SourceVsGenerator

A frequent question comes when you want to add to Cocoon a new way to obtain data : "should I write a Source or a Generator" ?

As you may already know, a Source is a means to access data, and makes no assumption on what kind of data it is. It provides the low-level plumbing to access the data, and makes no assumption whether that data is XML or even can be represented as XML (e.g. GIF images).

On the other side, a Generator is the first stage in a sitemap Pipeline and produces XML from a given environment. We have some generators that rely on sources, other that don't. The RequestGenerator for example produces some XML, but doesn't rely on a Source.

Source-related generators take a Source as input and generate XML from the data given this Source. The most well-known is the FileGenerator, which feeds an XML parser with the Source data. This makes the important assumption that data is an XML text. But there are also other source-related generators that accept non-XML data : the SWFGenerator reads a Flash document from a source and converts it to XML.

So a source-related generator should be considered as a parser (in the general meaning of the word), that takes data from a source and produces XML.

Now that things are clear, lets add some confusion 😉

Some sources provide access to native XML datasources, such as xmldb. For these sources, there is no need for a parser since the data can already been accessed in its XML form (i.e. DOM or SAX). Moreover, it would be a waste of time to ask the source to serialize XML data to textual form and then feed a parser. So these Sources implement the XMLizable interface to provide direct access to XML data.

Cocoon's SourceResolver takes care of that and provides a `toSAX(Source)` method that decides to use a parser or not depending on the XMLizable nature of the Source. Of course, if the source isn't natively XML, this assumes it provides XML in its textual form.

So how to make the choice ?

Nearly all generators could be rewritten as sources, for example the RequestGenerator could be written as a "request:" protocol. But does this make sense ? In that particular case, no.

A new protocol makes sense if several, different data (documents, pieces of information) can be obtained using this protocol. For example using an FTP protocol you can fetch several kind of files from the FTP server. A request protocol on the other hand addresses only one piece of information, the request.

- Sources have a wider usage range than generators. Cocoon uses sources everywhere it needs to access data, since sources provide independence against the access means. Sources are used to read XSL stylesheets, the sitemap, XSP source files, images (in readers), etc.
- Generators are sitemap-only components, but are more easily configured than sources, either using configuration in `<map:generator>` or parameters in `<map:generate>`