Project-Builder.org: A GPL Continuous Packaging Solution

Bruno Cornec
Open Source and Linux Master Technology Architect
2010-06-23
Introducing Myself

- Software engineering and Unixes since 1988
  - Mostly Configuration Management Systems (CMS), Build systems, quality tools, on multiple commercial Unix systems
  - Discover Open Source & Linux (OSL) & first contributions in 1993
  - Full time on OSL since 1995, first as HP reseller then @HP

- Currently:
  - Master Technology **Architect** on OSL for the HP/Intel Solution Center, Grenoble
  - OSL HP **Advocate**
  - EMEA OSL HP **Profession Lead**
  - Solutions Linux Conference board member
  - MondoRescue, Dploy.org, Project-Builder.org **project lead**
  - LinuxCOE, mrepo, tellico, rinse, fossology, collectl **contributor**
  - Mandriva, Fedora **packager**
What is the problem?

Users / Sysadmins environment:

Users: easyness (GUI, CLI)
Admins: distribution compliance, smooth integration – reduce admin load
What is the problem?

Developers environment:

- Flat Files
- CMS
- CVS
- SVN
- GIT, ...

Sourceforge
ftp, web

Project

autotools?
Install proc?
Soft. Eng.?
Fear of distro?
What is the problem?

Packagers environment:

Does s/he only exists?
On time? Lag behind?
Distro coverage (ver, #)
Beta avail?

CMS  CVS  SVN  GIT, ...

Distro ftp, web

rpmlint? lintian?
What is the solution?

- Packaging should be a project concern as well as coding, testing, installing, .... especially for smaller projects
- Package early, package always
- Use packaging as your only way of delivery (not a dream) Overhead minimal (not much longer than providing a tgz), maximum benefit (consistancy, reproduceability, team work improvement, distribution integration, improved deployment)
- Packaging is in fact a marketing activity based on a technical content. It's an easy way to extend your user base, and improve your community. Makes tests easy.
- THE SOLUTION IS CONTINUOUS PACKAGING
Multi platform life cycle

User Requirements

Functional Specification

Design Specification

Coding

Unit tests

Integration tests

Validation

Coding

©2010 HP/Intel - Project-Builder.org - A GPL continuous packaging solution
My answer

- I'm a sysadmin so I want to deal with packages only (deployment server integration)
- I'm a developer so I want to ease my tests without screwing up my system and testing as a std user/admin
- I'm a packager so I want to streamline that work for new projects I'm interested in packaging
- I'm the marketing department of my own project and use packages as a “competitive advantage”
- I'm a software engineer and want no duplication of code nor information (metadata)
- Founding no tool existing to help me doing all that, so I started writing one to scratch my own itch
HP+Intel Solution Brief

One Source for Multiple Platforms: Best Practices
Improving Quality and Reducing TCO on the Road to Intel® Xeon® processors Migration

Common Code Development for Multiple Platforms

For network equipment and software solutions providers, developing and maintaining the same applications on multiple platforms are often the result of the three exercises: Migrating to industry standard platforms themselves (Solaris or Linux), dealing with customers moving from SPARC® to industry standard platforms, or dealing with customers’ heterogeneous systems comprised of all of the above.

During a software development project development lifecycle, continuous packaging enables the production of identical deliverables at each step of the development process including testing, integration, validation, or final delivery, while addressing and taking into account dependencies and potential conflicts. This very important task generates the need for packaging easily and often.

Applications developed and packaged for telecom operators are frequently coded in C or C++, for which at least 90% of the code is identical between a Solaris and a Linux platform. HP and Intel’s experts took advantage of this favorable condition to reduce costs and enhance overall efficiency through the creation of a tool aimed at easier applications development, packaging, and maintenance on multiple platforms.

Migration activities with support from HP and Intel

The centralized HP Laser Team gathers Oracle experts who work closely with infrastructure project leaders on infrastructure management, provisioning, deployment, imaging and other related activities.

In addition the HP System Technology and Software Division (STSD) team apply expertise to the development of the Solara to Linux Porting Kit (LPK), while the HP Enterprise Services (HTSS) is in charge of porting activities, consulting and outsourcing of portal systems and infrastructure.

Description of the Development Process

Supporting several target platforms in the development process involves providing tools at two important steps:

Step 1: Code development

The customer may already have an existing process and build chain to produce the Solara binaries from the source repository. Using the LPK helps avoid the code differences in regards to Linux and provides the low level modules to emulate the Solaris environment and produce the binaries on a Linux environment.

The HP Intel CME Solution Center helps with the overall process and with the collaboration of both HP SPARC team and the customer’s team.

Step 2: Software Packaging

Since 2007, HP Intel CME Solution Center contributors have developed the open source project shippable.org project in order to support continuous software packaging activities. This tool has been developed to avoid code duplication and provide separation from the project and management in a distributed source repository. In addition, it makes the code agnostic to the build environment and targeted operating system. It provides generic and virtual machines environments.

Along with the code development, intermediate deliveries are packaged just like final deliveries of the applications. This tool allows customers to both minimize the risk and leverage maximum code reuse when moving from one to multiple platforms, while ensuring the quality of the software.

Steps 1 and 2 will soon be joined in one generic tool, supporting and performing code development and software packaging.

HP & Intel Key hardware components

HP Blade System c7000 Enclosure

The BladeSystem c7000 enclosure provides all the power, cooling, and I/O infrastructure needed to support modular server, interconnect, and storage components today and throughout the next several years. The enclosure is 10U high and holds up to 32 servers or 16 storage blades plus optional redundant network and storage interconnect modules.

Intel® Xeon® processors

The Intel Xeon processor 5500 series are built with 45nm new Nehalem micro-architecture with up to eight cores in a two-processor configuration. This new micro architecture delivers more performance in the same platforms and at the same power consumption, giving customers the flexibility to match performance, power and cost requirements with your unique requirements and delivering advantages beyond just pure performance.

Technology for better business outcomes

© 2009 Hewlett-Packard Development Company, L.P. The information contained herein is subject to change without notice. The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

For more information, visit http://www.hpintelleco.net

One Source in Multiple Platforms, January 2009
Goals

- Project-builder main goal is to help you package continuously being agnostic:
  - CMS agnostic (no CMS but guys it's 21st century now, SVN, CVS, Mercurial, GIT, SVK....)
  - OS agnostic: Linux: RPM, deb, ebuild, slack based, ... 60+ distros at the moment – repositories for yum, urpmi, apt. Solaris pkg.
  - Build environment agnostic: local, VM (QEMU, KVM...), VE (rpmbootstrap, rinse, mock, debootstrap...)
  - No project impact (preserves the md5sum of the delivered upstream sources). Can be completely external.
Goodies

- Project–builder provides additional goodies:
  - Easy VMs/VEs management
  - Macro system with perl variables to avoid duplication
  - Skeleton generation to help starting
  - Manages delivery up to your ftp server (ssh based)
  - Manages announces on mailing lists
  - Integration of tests in the process
  - Manages patches when not upstream
  - Checks validity of packages built (lintian, rpmlint)
  - Easy creation of new versions for upstream management
VM and VE Management

- **VE Management**
  - Provides rpmbootstrap (modeled after debootstrap) to create chroot for multiple RPM based distributions (Fedora, Mandriva, CentOS tested, OpenSuSE in progress) or Uses debootstrap for .deb based ones.

- **VM Management:**
  - Uses KVM or QEMU iso install possibilities
  - Final setup to add pb tools in VM|VE and a pb build account
  - Snapshot of the VM|VE available
  - After that building in the VM|VE is similar to building locally (using build2vm|build2ve instead of build2pkg).

- **ssh communication for VM**
Metadata Management

- Packaging process independent of the project and external
- Metadata isolated – No duplication

Tree will look like this:

```
maint pbdefdir                    PBDEFDIR
  |                                |
  |                                |
  pbproj1                          PBPROJ
  |                                |
  pbproj2                          pbproj1
  |                                |
  |                                |
* * *                              * * *
  tag    dev    pbconf    ...    build    delivery    PBCONFDIR
  |        |        |        |        |        |        |
  |        |        |        |        |        |        |
  pbrc    PBDESTDIR    PBBUILDDIR
  |        |        |        |        |        |        |
  1.1     dev    tag
  |        |        |
  |        |
  1.0     1.1
  |        |
  PBROOTDIR

  pkg1    pbproj1.pb    pbfilter    pbcl
  |        |        |        |
  |        |        |
  rpm    deb    pbfilter
```

(*) By default, if no relocation in .pbrc, dev dir is taken in the maint pbdefdir (when appropriate) Names under a pbproj and the corresponding pbconf should be similar
Configuration file usage

- Macro example:

  ./rpmbootstrap/pbfilter/all.pbf:filter PBSUMMARY = rpmbootstrap is a tool similar to debootstrap for RPM based distributions
  ./rpmbootstrap/rpm/rpmbootstrap.spec:Summary: PBSUMMARY
  ./rpmbootstrap/deb/control:Description: PBSUMMARY
  ./rpmbootstrap/pkg/pkginfo:NAME="PBSUMMARY"

- Instantiation possible from distro-ver-arch, distro-ver, distro, distro-family, distro-type

- Similar approach for all other parameters (Cf: man pb.conf)
Multi platform development cycle

SSWL

< 5%

90+%

SLWL

< 10%
The Big picture

Packagers

Developers

Project

Packages

newver / newproj
cms2build
build2vm
newvm / setupvm
build2pkg
pkg2ssh
build2ssh
Commands

cms2build: Create tar files for the project under your CMS. CMS supported are SVN and CVS parameters are packages to build if not using default list
build2pkg: Create packages for your running distribution
build2ssh: Send the tar files to a SSH host
pkg2ssh: Send the packages built to a SSH host
build2vm, build2ve: Create packages in VMs/VEs, launching them if needed and send those packages to a SSH host once built
launchvm: Launch one virtual machine
script2vm, script2ve: Launch a VM/VE if needed and executes a script on it
test2vm, test2ve: Test inside a VM/VE
newvm, newve: Create a new virtual machine
setupvm, setupve: Setup a virtual machine for pb usage
newver: Create a new version of the project derived from the current one
newproj: Create a new project and a template set of configuration files under pbconf
Multi platform packaging cycle

Developers

Packagers

VMs/VEs

Packages

VM/VE
To be done

- More Tests on VEs (mock (obsolete), debootstrap, rinse OK + patches, rpmbootstrap introduced) LSB chroot support.
- Support for libvirt, virsh
- Other CMS (Bazaar, ClearCase…) only when/if needed
- Other VMs (VMWare, Xen, …) only when/if needed
- Multiple delivery means
- Look at interactions with Buildbot
- Add signature support
- Create a reference documentation in progress (Lab doc exist)
- Config-Model for configuration file management
Learn Project-Builder.org

- Start with the Lab (63 pages) and come to the session 1751 (2010-06-23 at 12:30PM)
- Use man (pb, pb.conf + 8 ProjectBuilder::* man pages)
- Use the mailing-list pb-announce and pb-devel at http://www.mondorescue.org/sympa
- Use examples from http://trac.project-builder.org/browser/projects/
Web Resources

Project-Builder Web site / Trac / Wiki
http://www.project-builder.org
http://trac.project-builder.org

Open Source at HP
http://opensource.hp.com

Other tools of interest:
– SuSE/Fedora/Mandriva Build Systems
– Buildbot
– vcs-pkg.org
"Changes are never easy to make. There is comfort and safety in tradition, but change must come, no matter how painful or expensive it may be."

-Bill Hewlett

Contact - Thanks

Bruno.Cornec@hp.com

(Open Source and Linux Technology Architect at the HP/Intel Solution Center)

http://www.hp.com/linux
http://opensource.hp.com

Thanks goes to:

Linus Torvalds, Richard Stallman, Eric Raymond, Nat Makarevitch, René Cognenc, Eric Dumas, Rémy Card, Bdale Garbee, Bryan Gartner, Craig Lamparter, Lee Mayes, Gallig Renaud, Andree Leidenfrost, Phil Robb, Bob Gobeille, Martin Michlmayr among others, for their work and devotion to the Open Source Software cause...

and my family for their patience :-)

"Changes are never easy to make. There is comfort and safety in tradition, but change must come, no matter how painful or expensive it may be."

-Bill Hewlett
Q&A