Claims about ODF support are typically meaningless

I know I'm repeating myself a bit. But as you know there's a Wikipedia page about applications that support the Open Document Format and it gets quoted and linked to. A lot.

I linked to Peter Murray Rust yesterday, and one of the commenters on his blog also talks about the number of implementations.

> OpenOffice.org is only one of the tools that can generate it, there are several others as well as various converters (e.g. SUN's MS Office plugin, Clever Age ODF translator) available for MS Word users.

But folks, as of 2008-05-13 mid afternoon Toowoomba time, that Wikipedia page is not much help to people who might want to, like you know work on real documents. GoogleDocs, for example will throw away your styles if you happen to care about them. And why would you? It's a web two-dot-oh world now what do we need with styles?

I'm going to get around to editing that page, but I'm not an expert so I have held off.

I just know that it's not useful. Lots of things on that list don't work with **my** ODF documents all created with applications derived from OpenOffice.org.

As part of my homework for when I do engage the page I went to the spec.

What does the ODF spec (v1.1) say about conformance? You need only read the first sentence of this extract, which I have highlighted for your convenience.

> **The OpenDocument specification does not specify which elements and attributes conforming application must, should, or may support.** The intention behind this is to ensure that the OpenDocument specification can be used by as many implementations as possible, even if these applications do not support some or many of the elements and attributes defined in this specification. Viewer applications for instance may not support all editing relates elements and attributes (like change tracking), other application may support only the content related elements and attributes, but none of the style related ones.
>
> Even typical office applications may only support a subset of the elements and attributes defined in this specification. They may for instance not support lists within text boxes or may not support some of the language related element and attributes.

So you don't have to do anything in particular to claim to support ODF. Maybe just allowing the top level element would be enough?

We clearly need to add some detail to the Wikipedia page about who supports what specifically.

My working definition of support for ODF (the text format is actually what I care about) is people using our word processing templates to edit files interoperably with Writer and some other application. And you know what? There's nothing outrageous in any of our templates, but the **only** applications that work are ones that are based on the original Star Office code base that spawned OpenOffice.org.

This is unsurprising. The file format was built around OpenOffice.org. Lots of people point out that nobody but Microsoft will be able to build an office suite that supports OOXML. People, this is the same process at work.

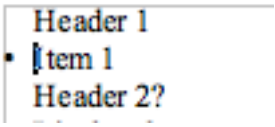
Let me tell you about a quick experiment I did. I looked up the bit in the ODF spec about lists. Apparently you can have a thing called a list-header at the start of the list.

> A list header is a special kind of list item. It contains one or more paragraphs that are displayed before a list. The paragraphs are formatted like list items but they do not have a preceding number or bullet. The list header is represented by the list header element.

So I made a .odt using NeoOffice, put in a list using the default style `List 1` saved it, unzipped it and added a list-header to the start of the list, then rezipped it and opened in NeoOffice. Hmm, well it does display, but the Writer interface seems to know nothing about list headers. The only way to create one seems to be outside of the application. Having a feature like this creates some very serious weirdness. You can load up a document with multiple paragraphs in the list header and they are preserved but if you try to add one then you just get a normal list item.

Now I know this is not quite in line with what I'm saying about the file format being largely derived from OpenOffice.org. I don't know the history of this element but OpenOffice.org doesn't support it in any useful way. Looks like something added by a standards committee raised on SGML with a clear idea of what makes a 'good' document format and not much consideration about what makes a usable word processor interface but that's just a guess.

Lets talk about the formatting I got from my experiment. Is this what the spec really means?



See how the list header (Header 1) is indented? Not formatting I can imagine using. But it seems to be what the spec says. “The paragraphs are formatted like list items but they do not have a preceding number or bullet.”

How many of the ODF cheer squad have read the standard? Dealt with document interop issues? Me, I have only glanced at the OOXML spec and so I don't go around telling people about how bad it is, but on the few occasions I have looked at ODF and tested support for various things in Writer I have found problems or lack of support in OOo. The good thing with OpenOffice.org, of course is that you can tell Sun about the bug and it will likely get fixed sooner or later, particularly if you can rally enough supporters to vote.

The bottom line is that if you want to work with ODT you have to check out whether the other applications are going to support the bit you want to use. I will get around to changing that wikipedia page to be more useful.