`apt-get install cran bioc`: On automated builds of 1700 R packages for Debian

Dirk Eddelbuettel[1] David Vernazobres[2]
Albrecht Gebhard [3] Steffen Möller [4]

[1]Debian Project

[2]Universität Münster

[3]Universität Klagenfurt

[4]Universität Lübeck

UseR! 2007 – Iowa State University, August 8 - 10, 2007

Outline

R – and its repos

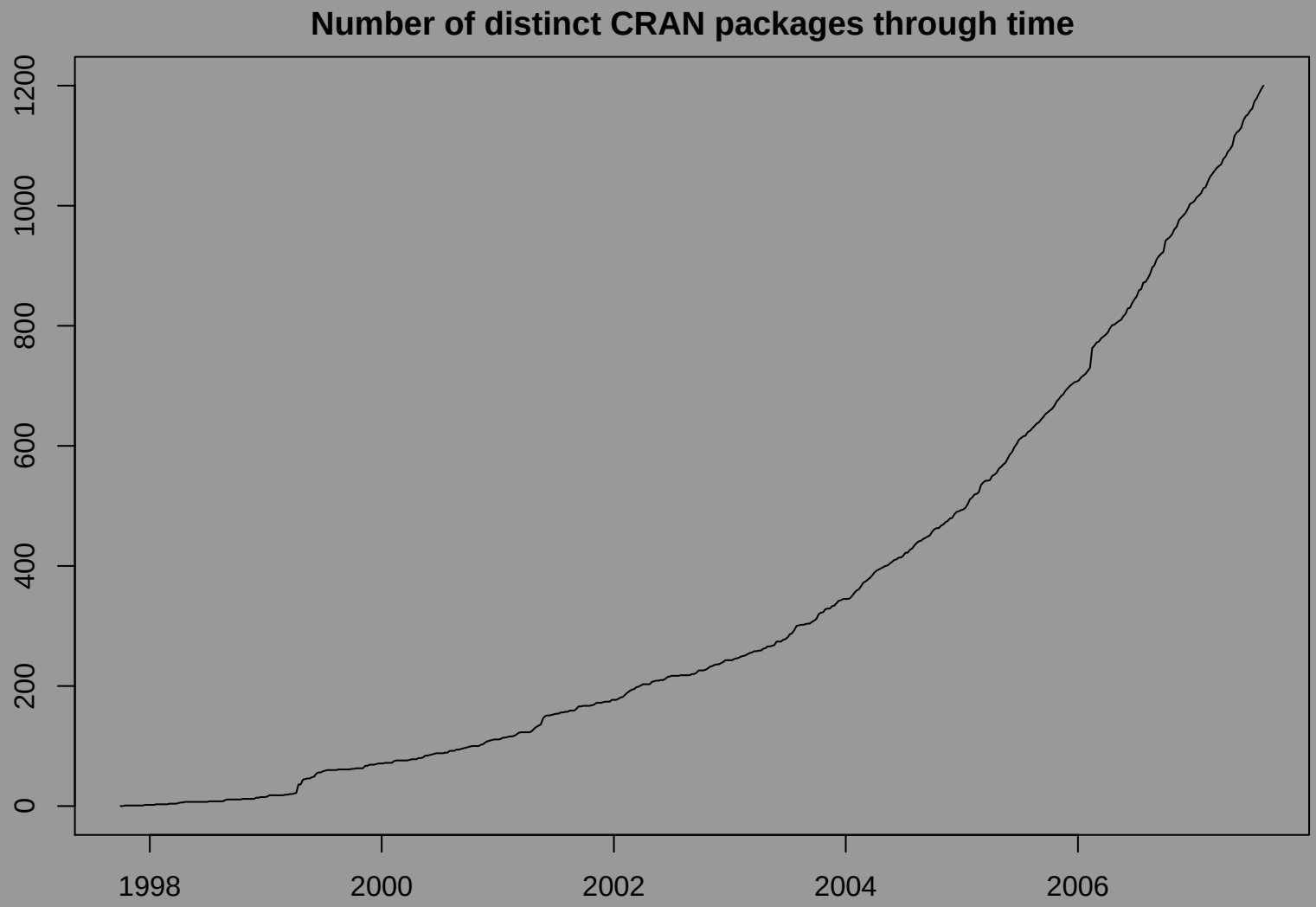
An open statistical language / environment – with lots of excellent code contributions

To restate a what is well known at *useR! 2007*:

- R has become a standard for statistical applications and research
- As *"success has many fathers"*, several key drivers can be identified as to why R has done so remarkably well
- However, we would like to stress *repos* here
- CRAN has been one of the drivers: an open yet rigourously QA'ed repostory which has experienced tremendous growth
- BioConductor has become a key technology enabler in bioinformatics, and provides a repo for that community
- Omegahat was meant to be a driver for the 'next R' and is still an excellent experimental area

CRAN Packages

Exponential Growth



The count is based on files found at CRAN/src/contrib and CRAN/src/contrib/Archive and may exhibit survivorship bias

Debian and Ubuntu

Open Linux distributions

A few key points:

- Debian is *the* community-driven Linux distribution where numerous volunteers provide close to twenty-thousand packages on around ten or so architectures
- Packages and package management “just work”: with arguably the most advanced and robust package management system, and a tremendous build and test infrastructure
- Ubuntu has taken Debian, added a fair amount of spit and polish, as well as regular bi-annual releases, and is rapidly gaining mind- and well as market-share as the Linux distribution to beat
- Lastly, we note that the CRAN backend is also implemented on Debian

Debian and R

Happy foreverafter

R itself, several related tools – and CRAN packages:

- R and Debian go back a long time:

```
edd@ron:~> tail -8 src/debian/R/R-2.5.1/debian/changelog

r-base (0.61.0-1) unstable; urgency=low

  * Initial release

 -- Douglas Bates <bates@stat.wisc.edu>  Tue, 23 Dec 1997 10:01:21 -0600
```

- Several related packages were added: XGobi (and later GGobi), ESS, PI/R (R inside Postgresql), Rserve, rpy, littler, rkward (a GUI), Shogun (ML tool).
- And around fifty CRAN packages have been added by several Debian developers.

Why build Debian R packages?

Bates, Eddelbuettel and Gebhard (UseR!, 2004) listed a number of reason that still hold:

- **Dependencies** are resolved automatically: *it just works*
- **Convenience** of installing binary packages via `apt-get`
- **Quality control** as build daemons, automated rebuilds, porting, ... all ensure that everything is pretty much buildable all the time
- **Scalability** as building one binary package and scripting installation on a cluster beats doing lots of manual installations
- **Common platform** as Debian forms the base for Ubuntu and several other derivative or single-focus distributions
- **Different architectures** ranging from small arm or mips based systems to amd64, sparc64, hppa or even s390 mainframes
- **Audience** given the reach of Debian and Ubuntu, large number of users can be reached with little effort

So what is a Debian package?

And how do I build it?

Building a Debian package is similar to using `R CMD binary` etc:

- Reads meta-information is read from the files in the debian/ directory
  - debian/control (similar to R's DESCRIPTION) lists names, maintainers, build- and run-time dependencies
  - debian/copyright lists all author, license holders and copyright statements
  - debian/changelog provides current and past version numbers with a list of all changes in chronological fashion
  - debian/rules is a Makefile containing all steps to configure, build, install, package-create and clean
- Employs a number of external scripts and tools tie into this, similar to what R has below `$RHOME/share`

An Example:

R DESCRIPTION for gWidgetsRGtk2

```
Package: gWidgetsRGtk2
Version: 0.0-19
Date: 2007-7-10
Title: Toolkit implementation of gWidgets for RGtk2
Author: Michael Lawrence, John Verzani
Maintainer: John Verzani <gwidgetsrgtk@gmail.com>
Depends: methods, gWidgets(>= 0.0.15), RGtk2
Suggests: cairoDevice (>= 2.2.0)
Description: Port of gWidgets API to RGtk2
License: GPL version 2 or newer
URL: http://www.math.csi.cuny.edu/pmg
LazyLoad: yes
Packaged: Wed Jul 11 16:38:07 2007; verzani
```

An Example: continued

debian/control for r-cran-plotandplayrgtk

```
Source: gwidgetsrgtk2
Section: math
Priority: optional
Maintainer: CRAN/BioC packagers <pkg-bioc-devel@lists.alioth.debian.org>
Standards-Version: 3.7.0
Build-Depends: r-base-dev (>> 2.3.0), debhelper (>> 4.0.0), \
         cdbs, r-cran-gwidgets, r-cran-rgtk2, dpatch

Package: r-cran-gwidgetsrgtk2
Architecture: all
Depends:  r-base-core , r-cran-gwidgets
Description: GNU R package "Toolkit implementation of gWidgets for RGtk2"
 Port of gWidgets API to RGtk2
 .
 Authors: Michael Lawrence, John Verzani
 Date: 2007-7-10
 .
  Homepage: http://www.math.csi.cuny.edu/pmg
```

An Example: Comments

Mapping from R to Debian

- The script maps the Build-Depends on RGtk2, gWidgets and and R itself
- From Debian, we get r-cran-rgtk2 and r-base-core – which are already available as part of the distribution
- We also need r-cran-gwidgets – a recursive dependency on another CRAN packages that we do have to build first
- Generally speaking, we need to map ‘opaque’ requirements like ’XML’ into the particular library naming schemes – this is done mostly with static information and lookups
- The dependencies are then resolved by walking down the dependency graph structure
- With build- and run-time dependencies covered, the actual building of the package is not much more than calling ’R CMD binary’ with a particular target directory.

Collaborative approach

Past and present

Gebhard had written an initial package creation script for SuSE Linux that he converted to Debian. Eddelbuettel extended it further. At the same time, Matt Hope created the Alioth project where these iniatives converged.

Using the Alioth host of the Debian Project – which provides free accounts to everybody willing to work on a project – we have

- SVN source control
- mailing list(s)
- storage to place packages

Several team members focus more on BioConductor software, and integration into 'virtual' computing grids – see Möller et al (NETTAB 2007).

Our technology

Using sripts to automate package building

The authorative source is
http://wiki.debian.org/AliothPkgBioc.
A brief synopsis is as follows:

- Check out sources from SVN
- Install the required Debian packages
- Call a first script to prepare directory layout
- Call another script to prepare a local package mirror
- Call the build script to start building per repo (cran, bioc-2.0, omegahat, ...)
- Upload packages

Most of these steps do not require intervention. However, the penultimate building step may require additions or corrections to the static mapping tables to encode package interdependencies.

Open issues

Things to address

What are next steps:

- Mirror space and bandwidth would be very helpful
- Volunteers are needed to broaden the build process: Debian/Ubuntu and R skills
- Better dependency mapping, possibly in a distribution-agnostic way
- Better package descriptions (that can be understand by non-subject matter experts)
- Information reuse: integration with CRANberries, possibly GUIs,
- Package management cooperation: R doesn't know it is on Debian, Debian doesn't know R has install.packages()

Conclusion

Our contributions

The ‘Debianization’ of CRAN, BioConductor and OmegaHat now provides:

- Close to 2000 new packages available for x86 and amd64 via alioth
- A proof of concept that this can work just like binary installs on other operating systems ...
- ... yet play to the strength of Linux and Debian: scale up to cluster size and work reliably with dependencies