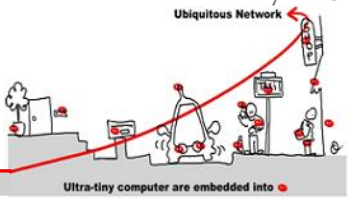
Tutorial: Creating a Web Service for Device: UPnP



The aim of this tutorial is to implement a new UPnP device which will be used inside the WComp framework to realize a composite service (which can be validated using tools you will see during next Course and Tutorial). The interface of our UPnP Device should be the same as the one defined below.

1 INTEL Tools for UPnP Technology

This tutorial is inspired by the Intel video you can find at the following address:

http://trolen.polytech.unice.fr/cours/mit/ videos/

1. First of all, you should install the software for **UPnP Technologies** (latest is v0.0.44). You can download it for the following addresses:

http://www.meshcommander.com/upnptools

or

http://trolen.polytech.unice.fr/cours/mit/eveloperToolsForUPnPTechnologies.msi

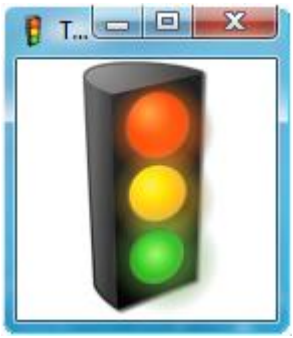
2. Start the "*Device Spy*", which is a "universal" Control Point. This tool allows you to search, discover and control existing instances of UPnP Devices on the local network. You can invoke methods, subscribe to events for any device you want. First of all, you should start the UPnP device Light and verify that you can interact with it with the "Device Spy".

2 Developing your own UPnP Web Service for Device

First of all you should have a look to the "Full Tutorial" video available at the following URL:

http://trolen.polytech.unice.fr/cours/mit/videos/Full Tutorial.wmv

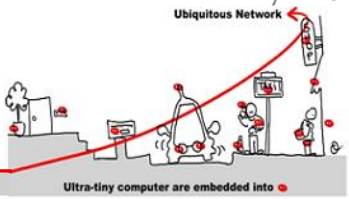
2.1 Traffic Light Device using software tools



- UPnP Devices
  - TrafficLight
    - urn:schemas-upnp-org:service:CarLights:1
      - State variables
        - GreenStatus
        - RedStatus
        - YellowStatus
      - GetStatus(boolean redStatus, boolean yellowStatus, boolean greenStatus)
      - GreenOff()
      - GreenOn()
      - RedOff()
      - RedOn()
      - SetGreen(boolean greenStatus)
      - SetRed(boolean redStatus)
      - SetStatus(boolean redStatus, boolean yellowStatus, boolean greenStatus)
      - SetYellow(boolean yellowStatus)
      - YellowOff()
      - YellowOn()
    - urn:schemas-upnp-org:service:PedestrianLights:1
      - State variables
        - Walk
      - GetWalkingStatus()
      - SetWalkingStatus(boolean status)
      - Stop()
      - Walk()

Figure 1: UPnP Interface of the Traffic Light Device

Tutorial: Creating a Web Service for Device: UPnP



2.1.1 Implementing Traffic Light Services

You should first start by creating the CarLights and PedestrianLights service with the **Service Author** tool (please respect the given names for you to be able to link you generated device to a graphical representation of the device). You should pay attention to define all the variables as evented ones and to define them before defining the methods that use these variables. After saving your defined services, you could have a look to the generated XML files.

When you define the methods, you should pay attention to the way the variables are defined (input, output or return). In the previous schema, if a function is defined with the following name "GetWalkingStatus()", this means that this function does not take any parameter. But as the name can tell it, the function must return a value containing the status of the status of walking variable.

2.1.2 Composing Services into a UPnP Device

Once you have created your UPnP services, you can assemble them inside a Traffic Light Device using the **Device Builder**. With this tool, you can assemble the created services. Pay attention to the parameters that should be modified (take a look to the video).

2.1.3 Generating C# code for your device

The **Device Builder** allows you to generate the source code skeleton corresponding to the description of your UPnP Device. You should pay attention to select code generation for C# Visual Studio and to modify the NameSpace to "TrafficLight".

2.1.4 Implementing your device

With your favorite C# Integrated Development Environment (you can download Sharpdevelop if you don't have any), you should now add the code in methods to give the correct behavior to your device.

2.2 Developing without code generation

The use of the tools presented above is not mandatory. You can also directly write your code in C # for example. In the case of simple devices such as we have described in the previous section, it is quite possible to implement them without these tools. Documentation on the UPnP stack we recovered is not written.

For the creation of a device, you will need to rootDevice = UPnPDevice.**CreateRootDevice** then modify the values of the object returned by this method to change the `FriendlyName`, the `DeviceURN`...

For the creation of a service you have to use listServices = **new UPnPService** then you add the state variables through listServices.**AddStateVariable** and methods through listServices.**AddMethod**. The addition of a service to a device is done with rootDevice.**AddService**. It then remains to call in the constructor to rootDevice.**Advertise**().

Starting and stopping the device are respectively through rootDevice.**StartDevice**() and rootDevice.**StopDevice**().