Driving Adoption of Open Source Automation Technologies

Philip Wernersbach
Lead Pack4Linux Developer/Maintainer

philip.wernersbach@gmail.com

Glen Wernersbach
Pack4Linux Implementer

glen@jetsoftdev.com

Abstract

While the technological features of a real time system are key, it means nothing unless users can easily deploy and use the system. In this article, we explained the challenges of deploying an RTAI system, including patching the kernel, installing the user space libraries, and tuning the system to be fast and secure. We explained how Pack4Linux addresses all of these problems and packages a ready-to-deploy RTAI system in a Live CD. We also showed how Pack4Linux is used by Jacobs Automation in their PackTrak systems to ensure mission critical packaging lines.

1 Introduction

The choices a user has when deploying a real time application are dizzying. We have soft real time in the vanilla linux kernel, RTLinux, RTAI, Xenomai, Adeos, and those are just a few. Unfortunately, none of these technologies are easy to set up, much less user friendly.

1.1 Websites

A quick look at each project's main page backs up that assertion.

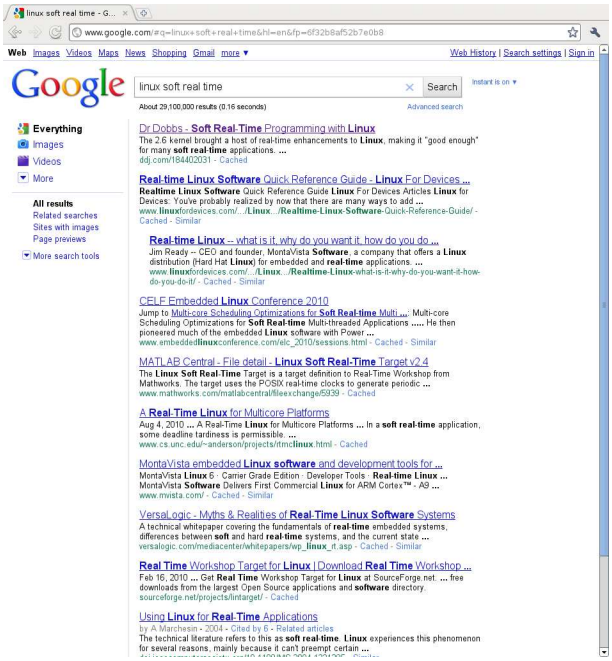

FIGURE 1: *A search for "linux soft real time."*

The soft real time in the Linux kernel has no official homepage and absolutely no official documentation except for some documentation in the kernel sources. A search for "linux soft real time" yields only one relevant link, and even that link contains nothing on actual implementation using Linux's soft real time.

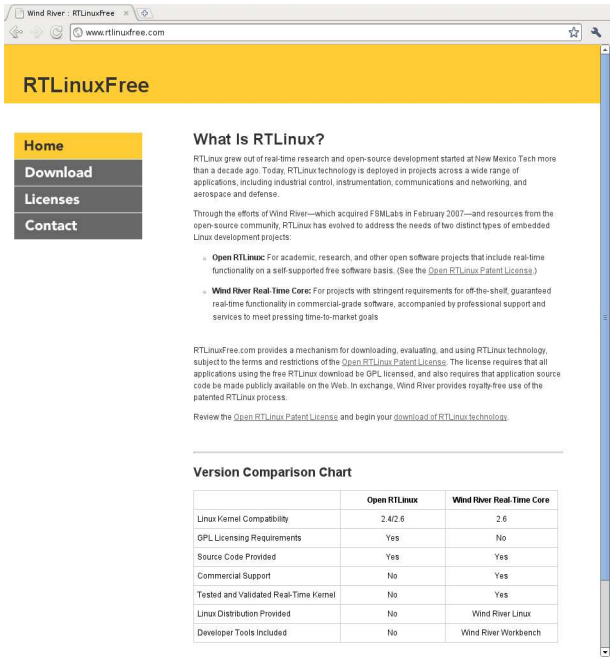

FIGURE 2: *RTLinux main page.*

The RTLinux main page contains a brief comparison of "Open RTLinux" to "Wind River Real-Time Core." Clicking on "Download RTLinuxFree" at the bottom of the comparison prompts you to enter personal information, including your first and last name, job, phone number, and address.

FIGURE 3: *RTAI main page.*

Depending on your browser, visiting the RTAI main page may cause a big security warning to come up, as the site's certificate is self-signed. Assuming this doesn't scare away the average user, once they click through the warning and get to the main page they're greeted with big globs of text. Even though there is a download link in the announcements, it is hard to pick out from the rest of the text.

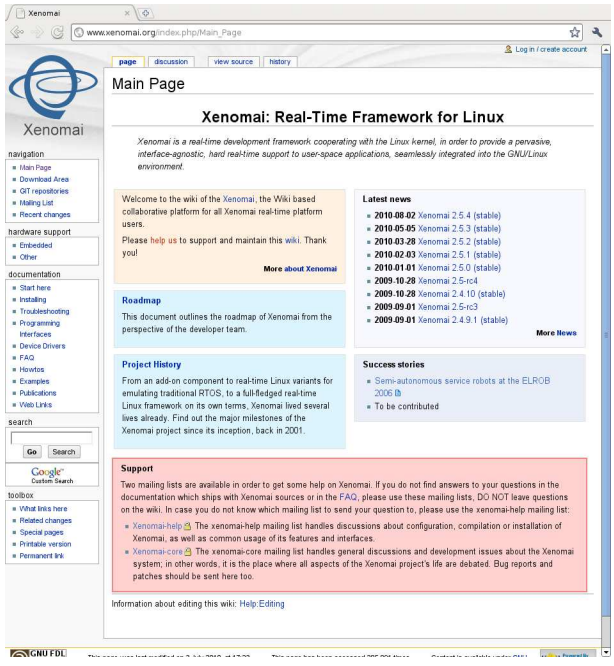

FIGURE 4: *Xenomai main page.*

The Xenomai main page is a little better, the links to download information for the various versions are clearly displayed on the main page, although you still have to click several links to get to a document on the actual installation of the software.

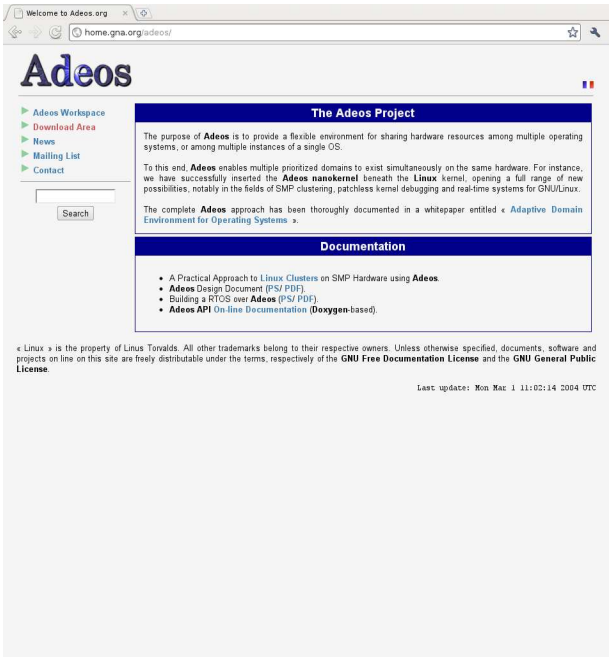

FIGURE 5: *Adeos main page.*

The Adeos main page gives you a very small description of the software, but the download link dumps you into an Apache style file listing. Normal users will get lost in the file labyrinth, and you can forget about finding documentation on installing or using Adeos.

As you can see, these are not unique situations. Out of these five open source automation technologies, only Xenomai's website was easy enough for the average programmer without Linux experience to download, install and use the software. This is simply a no-go. When dealing with adoption of open source automation technologies, each project must have a clean website, with easily accessible downloads, and enough documentation to get the average programmer to a basic sample program without reading the API documents.

2 Setting Up RTAI

For this paper, we will use RTAI. It is free, doesn't have patent or licensing restrictions, and is very, very fast.

2.1 Patching the Linux Kernel

In order to use RTAI, you will first have to patch the kernel. To begin, go to the RTAI website and download the source tarball, which at this time is rtai-3.8.tar.bz2. Next, extract the tarball somewhere you will remember. Under the base/arch folder you will folders for different architectures. Go into the correct folder and apply the correct patch to your kernel. Now compile your kernel and update your bootloader. Reboot.

2.2 Installing the User Space Libraries

RTAI relies on some user space libraries to function correctly. To install these, go back into the RTAI source folder and do the standard "./configure; make; make install" build process.

2.3 Tune the System

In order to achieve a suitable latency, you will need to disable all of the services you don't need. How to do this is beyond the scope of this paper, as it varies by distribution. Next you will need to secure all of the remaining system services to ensure that you do not have any security holes.

Tuning the system is the fun part, because even after you disable services and secure the system, you may find that your distribution automatically starts some processor hungry services that you can't disable, or does some automatic runtime configuration that leaves your system in an insecure state.

2.4 Run the Tests and Code

If everything was successful, which it probably wasn't, you should be able to run the tests RTAI installs. Common problems include incompatible kernel options being enabled, which leads to random crashes and high latency, services which interfere with RTAI being automatically started, or runtime configurations which are incompatible with RTAI.

Whichever way you look at it, all of these steps are a pain to go through, and they waste time. Users like to be able to download something which does these steps for them, and leaves them with a working RTAI system that they can use in a short period of time.

3 Pack4Linux

Pack4Linux is an enterprise ready Linux distribution geared towards real time applications. It is based on Slackware Linux, and aims to be powerful and robust, yet simple to use. It eliminates all of the previous steps and substitutes three: download the CD image, burn it, then reboot and use it.

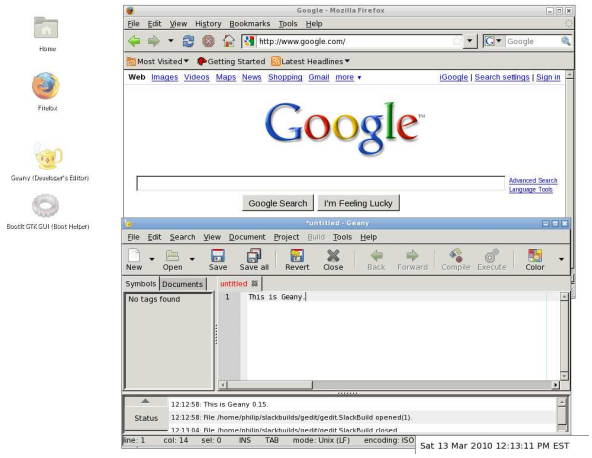

FIGURE 6: *Pack4Linux running Firefox and Geany (the developer's editor).*

3.1 Technology

Pack4Linux uses the rock-solid and stable Linux kernel 2.6.24, prepatched with RTAI. It also includes the RTAI user space libraries, and a minimal Xorg installation and Openbox desktop environment. It uses the GCC toolchain, Geany programmer's IDE, and Mozilla Firefox web browser. In other words, everything that one needs to make and run powerful RTAI programs. Pack4linux also has a "secure by default" mentality, and the default installation is rock-solid. It is a Live CD, which means it can be run off of a CD on any PC without modifying the PC. It can also be installed to a hard disk.

3.2 End-User Implications

Pack4Linux bridges the entry gap into open source automation for new users. The installation process is straight forward and well documented, as is its usage. All of the documentation can be found on the Pack4Linux Wiki, and key documents are linked to on the main page. Pack4Linux is also useful for users experienced with RTAI, as it allows them to focus on programming instead of the other hassles involved with RTAI.

Open source automation technologies need to start appealing to the end-user, the Average Joe programmer with little Linux experience, who is interested in Linux and RTAI but just wants to Get Things Done and not have to worry about logistics. Pack4Linux is a step towards this direction. In fact, it is such a step that Jacobs Automation has decided to use Pack4Linux in their mission critical PackTrak systems.

4 PackTrak

A success story for RTAI:

Jacobs Automation had developed a specialized linear motor system that replaced the chain in assembly lines. The challenge they ran into was how to connect it via software to the outside world so it could be controlled like any other piece of factory automation equipment. While they were qualified in these areas, they really did not want to be driver, real time, or operating system programming experts because these were too far away from their core skills of building automation machinery. Getting their product to work under a real time connectable platform was just something necessary they were going to have to do to sell it.

It came to their attention that a major German industrial company had developed a hardware flavor of real time Ethernet suitable for controlling factory automation. One implementation of the company's chipset was on standard PCI and PCI104 boards for Intel PC's. Their example drivers used RTAI and contained a set of instructions. After reviewing their options for porting this source code, Jacobs Automation realized it would be too big of a project and too expensive per seat to convert the drivers to a commercial real time platform. This is probably why the manufacturer was using RTAI in the first place.

The instructions that came with the real time Ethernet card gave a basic step by step method for installing Linux, Adeos, RTAI, and the hard real time Ethernet driver. Unfortunately, the instructions were for an archaic version of Linux and RTAI. It was hard to find the right source packages to follow the installation instructions as-is. Even when Jacobs Automation had the right packages, they had to start from scratch five to ten times before they got a working operating system that would run RTAI in hard real time with little latency. One frustrating portion of this time was that the documentation for configuring Linux and RTAI were spread over the Internet, and it was very hard to determine which instructions applied to their particular distribution of Linux. The

process of getting their first stable RTAI system took about two weeks, which was exactly the type of time commitment they were trying to avoid. However, once Jacobs had the system setup, they found that it was very easy and straightforward to program the communication gateway between their unit and the outside world using RTAI. It took Jacobs less time to program their application than it did to setup RTAI.

Once they were ready to test out the system, they needed to setup an additional real time Ethernet system to test their communication gateway. The problem was the hardware settings of this new system were different, so they had many of the same problems getting this second system work successfully in hard real time as they did the first. After finishing the debugging process of their application, they realized that they needed to be able to distribute it on a Compact Flash disk. After experimenting with installing Compact Flash for a while, they realized that the flavor of Linux that the vendor recommend was not portable system to system, and any small change in hardware would prevent it from booting or running in hard real time. All of this was getting Jacobs Automation too far away from their core goal of writing a real time application that would connect their automation machinery to the world.

From these experiences, the need for Pack4Linux was born. Pack4Linux fit Jacobs requirements because it allowed them to use their Compact Flash disks unmodified on their different systems, it was open source, and it ran embedded and headless. It also sped up their deployment time, as they didn't need to patch the system for RTAI, and it ran persistent off of Compact Flash.

After the first stable Pack4Linux release, Jacobs found that the release had lived up to those goals. They now distribute their communications gateway application with their hardware on a 4G industrial Compact Flash disk which can be taken from one PC system to another, and can boot and work flawlessly in hard real time. They have since added other drivers to their application such as digital IO, and modifications to be able to remote into the machine over the real time Ethernet card when needed.

5 Preparing for the Future

While open source automation has achieved a moderate level of success, some fundamental changes need to happen so that it can become the standard of the future.

Firstly, online resources need to be refactored. Websites for real time technologies need to be revamped for easier navigation and more direct access to non-technical, end-user documentation. Second, real time technologies need to put more emphasis on documentation, and less on additional features, as additional features are useless if noone uses them. Third, real time technologies, and the automation industry as a whole, needs to make the jump to 64-bit. Some of these technologies, such as RTAI, have released experimental 64-bit versions. In the future, Pack4Linux plans to release experimental 64-bit versions along side stable 32-bit versions to help ease this transition for people.

With everybody's help, we can make open source the automation standard of the future.