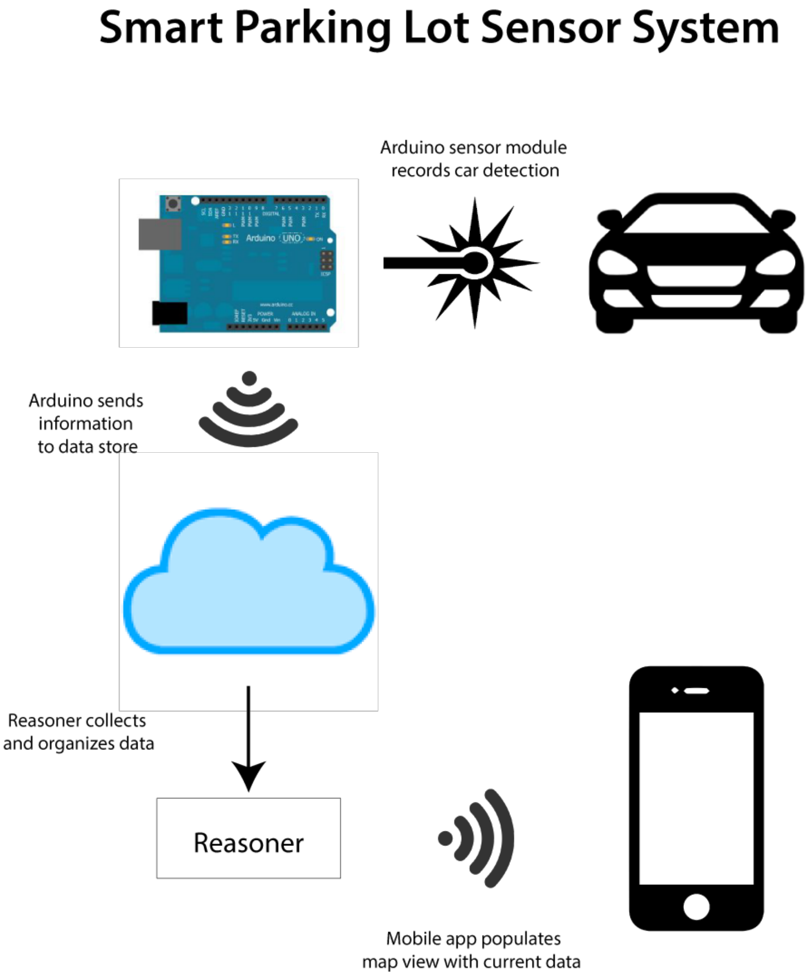


Figure 2: OpenDaylight Architecture and how it will interact with IoT applications and data input devices

Figure 3 below shows how data transfer will take place and overall working of Smart Parking App. Here, an arduino sensor detects presence of car in a lot and sends data accordingly to the ODI Data Store which is then analysed and presented to the user in an application, preferably on maps view.


Smart Parking Lot Sensor System
Arduino sensor module records car detection
Arduino sends information to data store
Reasoner collects and organizes data
Reasoner
Mobile app populates map view with current data

Design Implications and Discussion

- **At Physical Layer and Connectivity Layer :** This project contains two different applications.
    - Our first application is an BU facility administrator reminder. Which uses mobile phones as input devices so as to integrate both Physical and Connectivity layer. Also, this would help in limiting spending on adding networking nodes to some other terminals and may help provide solution with lower prerequisites. It also uses REST application programming interfaces (APIs) for message passing techniques.

  - Another application is an empty parking space finder of BU. The input will be gathered through a set of sensors installed at the entrance and exit of parking lots, when a vehicle enters/leaves the parking lot, the sensor will update the data about vacant parking space in the data store.

- **At Data Storage / Element Layer :** Both of the two applications of the project uses ODI - Open Daylight Data Store.

- **At Data Abstraction and Application Layer :** The model at the higher layer would provide an application mostly in the form of an iOS app written in Objective C keeping in mind the larger scope and target it provides along with the ease of deployment. This would be used to provide Processed Data in a human readable and understandable format.

Acceptance Criteria

The minimum acceptance criteria for this project would be at least two working applications which may be integrated in just one app, using one or more data store to make life easier for the end users and at the same time save them money. These apps have to be easy to use and comfortable to navigate through. The information displayed must be accurate enough for users to depend on them all the time without having to use any other app and they should be mostly bug free and stable.

Both apps must be completed before the deadline and have complete integration with Cisco´s OpenDaylight architecture. Users of the parking app should be able to decide whether or not they should go to the parking lot of their choice depending on how full it is while users of our maintenance app should be able to submit their complaints accompanied by a picture that is successfully sent to BU´s maintenance department. The users will also be able to see if a problem is solved and get notified about it.

Release Planning (Sprint Planning And Schedule Overview)

- Sprint 0
  - Brainstorm ideas
  - Meet with mentors
  - Project Proposal due by end of Week 3 (Feb 8)
  - Setting up development environment and installing OpenDaylight
- Sprint 1 (due by Week 4, Feb 15)
  - End-User Surveys

    - Familiarize with CRUD messages, Postman, and OpenDaylight
    - Research technologies to be used
- Sprint 2 (due by Week 6, March 1)
    - Deploy and calibrate sensors
    - Improve data collection techniques
    - Work with ‘data in’ phase (Create, Update)
    - Create a basic mobile application to display data (Retrieve)
    - Class demos on Feb 26 and March 5

[Spring Break March 9-15]

- Sprint 3 (due by Week 8, March 22)
    - Add more features and functionality to mobile application
    - Test basic logic for reasoner to analyze data
- Sprint 4 (due by Week 10, April 5)
    - Improve reasoner logic for additional data analysis
    - Explore and implement test cases
- Sprint 5 (due by Week 12, April 19)
    - Deploy beta version of application
- Sprint 6 (due by Week 14, May 3)
    - Utilize feedback to make improvements

An ambitious schedule includes putting the application out for use for a network of public networks in the end stage.