

SELECTING AN EMBEDDED RTOS



FEATURED INTERVIEW:

EXCERPTED FROM WWW.EG3.COM

Prepared by:

eg3.com

Jason McDonald, Senior Editor

eg3.com

tel : 510.713.2150

email : info@eg3.com

web : <http://www.eg3.com>



ECOSCENTRIC: THE ADVANTAGES OF USING A "FREE" RTOS

15 December 2008: The Advantages of Using a "Free" RTOS

INTERVIEWEE. PAUL BESKEEN
 CHAIRMAN & DIRECTOR OF BUSINESS DEVELOPMENT
 TEL. +44 1223 245 571
 EMAIL. info@ecoscentric.com
 COMPANY. ECOSCENTRIC
 WEB. <http://www.ecoscentric.com/>

- Q. First of all, tell us a little bit about yourself and your responsibilities at eCosCentric.**
- A. My career spans over 20 years in the engineering and technical management of embedded operating systems and related software. I started at Perihelion Software, a company specializing in parallel and embedded operating systems. The ten year stint at Perihelion culminated with my appointment as Technical Director (CTO) of the company. I subsequently worked for Virtuality Inc, a developer of virtual reality hardware and software, as manager of its Operating Systems group. In 1996 I joined Cygnus Solutions as Director of Engineering, tasked with building and heading up a team to create an open source operating system designed specifically for the deeply embedded space – the eCos RTOS. I continued in this role after the merger of Cygnus Solutions with Red Hat. I am a founder and the Chairman of eCosCentric – a company spun out of Red Hat to focus on the eCos RTOS and associated technologies.
- Q. Elsewhere in the guide, we have a product Q&A interview on eCosCentric, but for the benefit of our readers, would you please give us just a short description of what the company provides in the RTOS market?**
- eCosCentric's main product is the eCosPro RTOS, which is coupled with a complementary range of services including porting and custom engineering, training and support. eCosPro is a distribution of the eCos open source RTOS that integrates all the host tools and runtime packages necessary for general embedded software development. See <http://www.ecoscentric.com/ecos/ecospro.shtml> for more information.
- Q. Our purpose in this keynote interview is to discuss the pro's and con's of using a "free" RTOS. So first of all, can you explaining the licensing model of eCos and eCosPro and how it is different from that of a commercial RTOS?**
- A. Compared to proprietary licensing models, open source licenses in general provide more freedom. Users of the eCos RTOS benefit from the freedom to customize the RTOS, and freedom from various license fee requirements.
- Source code for all eCos runtime packages is included in all releases of eCosPro. Engineers have complete freedom to inspect, instrument, modify and customize the code as they see fit.
- None of the eCosPro licenses require any royalties or product fees to be paid. You may develop as many designs, and ship as many units as you wish, with no requirement for external audit, unit reporting, or additional payments.

The eCos runtime sources are covered by two main open source licenses. The core eCos code is provided under the GNU Public License (GPL) coupled with an exception clause. The exception clause modifies the GPL into a non-viral variant, enabling other software to be linked with eCos without that software itself needing to be licensed under the GPL. This provides the flexibility to mix and match open source and proprietary code in your product without any cross contamination. The eCos network stacks are covered by the permissive BSD open source license. eCosCentric's eCosPro releases add additional features that are provided under a permissive commercial license. This license simply restricts the ability to distribute the eCosPro specific licensed source code to third parties. On the host tools side the GNU compiler tools are licensed under the GPL, and the Eclipse IDE under the Eclipse Public License (EPL).

Q. What is the difference between *eCos* and *eCosPro*?

- A. The eCosPro distribution is designed to provide a high quality platform suitable for commercial eCos-based product development. It delivers key improvements over the stock eCos in three key areas: functionality, testing, and support.

The eCosPro distribution integrates all the disparate host tools and runtime elements required to develop eCos-based embedded applications. Both Windows and Linux are supported, and host specific installers and tool binaries are provided. This simplifies and speeds the process of getting started, and as the distribution been tested as a cohesive whole, ensures that all the tools and runtime code are fully compatible.

eCosPro adds a number of host based tools including an Eclipse-based IDE, memory allocation debugger, and code coverage & profiling tools. Runtime functionality is extended with a range of new features including the standard C++ library, object code loader, small footprint networking stack, MMC/SD/SDHC driver, and in-field upgrade packages. Many existing packages have been enhanced, for example the FAT filing system now has FAT-32, long filename, removable media, and internationalization support. Similar functional improvements have been made to other standard eCos packages.

An extensive range of eCosPro-exclusive CPU, target platform, and device driver ports are also available. Device support varies by platform but typically includes a mix of serial, Ethernet, flash, RTC, watchdog, I2C and SPI drivers. Availability of a port and associated device drivers for your target hardware can help to significantly reduce product development schedules.

All releases are rigorously tested and quality assured to help ensure that eCosPro provides a stable and reliable base for your application. A comprehensive automated test infrastructure typically executes over 21,000 runtime tests on each target platform. In addition to this validation of compiler tools and runtime code, strict QA procedures ensure that the installation and functionality of the host tools are up-to-par. This approach is in stark contrast to the unpredictable quality and robustness of downloading from the public eCos repository on any given day.

All eCosPro releases are backed-up by eCosCentric's support services. These cover both the provision of advice relating to the use of eCosPro and associated tools, and resolution of any issues you might encounter with the RTOS. In the rare event a bug is found in eCosPro a patch or workaround will be developed to resolve the problem. Support services are provided within a guaranteed response period, ensuring that you are never left on your own. This commercial grade service is very different from the type of response that you might get when posting a query on an internet mailing list. In that situation there are no guarantees that you will get a response, or that a response will be timely or accurate. It is even rarer for some to provide a bug-fix tailored to your specific issue. No one is under any obligation to help you.

Q. Recently, we did an podcast interview with John Carbone of Express Logic, and he pointed out that once you end up paying fees to companies for “free” RTOSes, one of their big advantages over commercial RTOSes goes away. You might pay “nothing” for the RTOS, he said, but lots for support or lots for the right to use it in a commercial project, and so the big comparisons become between the costs of using one vs. the costs of others, as well as the technical differences. Do you agree that in some ways the whole “buzz” around “free” is a little misleading?

A. There are of course two different meanings of the word “free”; one is in terms of cost, the other in terms of freedom. Given the sustained growth over the last decade in the use of commercial open source offerings, it is clear that engineering and management see benefits from both aspects of “free”.

As an organization or project team you may be able to support yourself completely from a public source base, but that may not be a valuable or efficient use of your limited resources. This is where commercial open source vendors come into play. These vendors can provide a stable development base that can provide the level of bsp, driver and other functionality that you require, without the investment in the staff, time and cost necessary to develop and maintain it yourself. Using an organization that specializes in development and support for your chosen open source system can be a very cost effective and more deterministic way of managing your core operating system requirements and ongoing technical support needs. Here’s the rub, basing your development on an open source operating system provides you with *flexibility*. If for whatever reason you are unhappy with your supplier then you have access to a wide selection of alternative contractors, consultants and companies that can support you. Alternatively after an initial development phase you may feel more confident that your engineering team can now support themselves sufficiently, and scale back the level of development and/or support from an external supplier. You are not locked in to a single-supplier proprietary system.

Proprietary offerings generally require payment of a per-product license and/or royalties. This is not the case for open source based projects. Product license costs may also not be a one-off cost, but can be recurring. Take the case of deriving a product from an existing one. Although the engineering effort may be minimal the proprietary supplier may well require you to pay for another product license. The absence of initial and ongoing license costs for open source based products provides significant cost savings and simplifies ongoing development. Trying to determine what your proprietary vendor may or may not consider a derived product, and other licensing small print, can be a minefield in itself.

The key underlying driver of the cost effectiveness of commercial open source solutions compared to proprietary ones, is that you are not only benefiting from the investment in the source base from a single supplier, but also from the wider open source community as well.

Q. The most famous “free” RTOS is of course Linux. It’s gotten all the more confusing as companies like LynuxWorks, Wind River, and MontaVista have all offered “commercial” versions of Linux. Help us out here - how does the eCosPro offering differ from an “embedded Linux” solution?

A. In many ways eCos and Linux are complementary as they provide solutions to different segments of the embedded space. eCos was specifically designed as an open source RTOS for the deeply embedded market. Linux in contrast was developed as a general purpose open source operating system that has successfully been used in higher end embedded devices. Linux memory requirements are typically measured in megabytes. In comparison eCos is an extremely scalable embedded operating system that, depending on

the required functionality, scales from a few kilobytes to a few hundred. The memory budget for an eCos based system is therefore very much lower than any Linux system, enabling lower unit costs to be achieved. Indeed, its footprint is capable of fitting within the confines of modern SoC designs that have only limited amounts of on-chip memory available. As product retail prices are typically three times that of the BOM cost, this can be critical for price sensitive, high volume designs.

Much of eCos's scalability is due to its configurable nature. The configuration technology built-in to eCos enables developers to specify the required functionality and characteristics of the operating system. This effectively creates an application specific version that is ideally suited to the requirements of a particular device. eCos is therefore suitable for a wide range of device types, from designs that might have traditionally employed a "roll your own" OS approach, through to the more sophisticated real-time applications that require features such as networking, file systems and so forth.

In comparison with Linux, eCos is a far less complex system, enabling individual developers to more easily comprehend its implementation, and to port, debug and customize it to meet their requirements. eCos implements a classic real-time multi-threaded architecture with priority based scheduling. This is a very efficient execution model that delivers deterministic response times, minimal interrupt latencies, and low overhead context switches. Linux by contrast employs a more sophisticated and heavyweight approach, but does provide virtual memory, paging and multiple processes with memory protection. For deeply embedded real-time designs the former rather than the latter functionality is usually a more suitable choice. Linux however excels where it can leverage its more extensive feature set in devices such as cell phones and PDAs.

In conclusion, both systems benefit from their open source development approach. For developers they provide the ultimate in flexibility and freedom, whilst from a commercial perspective they are attractive due to the lack of royalty and license fees. Both are backed up by commercial distributions and support services. Where they differ is in the embedded market segments that they service. eCos is more suited to embedded devices that require instant startup, real-time performance, and have more challenging RAM/ROM budgets and processor performance targets. Linux is more suitable for higher-end devices that are not too restricted in machine resources, and that require more sophisticated functionality.

Q. One of the complexities of Linux is the GPL license and how it might "require" a company to open source any changes made to Linux. Is this the same with eCos / eCosPro?

A. The core eCos source is licensed under the GPL, and therefore any modifications to that code are covered by the same requirements as for Linux.

Use of the GPL license is a benefit to eCos as it is the most well known and understood; however, as noted earlier, the GPL used by eCos is also covered by an exception clause. The clause removes the "viral" requirement that code linked with the GPL code must itself come under the GPL. This is necessary given that eCos applications are directly linked to the eCos RTOS, rather than being loaded dynamically as separate processes.

The exception clause simplifies the deployment of eCos products in that it gives you the freedom to link eCos with code covered by any type of license, and does not require that code to be made public. Your application and any middleware are NOT required to be made public. eCos packages covered by the BSD license, such as the networking stacks, are likewise NOT required to be made public. Furthermore, if you have made additions to the RTOS that are not derived from GPL code, then these additions do NOT have to be made public.

Q. Among the “free” RTOS another one is “freertos” of Freertos.org. It seems very similar to the eCos / eCosPro model in that there is a “free” non-commercial version and a “commercial” version for which you pay a licensing fee. How are you similar and how different from FreeRtos?

A. A similar approach is taken by eCosCentric. eCos is freely downloadable from the open source project, or you can elect to buy a commercial version - “eCosPro” from eCosCentric. Either can be used to develop commercial products with, but as described above eCosPro also provides additional tools, runtime features and ports, is extensively tested, and is backed-up by a comprehensive technical support service.

The two offerings differ far more at the technical level. FreeRTOS is a basic runtime kernel that only provides a limited set of features: binary semaphores and mutexes, queue handling, and task management. eCosPro can be configured down to similar level of functionality, if memory footprint is critical, but in contrast is also scalable to much greater levels of functionality. If a project needs to be expanded to cover more features such as device support, file systems, and so forth, then eCos can readily accommodate this. Time invested in familiarizing yourself with eCos can also be applied to subsequent, more complex, embedded applications.

Q. What are the advantages of going with eCos/eCosPro vs. “doing it yourself?” As you know many developers even today build their own RTOSes vs. buying one or even using a GPL solution.

A. eCos and eCosPro are a great alternative to rolling your own. Many of the issues that lead engineers to build their own OS are already answered by eCos, and likewise many of the pitfalls and drawbacks of a roll your own system are avoided by basing your system on eCos.

The configurability of eCos enables you to rapidly create an RTOS that fits your applications requirements in terms of functionality, performance and footprint. The configuration system effectively creates an application specific RTOS. In its simplest configurations eCos requires minimal system resources, yet it has the ability to be easily reconfigured to encompass additional functionality such as file systems and networking. Extending roll your own systems, when a projects initial requirements expand to incorporate more complex features, can be extremely difficult.

If there are specific features or requirements that are not covered by the available configuration permutations, then you have the ability to modify, customize and extend the source code to meet your needs. This can be far simpler than creating an entire system from scratch. Embedded industry commentator Jack Ganssle has commented that eCos is “the very model of how we must write code – beautiful, well documented, clear, concise, and extensible”.

The use of eCos means that you are starting with a mature system that has a proven track record. There is no need to go through the growing pains associated with creating and debugging a roll your own system.

Given the open source licensing and source code availability you have the rights and potential to be completely self sufficient and standalone. This includes independence from vendor auditing and reporting requirements, as well as license & royalty fees.

When you use eCos as the basis of your project you are not on your own. The eCos community and vendors such as eCosCentric are available to help with technical support, engineering services, training, and compatible middleware. Rolling your own can mean that vital knowledge of system internals is held by one or just a handful of engineers. Maintaining or extending a roll your own system can then become problematic over time due to personnel changes.

Why re-invent the wheel when eCos is mature, capable and malleable enough to suit most embedded system technical requirements, and is licensed in a way that makes it cost-effective and flexible from a commercial perspective?

Q. Thank you for this keynote interview.