ICE as a blog editor

I noticed yesterday another frustrated aside from Peter Murray Rust embedded in a post about chemistry and e-science:

> [Note: I will continue to try to format the code - Wordpress makes it very difficult]

Also yesterday I wrote about how we are breaking ICE up into more digestible pieces, one of which is the ability to post to a weblog using Atompub. Daniel de Byl has just posted a demo using OpenOffice.org Writer to publish a nicely formatted blog post to WordPress.

And today a supportive reply from PMR to my post with a poem in it! Thing do indeed take time, I've been at this since 1996. I think we're getting there now, though.

I thought I'd try out the new ICE services using one of Peter's posts and see what happens. I think that the ICE toolbar in Writer could help transcend the formatting problems with WordPress and we could look at doing interesting stuff like CML integration.

Here's his post (embedded in mine as a blockquote):

> **CrystalEye: using the harvester**
>
> Jim Downing has written a harvester for CrystalEye. I thought I would have a try and see if I could iterate through all the entries and extract the temperature of the experiment. This is where XML really starts to show its value over legacy formats. Jim's iterator reads each entry and copies it to a file; I decided to read the entry as an XML document, search for the temperature using XQuery and announce it. It's simple enough that I thought I could do it while watching Liverpool (I used to live on Merseyside). Unfortunately (or fortunately) the torrent of goals distracted me so it had to wait till today.
>
> The temperature is described in the IUCr dictionary and held in CML as (example):
>
> 293.0
>
> So this is trivially locatable by XQuery (with local-name() and @dictRef):

```
// iterate through all entries

for (DataEntry de : doc.getDataEnclosures()) {

if (downloaded >= maxHarvest) {

return downloaded;

}

InputStream in = null;

try {

in = get(de.url);

// standard XOM XML parsing, creates a
```

```
Element rootElement = new Builder().build(in).getRootElement();

// standard xquery

Nodes nodes = rootElement.query(

".//*[local-name()='scalar'"+

and @dictRef='iucr:_cell_measurement_temperature']");

// if there is a temperatute extract the value

String temp = (nodes.size() == 0) ? "no temp given" :
nodes.get(0).getValue();

System.out.println("temperature for
"+rootElement.getAttributeValue("id")+": "+temp);

downloaded++;

} catch (Exception e) {

e.printStackTrace();

} finally {

IOUtils.closeQuietly(in);

}

}
```

and here's the output:

```
1625 [main] DEBUG uk.ac.cam.ch.wwmm.crystaleye.client.Harvester
- Getting
http://wwmm.ch.cam.ac.uk/crystaleye/summary/rsc/ob/2007/22/data
/b712503h/b712503hsup1_pob0401m/b712503hsup1_pob0401m.complete.
cml.xml

temperature for rsc_ob_2007_22_b712503hsup1_pob0401m: 115.0

2297 [main] DEBUG uk.ac.cam.ch.wwmm.crystaleye.client.Harvester
- Getting
http://wwmm.ch.cam.ac.uk/crystaleye/summary/rsc/ob/2007/22/data
/b710487a/b710487asup1_ljf130/b710487asup1_ljf130.complete.cml.
xml

temperature for rsc_ob_2007_22_b710487asup1_ljf130: 150.0
```

etc.

It will take the best part of the day to iterate through the entries, but remember that CrystalEye is not a database. We are converting it to RDF (and anyone interested can also do this) when it can be searched in a trivial amount of time and with much more complex questions. (Remember that CrystalEye was not originally designed as a public resource). Until then anyone who wishes to use CrystalEye a lot would do best

> to download the entries and build their own index.
>
> [Note: I will continue to try to format the code - Wordpress makes it very difficult]

Easy enough to do in ICE apart from the slightly clunky way quoting works. We really need the ability to import HTML properly formatted as a blockquote. This would be very important for PMR, as he likes to quote big chunks. In HTML all you do is wrap `<blockquote>` tags around the source for the quote and you're done. In ICE you have to imply the quote by marking the first paragraph as `bq1` style using our easy-to-click toolbar buttons, then indent the subsequent paragraphs appropri ately with the indent button. We'll work on automating that but it's not that bad even now.

I considered using this tip to change my CSS so that stuff in `<pre>` tags wraps but on the test site I would have to pay to change the CSS. The theme I'm using uses this: `pre {overflow: auto;}` . PMR has used `<code>` inside a paragraph. Not sure what the solution would be there but maybe the same CSS would help.

You can see a draft version of this here post on my test blog.

If you want to try this out and blog to WordPress from OpenOffice.org then see the instructions that Daniel has put up on the ICE site and remember this is bleeding-edge alpha-quality Windows-only software at this stage. Remember also to actually read the instructions. The URL to use for WordPress is really important, for example.