

Autogenerating CML bindings for XMPP services with XMLBeans

Egon Willighagen 

Published March 14, 2009

Citation

Willighagen, E. (2009, March 14). Autogenerating CML bindings for XMPP services with XMLBeans. *Chem-bla-ics*. <https://doi.org/10.59350/et43s-kc120>

Keywords

Cml, Java, Ubuntu, Xml, Xmpp

Abstract

I blogged earlier about our efforts to create a better SOAP service architecture, based on XMPP:

Copyright

Copyright © Egon Willighagen 2009. Distributed under the terms of the [Creative Commons Attribution 4.0 International License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

I blogged earlier about our efforts to create a better [SOAP](#) service architecture, based on [XMPP](#):

- [Details behind the “Calling XMPP cloud services from Taverna2”](#)
- [Calling XMPP cloud services from Taverna2](#)
- [Next generation asynchronous webservices](#)

So, I set up XMPP services for QSAR descriptor calculation, 2D diagram and 3D geometry calculations and a few more, using the [CDK. Chemical Markup Language](#) has been my primary choice for some 10 years now (see [Peter’s blog](#)) as it allows me to do things I cannot do in other formats.

Now, our XMPP services publish themselves what data types they allow as input and what they output in return. They do this by publishing XML Schema to describe the input and output types. My CDK services use CML, so they return the CML schema. Johannes’ [xws4j](#) implementation of the [IO-DATA](#) specification has an add on that can build bindings to the schema on the fly. Now, CML comes with a good [XOM](#)-based binding (called [CMLXOM](#)) so this is not strictly necessary, but for less common schemata it is worthwhile: you can always create bindings for brand new schemata, for older versions, for whatever. Services can even create their own local schemata, and people will still be able to easily use them. This is to me a big plus for this architecture.

Anyway, while CMLXOM exists, we wanted to show that the on-the-fly creation of bindings works, even for large schemata, such as CML. However, one of the older flavours had a small error in a regular expression in a data type CML defines. Johannes therefore asked me to test building bindings for the CML schema version used in my services. He advised me to use [scomp](#) for this, which is a command line utility around the [XMLBeans](#) library used for the binding generation.

As I am running [Ubuntu](#), I preferred installing [the packaged version](#) instead of installing the binary provided by XMLBeans. Now, after I did this, I noticed that this .deb did not install the [scomp](#) utility, so I filled a [wishlist bug report](#). Earlier this week I already encountered another bug, but this package being Java, I had a good idea on how to fix the bug.

And so I implemented my own wishlist. I’m sure there is room for improvement, as my .deb packaging skills are a bit rusty (a very long time ago I have been in the Debian New Maintainers queue, but by the time they solved the long queue delays, I was too occupied with other things. Yes, this was a long time ago already :). Anyway, Ubuntu’s [LaunchPad](#) has a nice feature, called the [Personal Package Archives](#). This service will, after I have finished hacking on the packaging specs in the famous *debian/* folder and tested the .debs build from it, will rebuild it and put the resulting package up for download.

Conclusion: a perfect opportunity to finally give this a try. The learning curve was surprisingly shallow, and the result can be seen in [my personal package archive](#):

[Profile](#)
[Related Software](#)
[Karma](#)
[Personal Package Archive](#)

PPA for Egon Willighagen

This archive is public

URL: <http://ppa.launchpad.net/egonw/ppa/ubuntu>

Description
Patched version of xmlbeans.

[Change details](#)
[Edit dependencies](#)

PPA build status
Total builds: 1
Failed 0
Pending 0
Superseded 0
Succeeded 1

[apt sources.list entries](#)

```
deb http://ppa.launchpad.net/egonw/ppa/ubuntu jaunty main
deb-src http://ppa.launchpad.net/egonw/ppa/ubuntu jaunty main
```

i This repository is signed with 1024R/48A348DD OpenPGP key. [Follow these instructions](#) for installing packages from this PPA.

[Repository disk usage](#)
[View build records](#)
[Copy packages](#)
[Delete packages](#)

- 1 source package (10.7 MiB)
- 1 binary package (2.2 MiB)
- Estimated archive size: 12.9 MiB

Package name contains: Published in Any Series

1 → 1 of 1 result First • Previous • **Next** ► • Last

Source	Published	Status	Series	Section	Build Status
xmlbeans - 2.4.0-0ubuntu3~ppa1 (changesfile)	10 hours ago	Published	jaunty	Libs	

Now, you can easily imagine that I will soon work on packaging stuff I did in the past too, such as update [libcdk-java](#) and now that OpenJDK in main can run [Jmol](#) reasonably, finally package Jmol for main. I just hope I remember my [Alioth](#) account, so that I can properly contribute to the [debichem](#) project.

Getting back to running *scomp* on the CML scheme, it works with one minor problem:

```
$ scomp -src . -d . cml.xsd
/home/egonw/tmp/cml/cml.xsd:10098:9: warning: p-props-correct.2.2: maxOccurs
must be greater than or equal to 1.
Time to build schema type system: 1.792 seconds
Time to generate code: 3.297 seconds
Time to compile code: 9.658 seconds
```

The problem is reflected by line 10098 which goes like:

```
<xsd:sequence minOccurs="0" maxOccurs="0">
```

which can be traced down to line 23 in [schema2/trunk/elements/tableHeaderCell.xsd](#). I filled a [bug report about this](#).