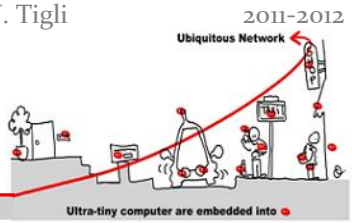
Tutorial 2a: Creating a Web Service for Device: UPnP



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 Lustre and Lesar seen during Tutorial 3). 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://kistren.polytech.unice.fr/cours/oc/td1/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://opentools.homeip.net/dev-tools-for-upnp

   or

   http://kistren.polytech.unice.fr/cours/oc/td1/DeveloperToolsForUPnPTechnologies.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://kistren.polytech.unice.fr/cours/oc/td1/videos/Full_Tutorial.wmv

2.1 Traffic Light Device

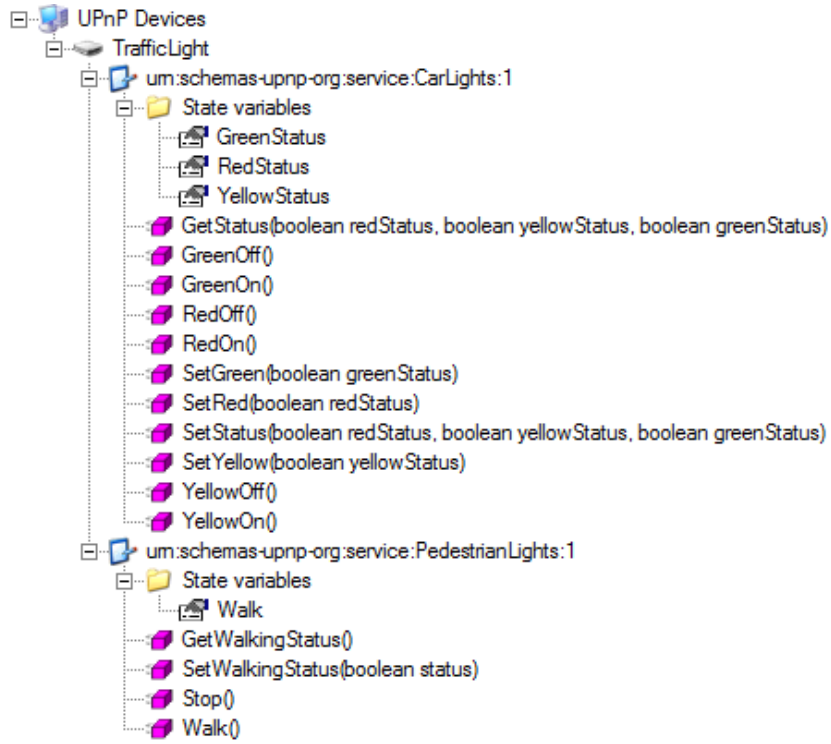
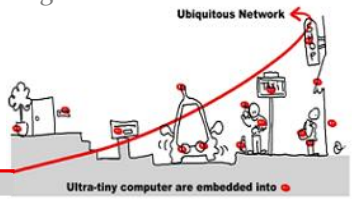


Figure 1: UPnP Interface of the Traffic Light Device

Tutorial 2a: 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 varaibles. After saving your defined services, you could have a look to the generated XML files.

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**.

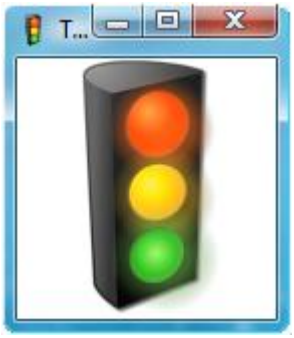
2.1.3 Generating C# code for your device

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 should now add the code in methods to give the correct behavior to your device.

2.2 Graphical representation of your Traffic Light



You can create your own Graphical representation of your Traffic Light, including your generated code inside a WinForms application with your IDE. But to avoid this step, which does not represent any interest for this tutorial, you should get the ready to use Graphical Traffic Light.

http://kistren.polytech.unice.fr/cours/muc/tutorial4a/TrafficLight.msi

3 Using UPnP Device with WComp Middleware

First of all, to be able to use your new UPnP Device with WComp Middleware, you should generate the UPnP Proxy component for the WComp Framework.

3.1.1 Implement the correct semantic for a Traffic Light Device

As you could see, the defined interface of the Traffic Light device allows activating any light without a correct semantic (depending on your country). Define a WComp component assembly to activate the traffic lights with the correct semantic depending on your country. You can create a validated component using Lustre and Lesar (seen during Tutorial 3) to drive the Traffic Light device.

3.1.2 Export the new correct service interface

Using WComp sensors and the SLCA model, export the correct new interface (go and stop).