Building Custom Debian Distributions with the CDDTk

IV Jornades de Programari Lliure
Campus de Vilanova i la Geltrú, UPC

8 July 2005

Sergio Talens-Oliag
sto@debian.org

What is Debian?

- The **Debian Project** is an association of individuals who have made common cause to create a *free* operating system.
- This operating system is based on the **Linux** Kernel and the **GNU** tools and is called **Debian GNU/Linux**, or simply **Debian** for short.
- There is work in progress to support other **Unix** like kernels (**Hurd**, **kFreeBSD**, ...)

Why Debian?

- More than 900 developers from all around the world that agree with the *Debian Social Contract*.
- More than 10.000 packages for 12 architectures.
- Standarization: *Debian Policy Manual.*
- Quality assurance: BTS, QA Team.
- Security team and security updates.
- All *software* in main is free (it has to comply with the *Debian Free Software Guidelines*).

Why not Debian?

- Too many packages: general users are only interested in a subset of those packages.
- Complicated to install and configure for unexperienced users.
- Special interest groups have different skills and needs (and usually they are users, not system administrators).
- Solution: *Custom and Derived Distributions*

Derived Debian Distributions

- Operating System Distributions based on Debian:
  - `deb` package format
  - `apt` repositories with reduced package lists
  - Local packages and custom versions of official ones
  - Custom installers (PGI, Anaconda for Debian)
- Problems:
  - Duplication of the infrastructure already in Debian
  - Packages don't have to be Policy Compliant
  - Local changes don't come back to Debian

Custom vs Derived Distributions

- **Custom Debian Distributions** are distributions derived from **Debian** that are 100% **Debian**.
- No fork from **Debian**: do not make a separate distribution but make **Debian** fit for special purpose needs instead.
- All CDD contribute to **Debian** and all **Debian** improvements contribute to all **Custom Debian Distributions**.

Motivation (1)

- Support of target users with common profile:
  - Less technical competence
  - Not able to install upstream programs with acceptable effort
  - No interest in administration
  - Interest in defined subset of available Free Software
  - Need for easy usage
  - Defined security profile

Motivation (2)

- Support of administrators with common profile:
  - Limited time frame
  - Seaking for time saving in often repeated tasks
  - Lack of specialist knowledge
- With CDD you can:
  - Market to specific groups and needs
  - Provide and/or sell user support
  - Create targeted documentation

Existing CDDs

- **Debian Junior**: Debian for children from 1 to 99
- **Debian-Med**: Debian in Health Care
- **Debian-Edu**: Debian for Education
- **DeMuDi**: Debian Multimedia Distribution
- **Debian-Desktop**: Debian GNU/Linux for everybody
- **Debian-Lex**: Debian GNU/Linux for Lawyers

Existing CDDs (2)

- **Debian-NP**: Debian GNU/Linux for Non-profit Organisations
- **Debian Accessibility Project**
- **Debian Enterprise**
- Other possible Custom Debian Distributions:
  - Debian-eGov, Office, Accounting, Geography (GIS), Biology, Physics, Mathematics

How?

- If you need special software, package it in **Debian**
- If you need special configurations, work with maintainers
- If you need stable software, fix bugs and submit patches to the BTS
- If you need translations, work with **Debian** translators
- If you need security, work with the **Debian** security team

Current Technology (pre cddtk)

- Metapackages:
  - Dependencies on other Debian packages (essential):
    - Use "Depends" for packages addressing an area of interest
    - Use "Recommends" for further interesting packages
    - Use "Suggests" for less important or non-free packages
  - Menu entries (recommended)
  - Configuration stuff (optional):
    - debconf questions (pre-seeding)
    - cfengine scripts
  - Special meta package: `<cdd>-common`

Open issues (1)

- Use of *Debian Package Tags* for CDDs (tags could be used instead of dependencies in metapackages, adds flexibility).
- Debconf preseeding support for all packages in a CDD (sometimes difficult, maintainers don't want to support special cases and there's no conscience about the importance of the CDD phenomenon).
- Standard way for modifying configurations after installation (cfengine, config4gnu, ...?).

Open issues (2)

- Standard way of generating CDD installation CDs (currently done in DebianEdu using d-i)
- Increase visibility of CDD in the standard Debian installation system
- Standard way of generating Live CD's of Custom Distributions: Knoppix, Metadistros, Casper...
- Assuming the current release cycle, add support for updating data for packages on stable (discover, l10n, ...)

Open issues (3)

- We need a new Debian Release System.
- Proposal 1:
  - Each CDD is a branch of the archive and has its own stable/testing package list and all packages are uploaded the same unstable pool; this needs:
    - Special promotion rules (to move packages inside each branch),
    - Reasonable subsets (CDDs or Core + Components),
    - BTS Support (for CDD specific bugs),
    - Security teams for each CDD.

Open issues (4)

- Proposal 2:
  - Find a common set of packages obtained from the *releasable set* of testing (i.e. no rc-bugs) and release stable CDDs from it.
- Proposal 3:
  - Testing proposed updates: some essential packages could be moved much faster to testing backporting from unstable on some architectures (when bugs are not present in those architectures).

More information about CDD

Mailing list

debian-custom@lists.debian.org

CDD Paper

`# apt-get install cdd-doc`

Wiki

http://wiki.debian.net/?CustomDebian

Alioth project

http://alioth.debian.org/projects/cdd/

What is the CDD Toolkit?

- A toolkit that tries to standardize the way developers define their CDD and provides tools to distribute, install, update and manage the customized system
- The basic idea behind the toolkit is to allow the distribution of CDD in *source* or in *binary* form using a single ``deb`` package that contains a description of the CDD and all the *metadata* needed to install or to generate an installer for it

What has to be done to build and maintain a CDD

- Package selection: definition of tasks and their package dependencies
- System configuration: pre-seeding for the initial installation and scripts to generate custom configuration files on installed systems
- Installation system: d-i support for CDD
- Distribution mainteinance: package updates on the archive, configuration scripts updates, bugtracking support, etc.

The CDD Description

- For this toolkit, a CDD is defined as a set of **tasks** that can be used to install and configure packages on the system for a given purpose
- A **task** definition has to include:
  - a list of packages or tasks that have to be installed to provide the *task's functionality*
  - the information needed to configure those packages: pre-seeding (useful for first installations) and scripts that generate customized configuration files

Toolkit components

- The **cddtool** program, used to parse the CDD description and generate the *binary description deb* and build CDD installers. This tool is currently written in **Python**
- Support scripts used to select, install, configure, upgrade and remove CDD tasks included on the *CDD binary description* packages currently installed on the system. The scripts are written in **shell** to avoid additional dependencies (CDD can be installed without a **Python** interpreter)

The cddtool program options

- **build**: Build metapackages, task lists, installation CD
- **get**: Download packages needed for the CDD
- **install**: Install packages related to a task
- **purge**: Purge packages installed for a task
- **reconfigure**: Reconfigure packages related to a task
- **remove**: Remove packages installed for a task
- **tdeps (*)**: Print task dependencies
- **tinfo (*)**: Print information about tasks
- **tmeta (*)**: Show metapackages' substvars
- **update**: Update packages related to a task

Runtime system

- **cddtk-apt**: script used to support the installation, upgrade and removal of cdd tasks using apt on a Debian system.
- **cddtk-di**: scripts used to support the installation of cdd tasks from the debian-installer
- **cddtk-bi**: scripts used to support the installation of cdd tasks from the base-installer system (this one is directly related to the debian-installer)
- **cddtk-divert, cddtk-cfg**: suport for cfg-scripts.

More information about CDDtk

Mailing list

debian-custom@lists.debian.org

SVN Repository

https://mixinet.net/svn/cddtk/trunk/

CDDtk packages and papers

http://people.debian.org/~sto/cddtk/

lliurex-cdd packages

apt-src http://lliurex.net/archive/ llx0509 lliurex

That's all, folks !