Tutorial

Publish/Subscribe Middleware in TinyOS 1.x

This tutorial explains the installation and usage of the Graphical User Interface (GUI) for the publish/subscribe middleware in the *contrib/eyes* tree of tinyos-1.x. It describes how the publish/subscribe demo application (*PSDemo*) is installed on network nodes and the standard TinyOS application *TOSBase* on the gateway node. The GUI will then allow to disseminate subscriptions in the network and display notifications arriving from network nodes by different ways of visualization.

1. Overview

The GUI exports access to a sensor network based on the content-based publish/subscribe scheme. To query the network for data, a user may issue *subscriptions*, which consist of *constraints* over the value of attributes. A constraint has the form (attribute, operation, value), for example a subscription that is meant to subscribe to temperature data below 20 degrees is represented by a single constraint (temperature, $<$, 20). A subscription may consist of multiple constraints which are then logically "AND"ed.

A subscription is issued via the GUI and then disseminated in the network where it is registered in the publish/subscribe middleware on every node (except for the gateway node). Network nodes then publish data in form of *notifications*. The publish/subscribe middleware compares the notification with the registered subscriptions and, if there is a match, notifications are routed back to the subscriber and displayed in the GUI.

The basic setup for the publish/subscribe demo application is depicted in Figure 1: It requires the TinyOS application *PSDemo* to be installed on all network nodes, *TOSBase* on the gateway node and the GUI running on the end system (PC, Laptop). Subscriptions are (by default) disseminated using the Drip protocol and notifications are sent back via the tree-based routing protocol Drain. This tutorial explains the setup for the eyesIFXv2 platform, other platforms may need according modifications.

2. Compiling the Java Classes

The following steps deal with the compile process of java classes used by the GUI. These steps have to be done only once and most of the compilation process is automated by the *make* system.

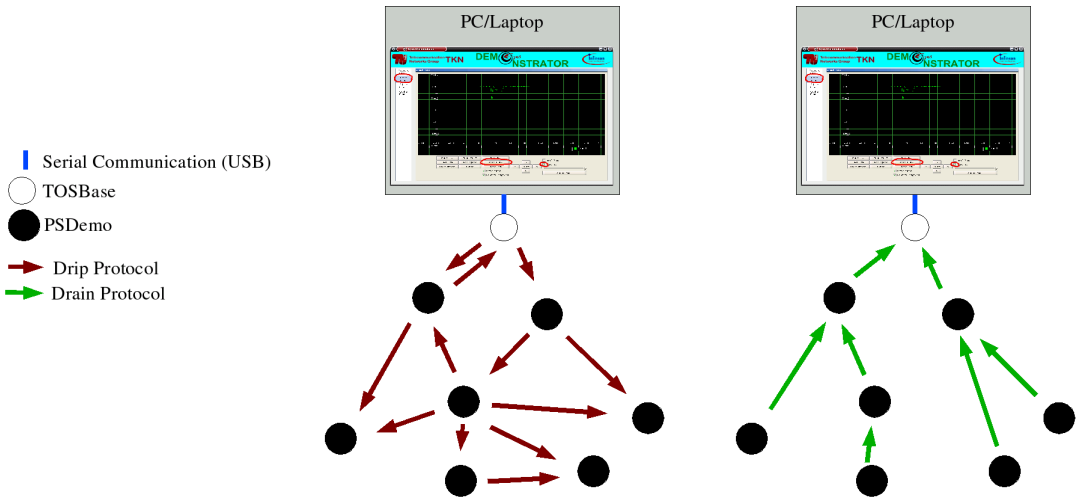

Figure 1: Basic overview: Subscriptions are disseminated via Drip (left side), notification are collected via the tree-based routing protocol Drain (right side). TOSBase is installed on the gateway node, PSDemo on all others.

1. Before you can start to compile the necessary java classes, make sure that your `CLASSPATH` environment variable is setup correctly and includes *$TOSDIR/../contrib/eyes/tools/java*[1]:

   ```
   export CLASSPATH=$TOSDIR/../contrib/eyes/tools/java:$CLASSPATH
   ```

2. First of all, make sure that you have compiled the generic TinyOS java classes. Type

   ```
   cd $TOSDIR/../tools/java
   make
   ```

3. Now the platform specific java message classes have to be compiled. Type[2]

   ```
   cd $TOSDIR/../contrib/eyes/tools/java
   make
   ```

4. Finally the GUI classes can be compiled. Type

   ```
   cd $TOSDIR/../contrib/eyes/tools/java/de/tub/eyes/apps/demonstrat
   ant compile
   ```

[1] Or a corresponding directory for any other platform
[2] Choose the corresponding directory for any other platform

3. Installing PSDemo Application

The *PSDemo* application is a TinyOS application that can publish notifications for all relevant sensor attributes on the platform (eyesIFX: temperature, light, RSSI, ...). It has to be installed on all network nodes (actually all but one, because we need one node as the gateway) that are going to take the role of a publisher. The GUI, i.e. you, will be the subscriber. How the binary is transferred to the node depends on your system configuration (e.g. on what USB-port you have attached the node), but you should make sure that the nodes all have different IDs to identify them later in the GUI. The *PSDemo* application is located in *contrib/eyes/apps*, type

```
cd $TOSDIR/../contrib/eyes/apps/PSDemo
```

Compile the application by typing

```
make eyesIFXv2
```

If you get compilation errors (warnings are OK), make sure you have as nesC version at least 1.2beta installed (type `nescc -version`).[3] Now flash the binary on the first node by typing for example

```
make eyesIFXv2 reinstall,56 bsl,1
```

Do this for all (but one) nodes and chose a unique ID each time (in the example 56 is the ID and the node is attached to port ttyUSB1).

The default subscription dissemination routing protocol we chose (Drip[4]) requires the standard TinyOS application TOSBase to be installed on the gateway node. Thus for the last node type

```
cd $TOSDIR/apps/TOSBase
make eyesIFXv2 install bsl,1
```

(Mark the gateway node, in case you want to use the setup later again.)

4. Running the GUI

Attach the gateway node to your PC and put batteries into the network nodes, they should briefly flash 3 LEDs after reset to show that they are ready. Before you can start the GUI you need to start the *SerialForwarder* program on you PC [5], which listens for TinyOS messages on the serial/USB port and exports them on localhost:9001 (by default). To start SerialForwarder, e.g. for a node attached to ttyUSB0, type

```
java -Djavax.comm.SerialPorts=/dev/ttyUSB0 \
-Djavax.comm.ParallelPorts= net.tinyos.sf.SerialForwarder \
-comm serial@/dev/ttyUSB0:eyesIFXv2 -port 9001
```

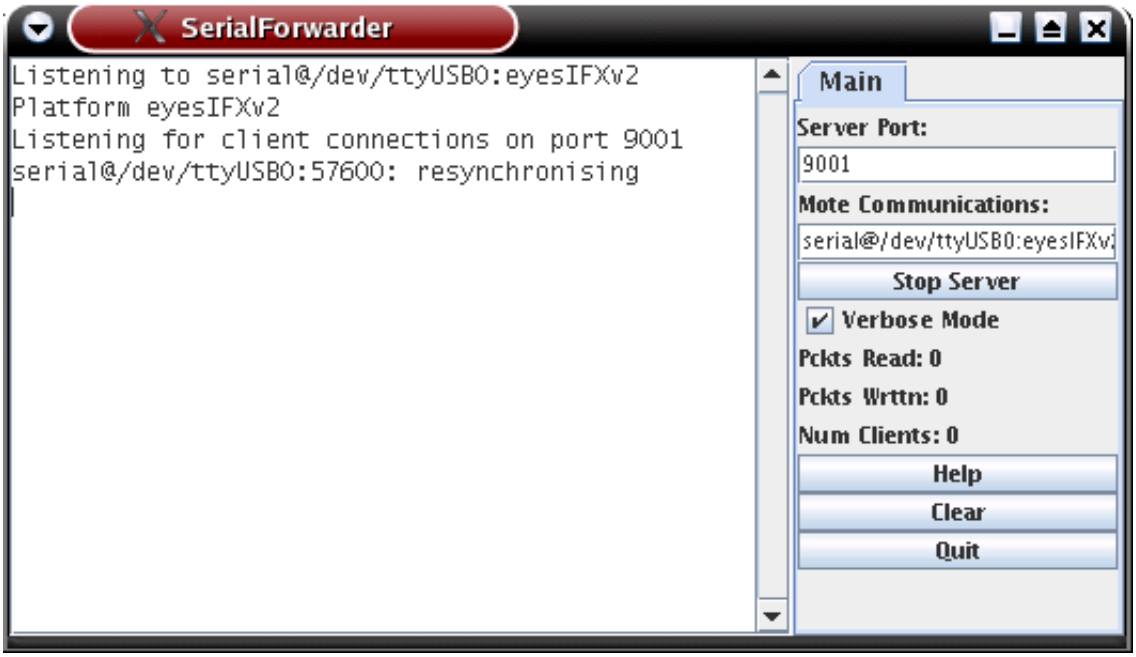

Figure 2: SerialForwarder (GUI version).

You should see a window as shown in Figure 2.

You can now start the GUI by typing

```
cd $TOSDIR/../contrib/eyes/tools/java/de/tub/eyes/apps/demonstrator
ant run
```

and should see a window as shown in Figure 3.

In order to issue a subscription, click on "Subscr Panel" in the left frame and then on the "New" button as shown in Figure 4.

In the pop up dialog you can choose the data that you want to subscribe. On the the left hand side all attributes are available in a drop-down menu, while the right side shows commands. There is currently only one command available, `RATE`, which specifies the frequency with which publishers should publish notifications (in milliseconds). Figure 5 shows an example subscription for `LIGHT` and `TEMPERATURE` at a `RATE` of 1 second. Note that constraints over attributes are logically "AND"ed, that means, all attributes must match, before a notification is published by a network node (in the example "ANY" is chosen resulting always in a positive match) . A more detailed overview of attribute semantics (and metrics) can be found in `$TOSDIR/../contrib/eyes/tos/lib/PublishSubscribe/attributes/README.txt`.

The subscription is sent into the network after clicking "Subscribe". Whether it has reached the gateway node can be determined by a look at the SerialForwarder ("Pckts Wrttn" number

[3]Otherwise download here http://nescc.sourceforge.net/

[4]http://www.tinyos.net/scoop/story/2005/2/16/174147/450

[5]http://www.tinyos.net/tinyos-1.x/doc/tutorial/lesson6.html

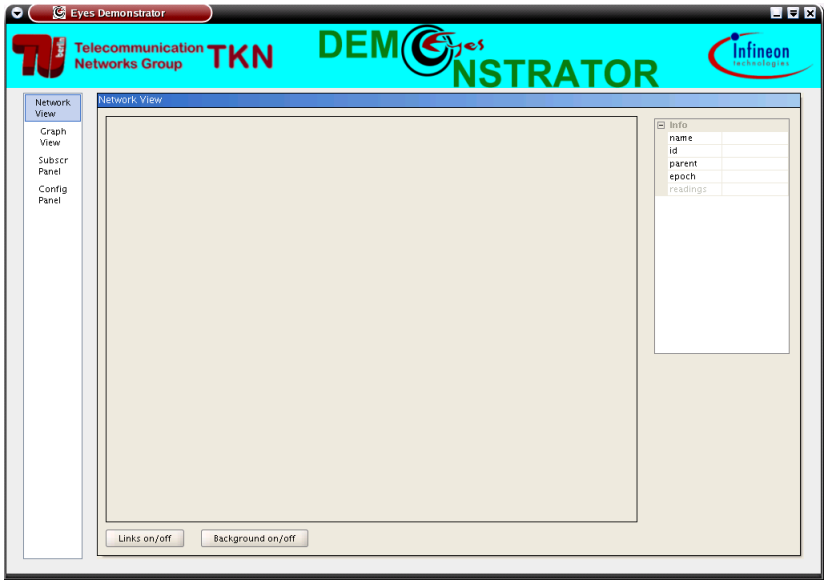

Figure 3: The Publish/Subscribe GUI after starting.

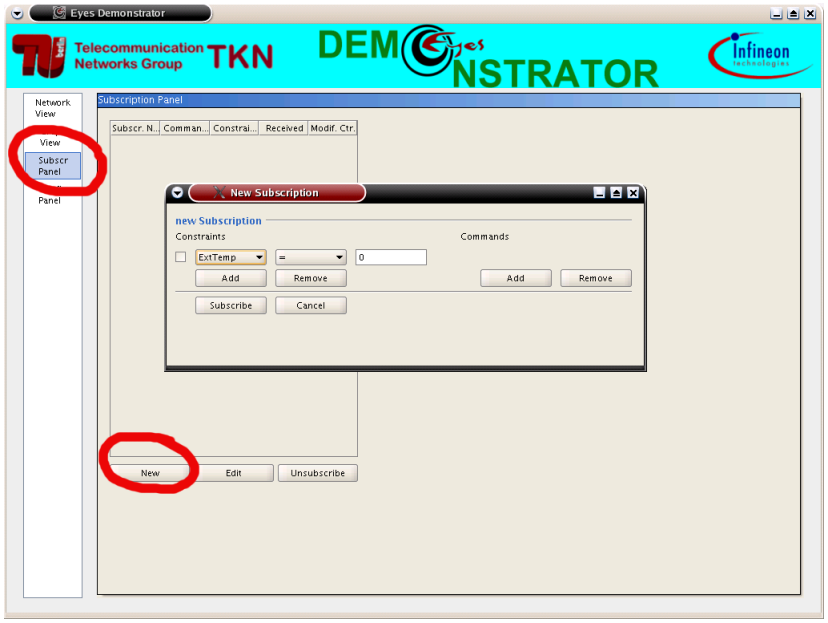

Figure 4: Issuing a new subscription.

should increase) and the gateway node, which toggles the red/first LED (counting from the USB interface side on eyesIFXv2) whenever a packet is bridged from the PC to the network - the meaning of all LEDs is explained in the appendix. Note: This LED should already toggle periodically (with low frequency), because the Drain (tree-based routing) protocol constantly sends beacon messages via the gateway to maintain the routing tree.

Whether a subscription has reached a network node can be determined by the toggling of the red/first LED (counting from the USB interface side on eyesIFXv2). Because we run

Figure 5: Choosing attributes and commands for a subscription.

the Drip protocol for subscription dissemination, it is sufficient that a single node receives the subscription, because then it will be spread in a gossip-like fashion to all other nodes eventually. This also means, that you may insert network nodes later on without the need for reissuing the subscriptions. Drip will take care that new nodes will be updated to receive the most recent subscription(s)[6].

In order to visualize the network topology (the routing tree) and the data encapsulated in notifications, you should first configure the desired visualization by clicking on "Config Panel" in the left frame, marking the attributes and clicking "Apply" as shown in Figure 6.

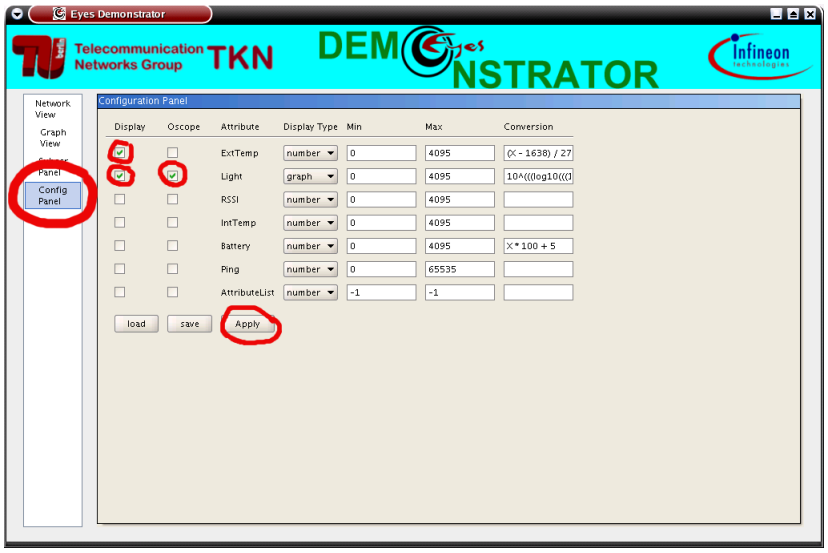

Figure 6: Configuring the visualization .

When you now switch to "Network View" by clicking on the according button in the left frame you will see the network topology as well as a visualization of the attribute data (`LIGHT` as a gradient and `TEMPERATURE` as a number). To display data of a particular node highlight the node ID on the right frame in the "Network View" as shown in Figure 7.

[6]Drip uses the Trickle algorithm: http://csl.stanford.edu/ pal/pubs/trickle-nsdi04.pdf.

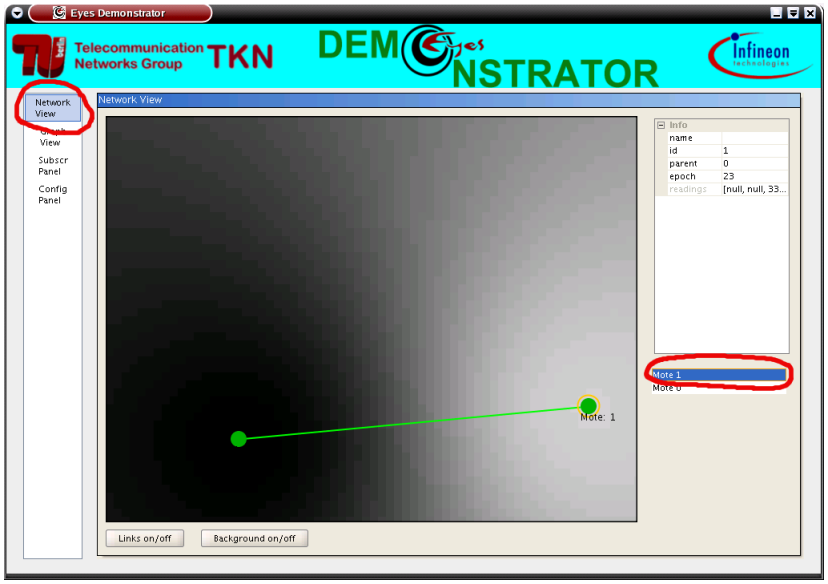

Figure 7: Choosing a node for detailed examination.

Now switch to "Graph View" by clicking on the according button in the left frame. Click "Zoom Out Y" until the Y-Axis shows values up to 4000. Then check the "Scrolling" checkbox as shown in Figure 8. You should now see the a data stream of the attribute that we marked in the "Config Panel" in the checkbox "Oscope" (in the above example we marked `LIGHT`).

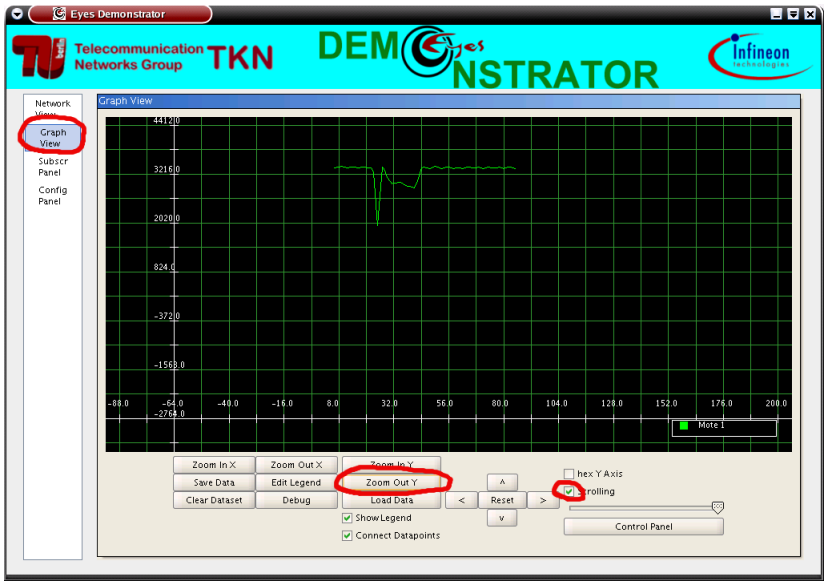

Figure 8: Displaying a data-stream in the oscilloscope view.

In order to unsubscribe or edit a subscription switch to the "Subscr Panel", mark the subscription and click "Edit" or "Unsubscribe".

5. Contact

Author: Jan-Hinrich Hauer, email: hauer@tkn.tu-berlin.de
TinyOS implementation by Jan-Hinrich Hauer, the GUI was designed and implemented by Till Wimmer: wimmer@tkn.tu-berlin.de.

A. Appendix

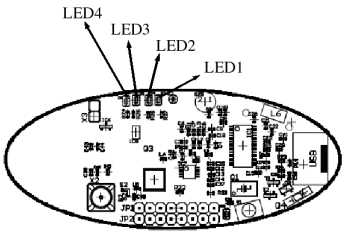

Figure 9: LEDs naming.

LED	meaning
LED1	Toggles when new subscription was received.
LED2	Toggles when application starts gathering data.
LED3	Toggles when a notification has been published and the message has successfully been sent ou
LED4	Toggles when Drip is active.

Table 1: LEDs in the **PSDemo** application.

LED	meaning
LED1	Packet routed from PC to WSN.
LED2	Packet routed from WSN to PC.
LED3	always off
LED4	always off

Table 2: LEDs in the **TOSBase** application.