chem-bla-ics

Details behind the “Calling XMPP cloud services from Taverna2”

Egon Willighagen

Published January 21, 2009

Citation

Willighagen, E. (2009). Details behind the “Calling XMPP cloud services from Taverna2”. In *chem-bla-ics*. chem-bla-ics. https://doi.org/10.59350/1zmv1-tnn62

Keywords

Xmpp, Taverna

On Monday I showed two screenshot showing our new XMPP-based web/cloud services in action inside Taverna.

I promised details, but realize I have actually already posted a lot of them in October :

> *Johannes ideas led to the IO-DATA proposal (XEP-0244), which is currently marked experimental and being discussed on the ws-xmpp mailing list. He gathered a few people around him to get it going, resulting in working stuff! Yeah!*

Joerg asked *Could you post more results, what is it, why do we need it, e.g. why are you mentioning SOAP and cloud? Do not know enough to see the bonus right now.*

What is it? IO-DATA is a protocol on top of the XMPP protocol to allow machine-to-machine communication. Actually, much like SOAP, RPC, and other platforms. How IO-DATA differs lies to some extend to the transport layer: instead of using HTTP, it used the XMPP transport protocol, also used for Jabber chat clients. It basically allows clients like Taverna to chat with services running elsewhere.

Why do we need it? Most services run over HTTP, making them web services. This is convenient, because there is much infrastructure around, like web browsers. REST services also take advantage of this. However, for heavy computing this sometimes leads to problems. For example, routers are known to have time outs on HTTP connections. To solve this, SOAP services often introduce a polling mechanism. IO-DATA takes a different approach. Instead of having to ask all the time how a calculation is doing, you can just wait for the service to send you a message when it is done. Instead of working around the lack of asynchronous aspects, IO-DATA introduces these in the protocol.

Other interesting features include that the IO-DATA integrates the interface formats for services into the service itself, SOAP needs WSDL for this, and that it features service discovery via DISCO. The latter is done with SOAP too, for example with UDDI and BioMoby. The latter also adds strong data typing for input and output of services.

IO-DATA addresses the data typing by allowing asking the service what XML Schema it uses for input and output. While XML Schema has alternative, and which may be prefered in some situations, it does allow strong data typing and supports a lot of formats in life sciences (which I'll summarise soon).

Moreover, if there just happens not to be a suitable schemata around, you can just define one yourself, which can be as simple as a single element wrapper around some custom text-based format. You worry about supporting many formats? Well, no need. Johannes' xws4j library, which I used for the Taverna plugin too, allows compiling a Java binding code. Bioclipse's script environment allows you do to this on the fly: you find a service, ask for the schema, compile bindings for input and output, set up the input with the input binding, send it of to the service, and use the output binding for convenient access to the computation results. Without having to reboot Bioclipse. Isn't that **cool**? Can your software do that? (See this example Gist: the io factory creates the binding).

Why do I mention SOAP and the cloud? It should be clear from the above why I mention SOAP: it offer the same functionality, but more conveniently, we think. I mention cloud here, to refer to cloud computing which is doing computation on the cloud, which is a synonym for the internet (see Cloud Computing @ Wikipedia). Because it does not use HTTP, we do not feel we can call it web service. Instead, cloud computing is a more general term, not tied to any particular architecture. IO-DATA is just one possible architecture, one we think is promising for life science applications.