

We recently released CogniDox 8.1 and amongst the new features was a CogniDox Word Add-in.

An Add-in is a software component that adds custom commands and new features to a primary program such as those in the Microsoft Office suite. The particular value of the CogniDox Word Add-in is that you can browse and issue commands to the CogniDox repository while you are using Word.

The first elementary fact we faced is that the Office suite is a collection of individual programs and we had to choose which one we wanted to extend. We've blogged recently on the ongoing ubiquity of Microsoft Word as the principal tool for knowledge workers, so it stood out from e.g. Powerpoint and Excel as the one to go for.

Everyone knows there are multiple versions of Office in use across the market. Ideally, we'd support all of them. Unfortunately, the differences between Office 2003 and 2007 makes that two quite separate tasks rather than one. It isn't easy to find out what share of users are using Office 2007 compared to older versions - there's a stat from Forrester that roughly 80% are using the latest version (2007). This seems on the high side, but then when you consider that new PCs will be shipped with the latest version and that 80% of Microsoft's Business Division revenue comes from Office B2B sales, perhaps it isn't so far out. I have to say we're still not fully convinced though.

So we choose Word 2007. Worst case scenario, everyone plans to upgrade to it and our work is re-usable for Office 2010. Having to support 2003 would have eaten into development time for a product that will become less relevant to our user base.

The next issue was how to tool ourselves for the task. MSDN tools have been prohibitively expensive. Luckily, a year ago Microsoft launched a program for new start-ups that would allow them access to MSDN on a very reasonable basis. You need to satisfy certain start-up criteria - in business for less than 3 years with under \$1M in revenue for example, and you have to be nominated by a Network Partner (often a VC or a BA group). But that gets you participation for up to three years and access to Visual Studio, Windows Server, SQL Server, SharePoint, MSDN, Microsoft Office and more.

We joined BizSpark in mid 2009 through the assistance of our local friendly and capable [Business Leaders Network \(BLN\)](#). With the help of their registration code it was really simple to do.

Once we had access to all these development tools we had to decide which ones to use. We went for Visual Studio 2008 using C#. There's a strong argument to do Office development in VisualBasic.NET rather than C#, because a lot of the Office Interoperability API calls use optional parameters. This is not supported by C# 3.0. As a result, you tend to wind up with fairly

verbose code. MS has a [KB article about this](#) . Visual Studio 2010 will bring in support for optional and named parameters, which will make working with the Office Interop APIs much more pleasant. As the Add-in is a relatively small project; we prefer C# much more than VB and VS 2010 will bring improvements, therefore we went C#.

At the protocol level, our interactions with the Add-in are via SOAP. CogniDox already had a SOAP interface used by our Linux command line client, so we were extending known functionality to provide a richer interface for the add-in. The benefit is we can reuse the SOAP API to do other integrations. CogniDox publishes its SOAP API using a Web Services Description Language (WSDL) file, and all it took to expose the API to the C# project was to point Visual Studio at the WSDL file URL.

This highlights the fact that Open Data formats can be as important as Open Source.

At the UI level we had to work with the ribbon UI and a custom task pane which are standard UI elements within Office 2007. One decision was between using the WPF (Windows Presentation Foundation) or WinForms for the UI controls. WPF is relatively new and set to be the future of Windows UI development, but we went the WinForms route because it gave us access to more mature documentation that sped up development time. We may re-visit this decision with future releases of the Add-in. Our priority for the initial release was to explore where we could go with the Add-in and get some real user benefit out in the field.

So, at least one LAMP shop gets to develop for MS Windows. Without the existence of the Microsoft Bizspark program, it's highly unlikely that we would have chosen to do that or be able to fund it.