IoTivity Services

IoTivity Services, which are built on the IoTivity base code, provide a common set of functionalities to application development. The primitive services are designed to provide easy, scalable access to applications and resources and are fully managed by themselves.

There are four primitive services, each with its own unique functionality.

1.1.1 Notification Manager

This service is responsible for **hosting** the resources of a Lite device or less power/memory capable IoT device by another smart device. It has an **event driven** mechanism to detect the presence of devices whose resources needs to be hosted. It registers a resource as **virtual** in the smart device base to differentiate and give priority to the hosting resource.

1.1.2 Protocol Plugin Manager

This service makes IoTivity applications communicate with non-IoTivity devices by plugging protocol converters.

It provides plugin manager APIs to start/stop plugins and several reference protocol plugins.

1.1.3 Soft Sensor Manager

This service provides physical and virtual sensor data on IoTivity in a robust manner useful for application developers. It also provides a deployment and execution environment on IoTivity for higher level virtual sensors.

Soft Sensor has two main components:

1. **Soft Sensor Manager**
   A service component that 1) collects physical sensor data, 2) manipulates the collected sensing data by aggregating and fusing it based on its own composition algorithms, and 3) provides the data to applications.

2. **Soft Sensor (= Logical Sensor, Virtual Sensor)**
   A software component that detects specific events or changes in a given context by applying its predefined process model with required data.

1.1.4 Things Graph Manager

This service creates **Groups,** finds appropriate member things in the network, manages member presence, and makes group action easy.

It benefits a 3rd party application developer in three ways:

1. Easily collect things for a specific service by the service characteristics, not by each thing's identification.
2. Doesn't require handling, tracing, or monitoring many things themselves.
3. Doesn't require sending control messages to several things.

1.1.5 Control Manager

Control Manager provides framework and services to implement a controller, a controllee and REST Framework for a controller. It also provides APIs for application developers.

Control Manager has three main components:

1. Controller

Uses Smart Home Data Model. Initiates device discovery. Creates RESTful control requests and sends to the controlled devices through Iotivity base. Subscribes to the interested resources.

2. Controllee

Handles the requests from the controller using a RESTful resource request handler. Implements data models for the corresponding resources. Notifies the events to subscribed controllers.

3. REST Framework

Provides REST framework for Control Manager. Applications can use the REST intefaces to invoke the requests on Control Manager using the REST Framework.