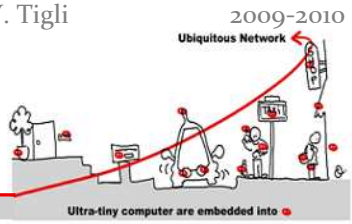
Tutorial 1 : Middleware for Ubiquitous Computing, WComp 2.0



1 Discovery of WComp

You can refer to the documentation available online as well as demonstration videos available for the installation and for taking middleware WComp in hand:

http://rainbow.i3s.unice.fr/wikiwcomp/doku.php?id=download_telechargement

At first, run SharpWComp and choose your language with the menu ""Outils / Options / SharpDelevop options / UI Language ..."

Then, create a WComp File :

- File / New /File ...
- WComp.Net / C # container -> created a new file Container.cs (tab at the top of the workspace)
- To manipulate the components, you must enable the graphical representation of the Container (WComp.NET tab at the bottom of the workspace).

Remember that you can save your components assemblies using the Export option of the menu WComp.NET.

2 Proxy Components for the services composition

2.1 Integration of a Web Service for Device UPnP

We want to access to UPnP devices in WComp. For this, we must generate a proxy component for the discovered UPnP devices. Take for example the Light UPnP device on the CD of the tutorial :

- File / New / File ...
- WComp.NET / UPnP Device WebService Proxy
- Select the Light in the device list, all methods and state variables that you want to access via the proxy component. Click on Next and then on Finish. You've just generated a proxy component for this UPnP device.
- Reload the components to access to the new generated component (Menu WComp.NET / Reload Beans ...)
- Find the component in the category Beans: UPnP Device (Tools tab) and instantiate it in the container.

To test this component and to verify its behavior, connect buttons to control it.

3 LCA Model

3.1 Application programming using components assemblies

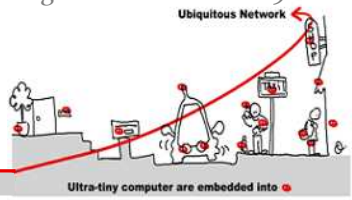
3.1.1 Simple Events

We hope to switch on and off the Light withtwo2 buttons. Create two buttons: one button On and one button Off, and connect the device Light to call methods on () and Off () device.

3.1.2 Complex Events

We now want to also control the Light with a CheckBox.

Tutorial 1 : Middleware for Ubiquitous Computing, WComp 2.0



Create a CheckBox and connect to the proxy component of the Light to control the device using the method *setStatus* (*boolean*).

3.1.3 A more complete application

We will complete this application to become familiar with the components.
If you double click your virtual Light, this has the effect of turning on and off. But you may find that your *CheckBox* does not reflect these changes directly on the device.

- Find a way for this to be the case (having a return to the assembly on a direct modification of the state of the virtual Light).
- We want to show the status of the Light in text in a Label or *TextBox* (consider using the component *ValueFormatter*)
- Connect the components needed to vocalize the state of the Light (using components *BoolFilter*, *PrimitiveValueEmitter* and of course *TextToSpeech*)

3.2 Creating a Component Bean

It may be necessary to create your own components if existing components do not meet your needs. The environment offers the opportunity to create a component from a skeleton.

- File / New / File ...

- WComp.Net / C # Bean -> created a new file Bean.cs

Now, just fill in the template to give the desired behavior to this new component.

We will make a component that adds two integers. This component will have two methods in entries: *intVal* and *intAdd* which allow you to specify an initial value and a second to add a new value. This component will emit an event which is the sum of both. We call this component *AddInt*.

Write the code, compile it and add the library created in the folder containing the components (*C:\Program Files\SharpDevelop\Beans*).

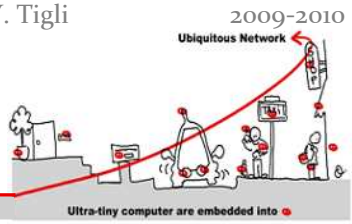
Make an assembly to test your component with two *TextBox* and components *StringToInt* and *ValueFormatteur*. It displays the result in a *Label*.

4 Model SLCA

You've discovered and manipulated the graphical interface that represents the assembly of components into a container. We will now discuss the services that may be involved in order to better understand the architecture SLCA.

4.1 Structural Interface: Handling the assembly by the interface Container Service

Tutorial 1 : Middleware for Ubiquitous Computing, WComp 2.0



To enable this UPnP interface in your container, you must enable the "Bind to UPnP Device" from the WComp menu. Using the tool "Device Spy" from Intel, you can discover the devices associated with the container WComp. We will first take an interest in *WCompNetAppli_Container1_cs_0* device that gives us access to the interface to manipulate the corresponding assembly. This service will allow you to add, delete, components and links.

Test the methods to list the contents of a container (components and links), and try to instantiate new components and links, to delete them with this interface.

4.2 Functional interface: Adding probe components

The second device (*WCompNetProbe_Container1_cs_0*) available is associated with an interface that will allow us to interact with the internal assembly.

The interface of this device is currently empty. The modification of this interface is achieved by adding or removing probe components, sources and sinks. These components are in the category *Beans: Basic* of WComp: *EmitProbe* and *SourceProbe* that will allow respectively to emit events from the service interface into the assembly on the one hand and to emit events from the assembly through the service interface.

Add a component to your EmitProbe assembly. You can see that the functional interface of the device is automatically changed. Using the "Device Spy, by method invocation, you can send a value and therefore generate an event in the assembly of components.

5 Designers

It is possible to use the model SLCA in order to act on the assembly of components from outside (without using the GUI of SharpDevelop). We will present two tools to interact with the assembly of components WComp.

5.1 Textual Designer

The TextDesigner can create components and links using a basic command language. Here are some links:

- CheckBeans

- CheckLinks

- CreateBean <*BeanType*> <*InstanceName*>

This allows you to create scripts to build and modify assemblies.

5.2 UPnP Designer

The UPnP Designer will enable to automatically generate the proxy component and instantiate it in the container, for each new UPnP device.

To test it, after launching WComp and UPnP Designer, enable the connection (to Bind on UPnP Device WComp) and run a new UPnP device. Automatically, the proxy component code is generated and a component of this type is instantiated and configured with the right properties (UPnP server address, etc..).