Crowdsourcing upstream Refactoring

Tom Marble, Bdale Garbee

January 29, 2013

Contents

1 Crowdsourcing upstream Refactoring

file:cover.txt

1.1 Overview: Topics to cover

...overview...

2 Background

2.1 About Tom

tmarble

2.1.1 Sun: OpenJDK

Part of the Java open source Core Strategy Team at Sun, first OpenJDK Ambassador (I went to a lot of conferences)

- FOSDEM
- FISL
- OSCON

ApacheCon 2006: Sun unBOF/Party

/Copyright 2006 Ted Leung: https://secure.flickr.com/photos/twleung/268116213/

2.1.2 Consulting

Cybersecurity
Performance Analysis and Benchmarking
Clojure
Very interested in open, embedded systems

2.2 About Bdale

bdale

2.2.1 One of the longest contuinally serving Debian Developers

DebConf 11

DebConf11
24.-30.07.2011. Banja Luka, BIH
Rule 1: United. We stand.
Rule 2: Divided ... we argue a bit on the lists and then - see Rule 1.

http://wiki.debconf.org/wiki/DebConf11/Pictures/GroupPhoto

2.2.2 AltusMetrum

Open Hardware - Open Software - Ham Radio - Rocket Science

3 Upstreams and distros have different goals

There are big differences between what

3.1 upstreams want

Simple user experience to download and run
Simple development experience
More features! :-P

3.2 distros want

A system built from consistently packaged, interoperable software elements
Complete modularity for

- maintainability
- reusability (archive size)
- security (fix problems **once**)
- license clarity

Clearly defined build dependencies
Repeatable, off-line, fully automated build procedures file:debian.svg

3.3 and ideally

We want both upstreams and distros to be satisfied

We want to increase the productivity of everyone so we can make more Free Software

file:Tux.svg

4 What does this problem look like?

Today each distro needs to do refactoring work in order to adhere to their own guidlines.

- As a community we are duplicating a lot of tricky effort
- We are doing a poor job of sharing knowledge about packaging
- We don't have a consistent plan for collaborating with upstreams

4.1 Because upstreams

Upstreams want to

- only have to worry about one "user" distribution
- simplify the build process by checking in binary artifacts

But these artifacts do not have corresponding source

- This could be for convenience – there is an upstream FLOSS project, but building it from source is tedious
- This could be a **binary BLOB**
- Licenses do not accompany these artifacts
- Build tools may use maven (downloads binary artifacts at run time) or worse – a custom built build tool.

4.2 . . . make more work for distros

Distros

1. only want one copy of each software project (library) in our archive
2. must have corresponding source

3. must have license clarity: is redistribution allowed?
4. must build repeatably (esp. in a clean room, offline environment)
5. must “discover” and refactor each dependency… find the source, find the license, and package it
6. if ! done then goto 1 :)

5 A “real world” example: OpenRocket

Some of the biggest challenges are Java upstreams.

One such upstream is OpenRocket (could not be nicer people!)

Motor configuration: [No motors]

Nuke Pro-Maxx
Length 112 cm, max. diameter 5.84 cm
Mass with no motors 584 g

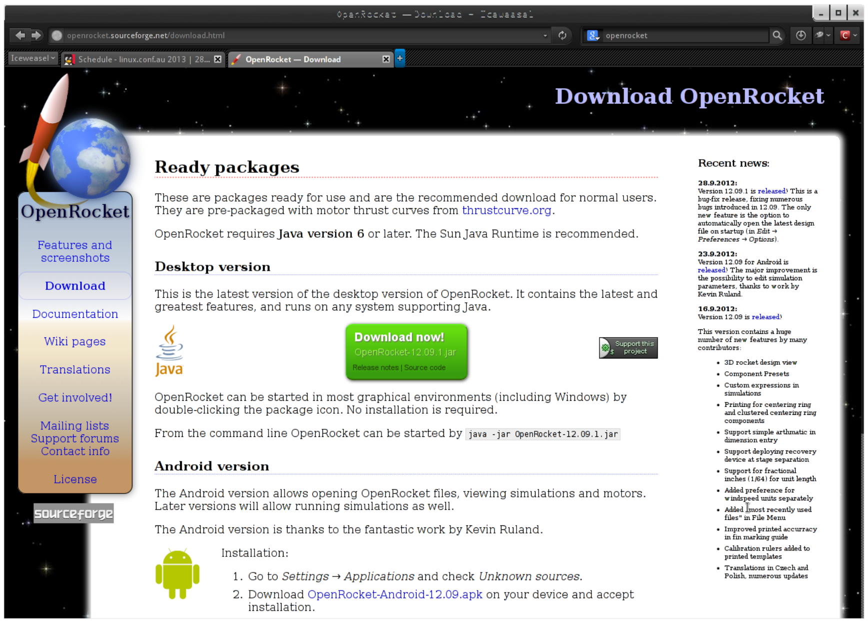
Stability:3.49 cal
CG:63.5 cm
CP:83.9 cm
at M=0.30

Apogee: N/A
Max. velocity: N/A
Max. acceleration: N/A

Warning:
Discontinuity in rocket body diameter.

5.1 upstream provides an uberjar

In packaging for Debian we find… an **uberjar**



Download OpenRocket

Ready packages

These are packages ready for use and are the recommended download for normal users. They are pre-packaged with motor thrust curves from thrustcurve.org.

OpenRocket requires **Java version 6** or later. The Sun Java Runtime is recommended.

Desktop version

This is the latest version of the desktop version of OpenRocket. It contains the latest and greatest features, and runs on any graphical system supporting Java.

OpenRocket can be started in most graphical environments (including Windows) by double-clicking the package icon. No installation is required.

From the command line OpenRocket can be started by `java -jar OpenRocket-12.09.1.jar`

Android version

The Android version allows opening OpenRocket files, viewing simulations and motors. Later versions will allow running simulations as well.

The Android version is thanks to the fantastic work by Kevin Ruland.

Installation:

1. Go to *Settings* → *Applications* and check *Unknown sources*.
2. Download OpenRocket-Android-12.09.apk on your device and accept installation.

Recent news:

28.9.2012:
Version 12.09.1 is released! This is a bug-fix release, fixing numerous bugs introduced in 12.09. The only new feature is the option to automatically open the latest design file on startup (in *Edit* → *Preferences* → *Options*).

23.9.2012:
Version 12.09 for Android is released! The major improvement is the possibility to edit simulation parameters, thanks to work by Kevin Ruland.

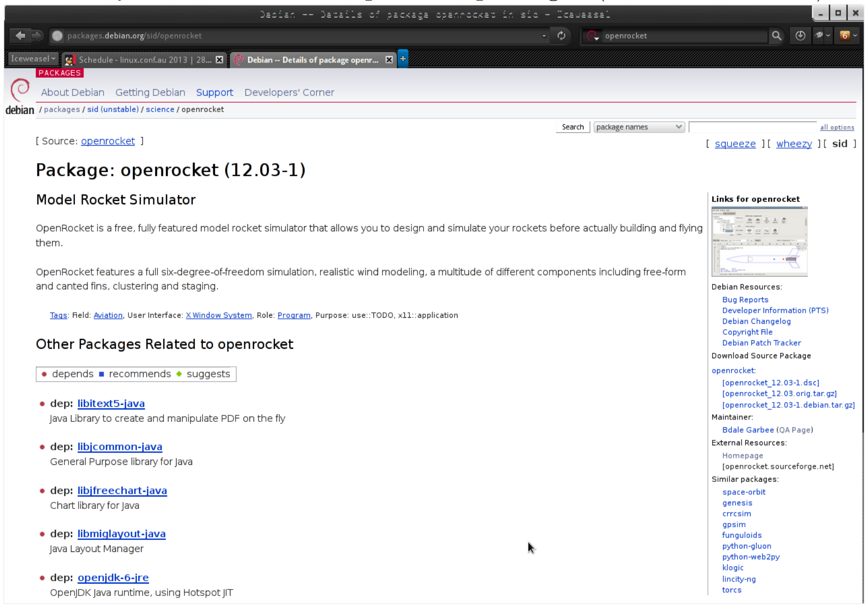
16.9.2012:
Version 12.09 is released!

This version contains a huge number of new features by many contributors:

- 3D rocket design view
- Component Presets
- Custom expressions in simulations
- Printing for centering ring and clustered centering ring components
- Support simple arthmatic in dimension entry
- Support deploying recovery device at stage separation
- Support for fractional inches (1/64) for unit length
- Added preference for windspeed units separately
- Added "most recently used files" in File Menu
- Improved printed accurracy in fin marking guide
- Calibration rulers added to printed templates
- Translations in Czech and Polish, numerous updates

5.2 . . . but we can't use it

We need fully refactored interdependent packages (each with source):


Debian -- Details of package openrocket in sid - Iceweasel
packages.debian.org/sid/openrocket
openrocket
Iceweasel
Schedule - linux.conf.au 2013 | 28...
Debian -- Details of package openr...
PACKAGES
debian
About Debian
Getting Debian
Support
Developers' Corner
Search
package names
all options

/ packages / sid (unstable) / science / openrocket

[Source: openrocket]

[squeeze][wheezy][**sid**]

Package: openrocket (12.03-1)

Model Rocket Simulator

OpenRocket is a free, fully featured model rocket simulator that allows you to design and simulate your rockets before actually building and flying them.

OpenRocket features a full six-degree-of-freedom simulation, realistic wind modeling, a multitude of different components including free-form and canted fins, clustering and staging.

Tags: Field: Aviation, User Interface: X Window System, Role: Program, Purpose: use::TODO, x11::application

Other Packages Related to openrocket

● depends ■ recommends ◆ suggests

- **dep: libitext5-java**
  Java Library to create and manipulate PDF on the fly
- **dep: libjcommon-java**
  General Purpose library for Java
- **dep: libjfreechart-java**
  Chart library for Java
- **dep: libmiglayout-java**
  Java Layout Manager
- **dep: openjdk-6-jre**
  OpenJDK Java runtime, using Hotspot JIT

Links for openrocket

Debian Resources:
- Bug Reports
- Developer Information (PTS)
- Debian Changelog
- Copyright File
- Debian Patch Tracker

Download Source Package

openrocket:
- [openrocket_12.03-1.dsc]
- [openrocket_12.03.orig.tar.gz]
- [openrocket_12.03-1.debian.tar.gz]

Maintainer:
- Bdale Garbee (QA Page)

External Resources:
- Homepage [openrocket.sourceforge.net]

Similar packages:
- space-orbit
- genesis
- crrcsim
- gpsim
- funguloids
- python-gluon
- python-web2py
- klogic
- lincity-ng
- torcs

6 What can we do?

Let's collaborate on educating upstreams!

6.1 Provide upstream guides

Let's draft/extend distribution agnostic refactoring best practices that offer an incentive:

"How to accelerate the adoption of your project!"

Debian: https://wiki.debian.org/UpstreamGuide includes pointers to: A blog series by François Marier https://wiki.debian.org/AdvantagesForUpstream

Fedora (Tom "spot" Calloway): http://www.theopensourceway.org/wiki/How_to_tell_if_a_FLOS

does your distro have an upstream guide?

6.2 Share metadata: CUDF

Repository is a set of Packages (installed or available)

Each package has a name and a version (there are many different versioning schemes :( )

Users want to request installing, upgrading, removing packages (etc.)

Users have preferences (trendy, cautious, etc.)

This is the "Upgrade Problem" and each distro handles it differently

6.2.1 Mancoosi

"Managing the Complexity of Open Source Software"

Research collaboration incluing the Université Paris Diderot, INRIA and others into the Upgrade Problem has proposed separating "dependency solving" from the the "UI" of package managers.

The result has been a pluggable dependency solver competition which has significantly outperformed **ad hoc**, hard coded solvers. The winning solver a much better score of maintaining constraints and in significantly less time (100:1).

The API for plugging in dependency solvers is based on CUDF: the Common Upgrade Description Format

6.2.2 CUDF example

preamble: property: bugs: int = 0, suite: enum (stable, unstable) = " stable ", package: car version: 1 depends: engine, wheel > 2, door, battery <= 13 installed: true bugs: 183

package: bicycle version: 7 suite: unstable

package: gasoline-engine version: 1 depends: turbo provides: engine conflicts: engine, gasoline-engine installed: true . . .

request: install: bicycle, gasoline-engine = 1 upgrade: door, wheel > 3

6.2.3 CUDF adoption

Uses of CUDF

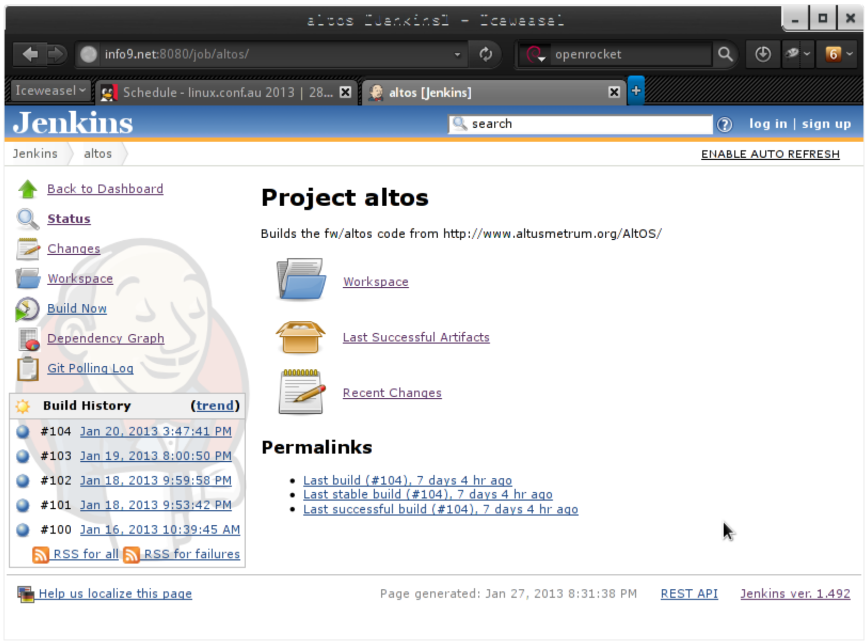
- The Eclipse IDE plugin manager
- The OCaml native package manager (OPAM)
- OpenSUSE libzypp
- Debian **apt-cudf** allows pluging in alternative dependency solvers to APT (including user preferences)

What can we ask upstreams to care about?
What metadata should they include?

6.3 Use continuous integration

Let's encourage upstreams to setup and share Jenkins!

Keep upsteam breakage to a minimum (or at least notify right away)


altos [Jenkins] - Iceweasel

info9.net:8080/job/altos/

Iceweasel | Schedule - linux.conf.au 2013 | 28... | altos [Jenkins]

Jenkins | search | log in | sign up

Jenkins › altos | ENABLE AUTO REFRESH

- Back to Dashboard
- Status
- Changes
- Workspace
- Build Now
- Dependency Graph
- Git Polling Log

Build History (trend)

#104	Jan 20, 2013 3:47:41 PM
#103	Jan 19, 2013 8:00:50 PM
#102	Jan 18, 2013 9:59:58 PM
#101	Jan 18, 2013 9:53:42 PM
#100	Jan 16, 2013 10:39:45 AM

RSS for all | RSS for failures

Project altos

Builds the fw/altos code from http://www.altusmetrum.org/AltOS/

- Workspace
- Last Successful Artifacts
- Recent Changes

Permalinks

- Last build (#104), 7 days 4 hr ago
- Last stable build (#104), 7 days 4 hr ago
- Last successful build (#104), 7 days 4 hr ago

Help us localize this page | Page generated: Jan 27, 2013 8:31:38 PM | REST API | Jenkins ver. 1.492

7 Conclusion

The problem is

- upstreams and distros have different goals
- yet we want everyone to be satisfied and productive

What can we do?

- Provide upstream guides and education
- Share metadata: CUDF
- Use continuous integration

Special Thanks to Stefano Zacchiroli for providing many great ideas and pointers.

We will share "slides" on Tom's microblogs: @tmarble

8 Q/A

Questions?
    Thoughts?
    Discussion

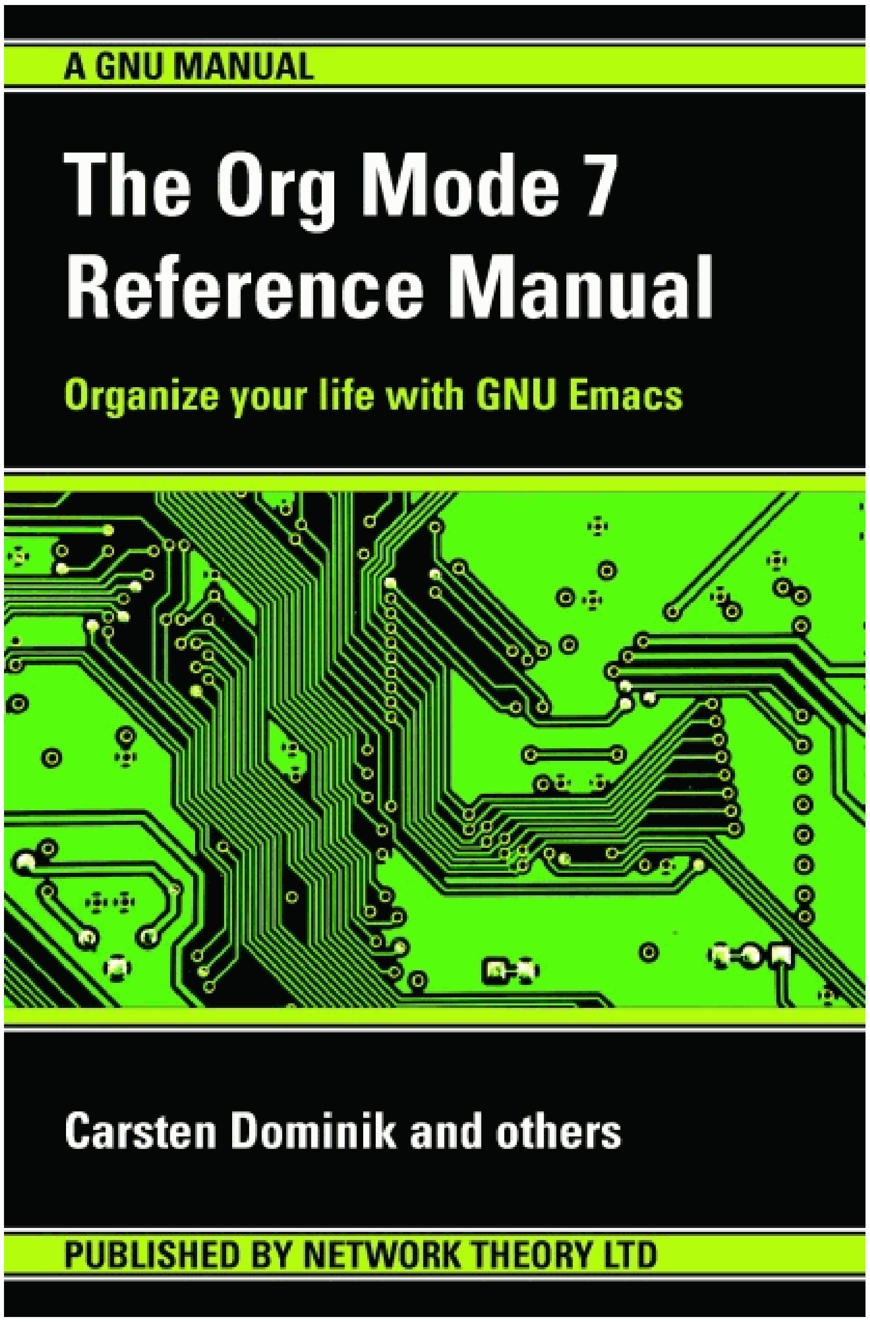
9 Background material

9.1 More on Mancoosi and CUDF

Mancoosi http://www.mancoosi.org/
    Publications by Stefano Zacchiroli http://upsilon.cc/z̃ack/research/publications/

9.2 What is that presentation tool?

Emacs!
    This is **org-tree-slide** from https://github.com/takaxp/org-tree-slide
    For more on org mode see http://orgmode.org/org.html
    Yes I will share my “slides” on my website http://tmarble.info9.net



A GNU MANUAL

The Org Mode 7 Reference Manual

Organize your life with GNU Emacs

Carsten Dominik and others

PUBLISHED BY NETWORK THEORY LTD