Next generation asynchronous webservices

Egon Willighagen ⓘ

Published October 31, 2008

Citation

Willighagen, E. (2008). Next generation asynchronous webservices. In *chem-bla-ics*. chem-bla-ics. https://doi.org/10.59350/7gx9j-dn554

Keywords

Bioclipse, Xmpp, Soap, Http

Abstract

Johannes joined a Bioclipse Workshop a long time ago, and introduced the participants to the idea of using XMPP (aka Jabber) for asynchronous web services. SOAP is commonly user to run webservices over HTTP, but via (SMTP) email and XMPP is possible too (see SOAP over XMPP). Using HTTP as transport layer has problems. The biggest problem, is possibly that HTTP connections are timed out, e.g. by intermediate router.

Copyright

chem-bla-ics

Johannes joined a Bioclipse Workshop a long time ago, and introduced the participants to the idea of using XMPP (aka Jabber) for asynchronous web services. SOAP is commonly user to run webservices over HTTP, but via (SMTP) email and XMPP is possible too (see SOAP over XMPP). Using HTTP as transport layer has problems. The biggest problem, is possibly that HTTP connections are timed out, e.g. by intermediate router. This makes it rather unsuited for long running jobs. Workarounds are easy to come up with, and *polling* is a common solution.

Johannes ideas solve this limitation by using the general XMPP protocol for chatting:

client

he, can you do something for me?

service

sure, I can do **generate3Dcoordinates** and **generateSMILES**.

client

ah, nice! what input does **generateSMILES** take? and the output?

service

input: CML, output a simple string.

client

ok, here's the CML

service

I'm done now. sorry that it took 10 minutes, but I'm running Vista...

client

excellent, please send me the results

service

ok, here is the SMILES for lacosamide: [CC(=O)N[C@H](COC)C(=O)NCC1=CC=CC=C1] {.chem:smiles}

Well, the important bit is in the last line. A job may take lone, even on clusters. The client might have to reboot meanwhile (possibly because of critical security updates)... the *service* will just continue, and send you a message when done. If you just happen to be offline, it will send a message when you are back online.

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!

Chemistry Development Kit XWS

Besides contributing to the proposal, I am also involved in this project by writing XMPP-webservices, for the CDK. This brings me to cdk-xws, which is the project to bring CDK functionality online as webservices using IO-DATA.

chem-bla-ics
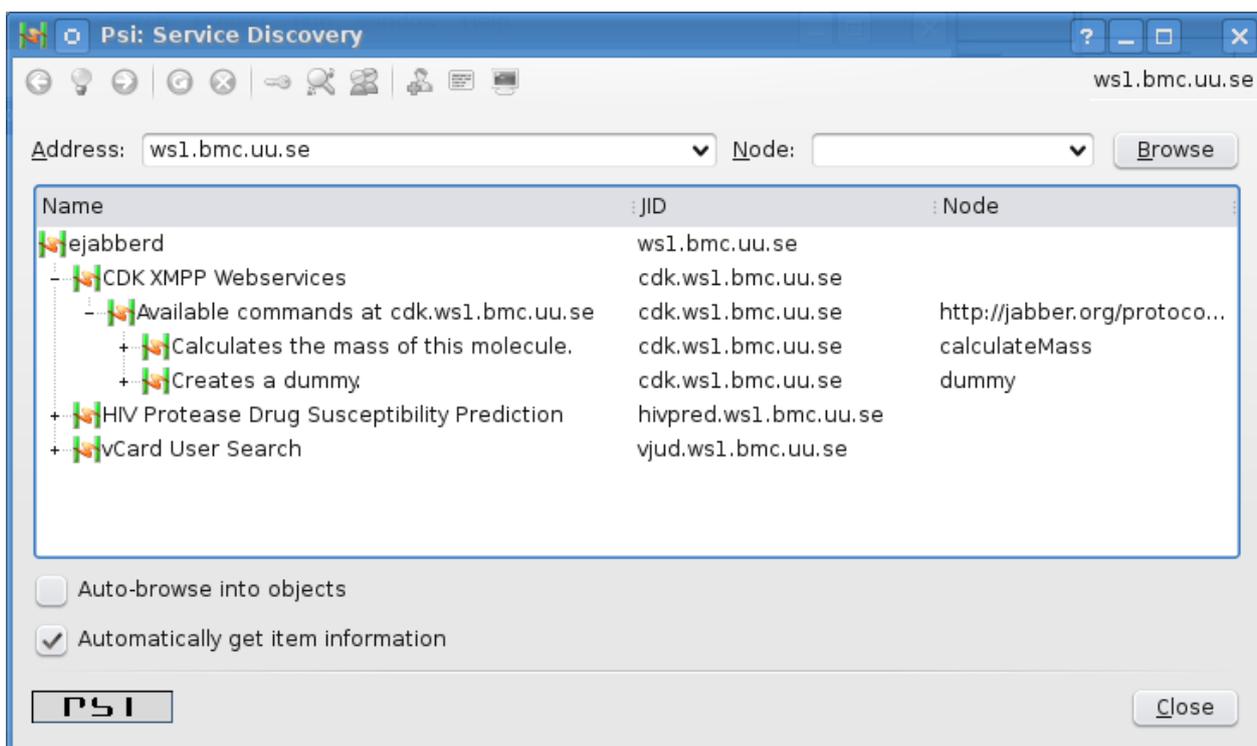
This shows three nodes, the first being the CDK service, with two functions, of which I only implemented one yet.

For the curious, this is what the XMPP messages look like:

```xml
<iq from="egonw@ws1.bmc.uu.se/home"
    id="JSO-0.12.5-6"
    to="cdk.ws1.bmc.uu.se"
    type="set">
  <command xmlns="http://jabber.org/protocol/commands"
           action="execute"
           node="calculateMass">
    <iodata xmlns="urn:xmpp:tmp:io-data"
            type="input">
      <in>
        <smiles xmlns="urn:xws:cdk:input">CCC</smiles>
      </in>
    </iodata>
  </command>
</iq>
<iq from="cdk.ws1.bmc.uu.se"
    id="JSO-0.12.5-6"
    to="egonw@ws1.bmc.uu.se/home"
    type="result"
    xml:lang="en">
  <command xmlns="http://jabber.org/protocol/commands"
```

```
        node="calculateMass"
        sessionid="XWS-1"
        status="completed">
    <iodata xmlns="urn:xmpp:tmp:io-data"
         type="output">
      <out>
        <mass>36.032207690364004</mass>
      </out>
    </iodata>
    <note type="info">Done</note>
  </command>
</iq>
```