

Experiences with Windows CE and Linux

Tim Chart, Nicholas Race, George Metaxas and Andrew Scott
*Computing Department, Lancaster University,
Lancaster, UK*

In the Internet Protocol Version 6 (IPv6) Research Group, based in the Computing Department of Lancaster University, U.K., we are keen to select as our research platform the product that is most closely aligned to our needs. The group has successfully used both the Linux and Windows desktop platforms in the past, and continues to embrace these platforms and others as the basis for research. But Windows CE has been adopted by an increasing proportion of the research group as our primary research platform. There are many ways in which both Linux and Windows CE operating systems excel, but neither is perfect. In this article, we attempt to explain how the strengths and weaknesses of both platforms have influenced our researchers.

The IPv6 Research Group has a strong research background in mobile systems. In order to consolidate our position, we aimed to move away from desktop-based implementations and towards embedded and mobile systems. Applications that showcase the array of lower-level technologies in which we have expertise are far more impressive if we can demonstrate them on mobile devices. In our case, the nature of the technologies demonstrated by our applications make it necessary for the platform to support reduced hardware architectures and the latest wireless technologies. The availability and compatibility of hardware, along with support, both software and human, remains a very important factor in our choice.

At this point, we should set the parameters for this discussion. We are primarily interested in mobile and embedded systems in our research. It is, unusual to see Linux outside of the domain of desktop and server machines. However, Linux can be installed on embedded systems providing the hardware architecture is open to the development community (for example, a headless PC). It is far more unusual to find Linux on a mobile device such as a PDA because such hardware is often designed specifically for its operating system, and the creation of a new hardware device requires a certain amount of commercial incentive. It is important to stress that in this article we are comparing Windows CE with mobile and embedded incarnations of Linux. It is fair to say that such flavours of Linux do not share all of the advantages of desktop varieties.

One of the greatest weaknesses of Windows CE is that it supports only a restricted set of hardware designs. While the list of compatible hardware is presently growing, previous iterations of the operating system required us to select our hardware from a short list.

Any devices and peripherals not supported out of the box are supported only by the manufacturer. Being at the mercy of manufacturers for appropriate drivers for a specific version of the operating system meant that a new driver often had to be sought at the next major upgrade, and we frequently found older hardware was no longer being supported. We feel that for open-architecture embedded devices, Linux would have excelled in this situation. The large developer community would undoubtedly have produced drivers to support the few devices and peripherals that were not supported out of the box.

However, most commercial Windows CE devices are based on hardware architectures designed specifically for Windows CE and some of the worst-supported commercial devices suffer from the lack of drivers. In contrast, very few mobile devices have been designed specifically for Linux. Furthermore, the open-source community is unable to support drivers for most mobile devices due to their closed architectures. A version of Linux has been made available for the previous generation of Compaq iPAQ. However, this was only possible because Compaq co-operated in producing a hardware abstraction layer for Linux. Where mobile devices are concerned, Linux users are just as much at the mercy of manufacturers for support as Windows CE users.

For open architectures, it is obvious that the availability of open-source drivers gives a clear advantage to Linux driver developers. The extensive library of such drivers provides a sound basis from which driver developers can create customised drivers. Windows CE provides a number of reference drivers, which are sufficiently general purpose to cover a wide range of hardware, however drivers for specific hardware are generally closed-source, making it difficult to implement simple modifications. For example, using Windows CE, we wanted to alter the behaviour of a driver for a popular brand of 802.11 card. However, the absence of a standard interface in the version of Windows CE that we were using and the lack of source code for the driver prevented us from discovering the proprietary interface we required.

In the past, both Linux and Windows CE have had reputations as being behind the cutting edge with regard to support for recent technology advances. In the case of Linux, hardware vendors would release Windows drivers with their hardware, but Linux support would inevitably be delayed several months until either the manufacturer or a third party produced

drivers. Similarly, Windows CE was frequently overlooked in favour of supporting its more famous desktop siblings. In both cases, the situation is rapidly improving. Manufacturers and vendors of off-the-shelf components now take Linux seriously and develop Linux and Windows drivers in parallel in the vast majority of cases, and the latest iterations of Windows CE support many exciting new technologies.

Upon brief inspection, Linux appears to have the edge in supporting new technologies, on the basis that distributions of Linux are released more frequently than versions of Windows CE. However, taking into account the versions offered as part of the Windows CE beta programmes, both Linux and Windows CE are fairly even. An additional complication where mobile devices are concerned is the lead time required for companies to create real off-the-shelf devices from reference versions of the operating systems. At the time of writing, the latest version of Windows CE running on a commercial device with the Pocket PC form factor is 3.0, yet the current version of Windows CE is 4.2. However, there are more up-to-date devices available in other form factors. This is perhaps less of a problem for Linux users, because the technology and knowledge exists in the open-source community to flash existing open-architecture devices with up-to-date versions of the Linux kernel.

While a self-install Linux for existing Pocket PC hardware takes away the dependence on a single company for updates, we were deterred by the fact that the lack of a clear development leader might leave us without information about the availability of updates and the intent to develop specific drivers and applications. In addition, the number of commercial devices built on Windows CE far outnumbers those based upon Linux, thus opting for Windows CE allowed us to spread the risk of supporting the operating system. If one hardware manufacturer stopped supporting Windows CE, the operating system would still be useful, as there would be other manufacturers providing support.

With source code being freely available for Linux, and recent moves by Microsoft to open up the source for Windows CE to universities, we have never had greater freedom to work using both platforms. At Lancaster, we have a number of projects and research ideas that would only be possible if we modify the research platform itself to extend operating system capabilities. In this respect, we feel that Windows CE makes for the superior platform.

The source code available for the Linux kernel and packages is, as a rule, difficult to pick up and edit. It is fairly poorly commented, if at all, and editing the kernel level source code is complicated by having to learn a different API within the familiar user space calls. That said, the comments in the source code are complemented by HOWTO documents and similar

guides and books, whereas there are few such guides available for Windows CE. In the past, however, we have found that Linux HOWTOs tend to refer to code that is many versions out of date.

The size of the Linux code base is also daunting. By contrast, the Windows CE source base is small. Despite growing at an impressive rate from release to release, on the whole it is very well commented and organised so it is possible to quickly find the code in which we have an interest without possessing considerable knowledge of the structure of the code. The deficiencies of Windows CE in hardware support work as an advantage, because the result is to limit the size of the source base. System-level code in Windows CE can use the same API as application code, so the learning curve for modifying the source is shallow.

At Lancaster, we consider ourselves a pioneer in adopting new technologies to teach our undergraduate students. We were early adopters of Java, C# and Microsoft .NET. All of our undergraduates are presently taught to use Microsoft APIs and environments. Since the majority of new postgraduate researchers are recruited from our undergraduate programme, we feel that it is a considerable advantage that our undergraduates can migrate to using Windows CE with relative ease from the Win32 API. As an addendum to this argument, those of our researchers that come from a Linux background were mildly disadvantaged by this.

Regarding application programming resources, we feel that both Linux and Windows CE are well featured and supported. The open-source culture of Linux developers creates a superior resource of diverse code examples, while the relatively immature Windows CE development community is less diverse. However, both platforms suffer the problem that developers tend to allow code to become obsolete and cannot be relied upon to continue to support and maintain legacy code. Linux in particular suffers from legacy and unsupported code as a consequence of the maturity of its development community. Yet we have had similar problems obtaining support for Windows CE code.

An advantage of being part of the university shared source programme for Windows CE is that it gives us access to a central point of contact for any problems we might have. However, we have found in the past that the programme can hinder our activities as we were unable to share our code with other interested universities. To some extent, we are no longer in this situation, and although we are unable to have an open-door policy for the code for some of our projects, we feel that this may even benefit the development community in the long run by encouraging active participation in the shared source programme.

It is difficult for us to compare the features and libraries available to our Linux and Windows CE

developers because each platform has its own to offer. For example, Windows CE has more mature support for .NET applications, which allows us to create more portable research implementations, and more importantly, rapid prototypes of our technology. This support is provided by Windows CE in the form of the .NET Compact Framework, which still impresses us with the extent to which it manages to hide the usual concerns and limitations of developing for embedded and mobile systems. However, Linux has a more mature security model and provides native sandboxing of processes, which is difficult to achieve using Windows CE.

It is true that Linux has more features and libraries than Windows CE, due to its open source library of software. This wealth of features often reduces the development time for some types of application. However, no amount of libraries and features can make up for an operating system that wasn't designed to operate in systems with constrained resources. Most of the open source back catalogue makes no special effort to control memory usage and simply assumes vast quantities of virtual memory. Windows CE was designed with constraints in mind. It provides a number of features that make trivial the task of creating lightweight and small footprint implementations.

The feature of Windows CE development that initially attracted our attention, and one which encourages us to continue using Windows CE as our primary research platform, is the available tool set. Windows CE applications are developed using eMbedded Visual C and Platform Builder. The latter is capable of building a full range of applications, drivers and core components. Both integrated device environments (IDEs) are similar, and integrate invaluable editing, building, and debugging tools that reduce the time we spend developing. The particular strengths of the IDEs are the debugging tools. The interface is intuitive and powerful. In addition to these tools, Visual Studio .NET can be used to develop applications for the .NET platform. We are also encouraged by Microsoft's plans to release a version of Visual Studio that unifies Windows desktop and embedded development under a single IDE. The fact that there is a standard environment supported by the operating system product group means that the potential of the IDE is maximised. By contrast, there is no standard IDE for Linux development, and certainly none amongst the unofficial IDEs that is as powerful as Platform Builder. Instead, we find that we tend to rely upon GNU Compiler Collection alone for Linux development and the other tools available, such as GNU Project Debugger, tend to go unused.

However, we have found that there is a disadvantage to having a single development IDE, and that is that it becomes the single point of failure. On more than one occasion when using older versions of Platform Builder, we have found bugs that prevented us from

making progress with our work. True, some of these may have been due to our participation in the beta programme, but if alternative tools had been available, we would have been able to proceed. The diversity of tools available for Linux means that we would not have been held up by bugs in one.

To summarise, we have and continue to use both Linux and Windows CE as research platforms. However, most of our researchers choose to use Windows CE. Although Windows CE is deficient in hardware and driver support, its tool support is exemplary and it excels in many other areas. Both Linux and Windows CE development communities suffer similar problems, but offer dissimilar advantages. Linux has the edge on the diversity and availability of source code, yet Windows CE has a far better organised and commented shared source package available. In fairness to Linux advocates everywhere, Linux was never built as an operating system to support mobile devices with severely constrained resources. We feel that the fact that Windows CE was built from the ground up for this specific purpose, Linux cannot compete with Windows CE in the context of mobile devices, and perhaps greater threats come from other competitors in the same market.