Update to the Monitoring model

Augusto Ciuffoletti

November 14, 2012

On Monday night Ralph pointed out (in a private mail) that the Sensor Mixin cannot be defined as a subcloss of "mixin", but simply as an instantiated object. This makes a problem to me.

Basically, I do not want to leave just on paper the basic concept, i.e., the Sensor. So I need to rearrange things, in order to make them consistent with the core, while defining a sensor object that can be type-checked statically or at runtime, i.e. a real type or class. Additionally, according with the KISS principle, the number of formalized concepts must not increase, and should possibly decrease and simplify.

My proposal is to eliminate the sensor as a mixin used as an aggregator of monitoring Resources. Instead it is defined as a Resource, characterized by the timing attributes that we already know. It has links (named Controls) to the Tools. Tools, like in the previous model, are defined as Resources. As in the provious model, the Tool is totally empty, and its attributes are all defined by mixin. The links equally have no attributes. There is no reason why Sensors and Tools do not have a state transition on/off controlled by an action, but this is not essential in the model, apparently. Several tools can be associated with the same Sensor.

Resource state transitions, in my mind, are useful for accunting services and for other reasons, and should be included in the Resource definition itself. But this is another story.

A novelty comes from the filters: the Filter resource disappears, and fitering capabilities reappear as a mixin for the Sensor.

A filter mixin is typically specified by an Id attribute and by three kinds of attributes: input parameters (URIs associated with a local attribute), a filtering process, and output parameters. Output parameters are relative to the Id, have a local scope, and can be reused elsewhere in the same Sensor specification. Input parameters are referenced relative to the Sensor the filter is mixed in. Ideally, the provider should arrange things so that, when a new Sensor is allocated with a given filter, the filtering process in configued to accept input values by way of an asynchronous channel (e.g, a pipe), while the designated Tool pushes new values on the channel. This looks like the optimal option, but the provider is free to select another.

Publishers are also defined as Mixins for the sensor, and are not considered as Resources anymore. A typical publisher indicates the Sensor attributes it takes care of, and specifies a treatment for those data. Not very different from the Filter, after all. There are no output paramentes but a link connects the sensor to another resource that is the destination of the data. The Publisher link has been removed, since it is not of particular interest from the discovery point of view. In addition, it wuld request the definition of a target resource specific for receiving monitoring results, and this looks useless complication.

The range of Publishers types is extremely wide. To name a few that are of specific interest:

- piping: this can be UDP or a local pipe;
- TCP connections
- recording into a database or storage
- production of triggers in an ESB
- no Publisher: data remains available through the URI

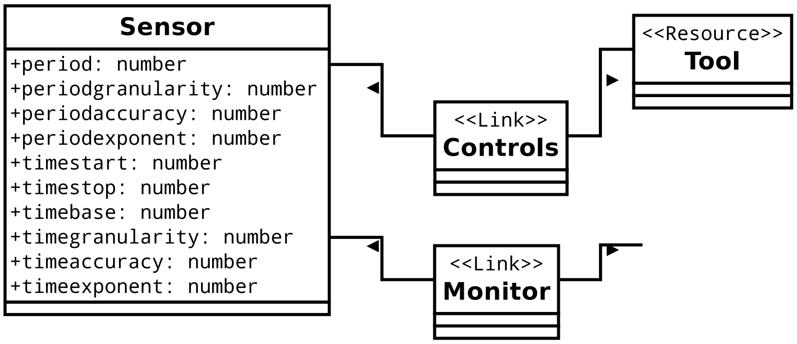

Figure 1: The new class diagram (mixins now are **not represented**)

In fact, Filters and Publishers are so similar that they might merge in the same concept.

A Sensor is linked to one or more Resources (typically *several* resurces) it monitors with Monitors link. The Link has no attributes. The same Resource can be monitored by one or more Sensors. Through the link we discover what kind of monitoring is operated on a Resource, and how it is available. In addition to authentication to the publisher the Link can be protected so that only authorized users can have access.

In the simple case that a Sensor is monitoring a single Resource, the Sensor can be omitted, and the Tool can be directly linked to the monitored resource, and the Filter and Publishing mixins aggregated with the Resource itself: this option should be limited to the ”one-resource” (singleton) Sensors, for instance preventing the definition of ”Monitors” links that do not originate from a Sensor.

The existence of a Publisher and Filter mixin with the function of a tag might be enforced including it in the occi namespace.

In conclusion: we now have 2 resources, and two types of links: this makes 4 boxes, one less than before, arranged on a single line, and the strange mix-in has been replaced by a link. The Sensor is now a Resource. There are simplification options for simple cases, and the whole thing is discoverable in a single hop from the resource (no hops if the Resource contains a singleton Sensor).