Boundaryless
Information Flow

Open Standards - Open Source

The Business, Legal, and Technical Challenges Ahead

June 24-25, 2003, Minneapolis, MN, USA

www.opengroup.org

THE Open GROUP

Open Standards and Open Source

a position paper prepared for the technical panel of the Summer 2003 Open Group conference on Open Standards and Open Source.

John Collins, Ph.D.

Assistant Professor and Director of the Master of Science in Software Engineering program, University of Minnesota.
`jcollins@cs.umn.edu`

Abstract

For a wide range of infrastructure software, Open Source solutions are robust, portable, and interoperable, and should be adopted over proprietary solutions in most circumstances. However, the situation for application software is murkier, and will remain that way until there is a robust standards framework for application-level interoperability and until open-source developers make usability a much higher priority.

Background

I have a somewhat different background from most of the open source community. I am currently an academic, involved primarily in teaching and research, and in academic administration of a professional masters program in software engineering. I also have over 30 years of industry experience, most of it in hardware and software product development. I have authored 16 U.S. patents and a number of research publications. I have very little corporate IT experience. My experience with open-source software began in 1978 with a Pascal compiler for the PDP-11 that I obtained from the Digital Equipment User's Society (DECUS). It was not a high-quality product, but I contributed several corrections and improvements. By 1981 we had acquired a VAX and began using BSD Unix.

Position

I believe that much Open Source software is ready for prime-time, but much of it is not very usable by the average consumer or corporate user.

In general, much open-source *infrastructure* software (operating systems, network infrastructure, Web infrastructure) is robust, scalable, portable, and interoperable, and businesses and institutions should now be in a position of explaining why they have *not* adopted it, because it is low-cost and free of encumbrances such as restrictive and sometimes intrusive license agreements.

There are indeed some infrastructure areas where open-source solutions have not achieved a critical level of capability. Examples include system-management tools (think OpenView, Unicenter, Tivoli) and message-oriented middleware.

On the other hand, most open-source *application* software, other than software-development tools, is not sufficiently robust or interoperable to displace proprietary solutions, at least outside the technical community. Interoperability at the application level involves both communication and document interchange, and I believe that in both these areas we lack the non-proprietary standards to enable the desirable level of interoperability.

There are certainly exceptions, such as Mozilla, and the large variety of text-based programmer's tools. The reason Mozilla is competitive is because it (and other open-source browsers) are based on a widely accepted and useful standard. But even Mozilla (on X-Windows, at least) does not support the kind of application-to-application interoperability that exists, and that users expect, on Windows or Macintosh with IE or Mozilla - the ability to drag-and-drop reasonably seamlessly between applications, and to open files by dragging icons onto the apps or their icons. Whatever one may think of WIMP interfaces, they are popular and reasonably comfortable to the vast majority of the user base. But most open source applications, at least in the standard X-Windows environment, do not support that level of interoperability, and there are many small, unfortunate bugs in the WIMP interoperability layer of many of these applications, such as starting up multiple copies unnecessarily.

There are 4 different window systems currently in general use, X-Windows, Microsoft Windows, and two versions of Macintosh windows. X-Windows is an older and lower-level standard, defined at a time when text-based applications were state-of-the-art. That is why text-based programmer's tools interoperate well in this environment. Higher-level interoperability, such as the ability to copy formatted text, structured data, or images of various types among unrelated applications, is mostly absent. ICCCM just isn't enough. The floor is littered with attempts to solve this problem, such as Motif and CDE, but none of them has had the scope and reach of the closed proprietary environments for obvious reasons. There are many competing schemes to support application-level interoperability, but none of them has caught on in the open-source community as a whole, and retrofitting the existing application base to a new standard is something the open-source community is unlikely to tacke unless there is broad agreement first. So we have OpenOffice, KDE, Gnome and other incompatible interoperability schemes. The same is true of document formats. The Mono project offers to enable Microsoft interoperability in Linux and Unix environments, but it's unlikely that existing applications will be upgraded to support it.

Interoperability through standard document formats is probably a more important issue in the long run than is run-time interoperability. For organizations, the ability to share documents is critical to basic business processes. Expensive upheavals have often occured when one person in an organization upgrades Word and begins producing documents that co-workers can't handle. Elaborate and expensive upgrade planning is seen as a solution, and extreme dependency on software vendors is not viewed with sufficient alarm. In the public sector, transparency demands that documents be prepared and stored in formats that are publicly documented so that public information is not held hostage by private interests. This issue is causing a number of public bodies to pass open-document-format legislation. This will help, but document formats can be very complex, and documents can embed dependencies on multiple applications. A standard would be better. Many have been proposed and implemented, such as RTF, ODA, SGML, DocBook (see myfileformats.com for a long list). But do we have a usable standard? Perhaps there is one out there, but the lack of adoption argues that we do not. Perhaps all that is needed is leadership?

If we are to have a full open-source desktop environment, interoperability and usability issues must be addressed. The community might be able to address usability issues, but this may be an area where commercial interests have an advantage. Interoperability will require new institutional commitments that can engender broad support in the developer community. Possibly the Desktop Linux Consortium can be that institution, but others such as OMG, W3C, and the Open Group also have roles to play, especially in the area of open formats for documents and structured data. Much work remains to be done, and education and leadership will be critical factors.