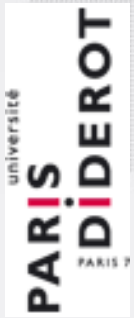
Maintaining Software Package Installations and Repositories for Free and Open Source Software Distributions

Ralf Treinen

PPS, Université Paris Diderot

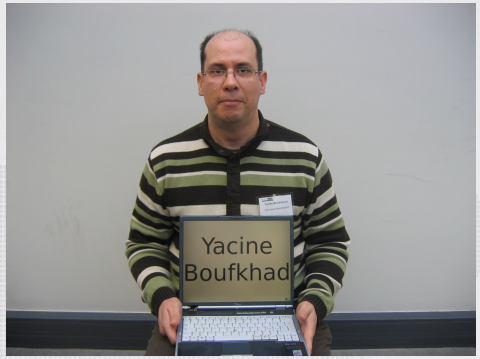


January 13, 2012

This is joint work with

Pietro Abate
IRILL

Jaap Boender
now U Bologna



Yacine Boufkhad
LIAFA, U Paris-7

Roberto Di Cosmo
IRILL

Jérôme Vouillon
IRILL

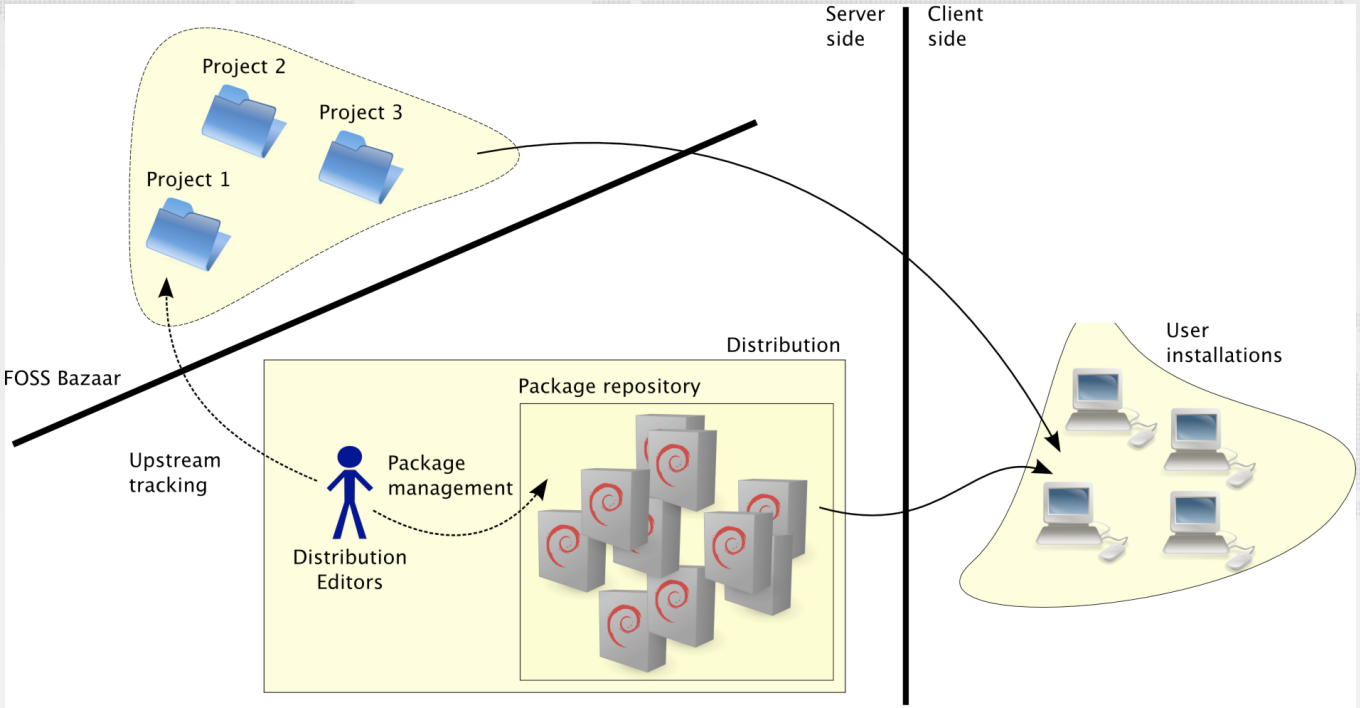
Stefano Zacchiroli
IRILL

Contents

The notion of distribution

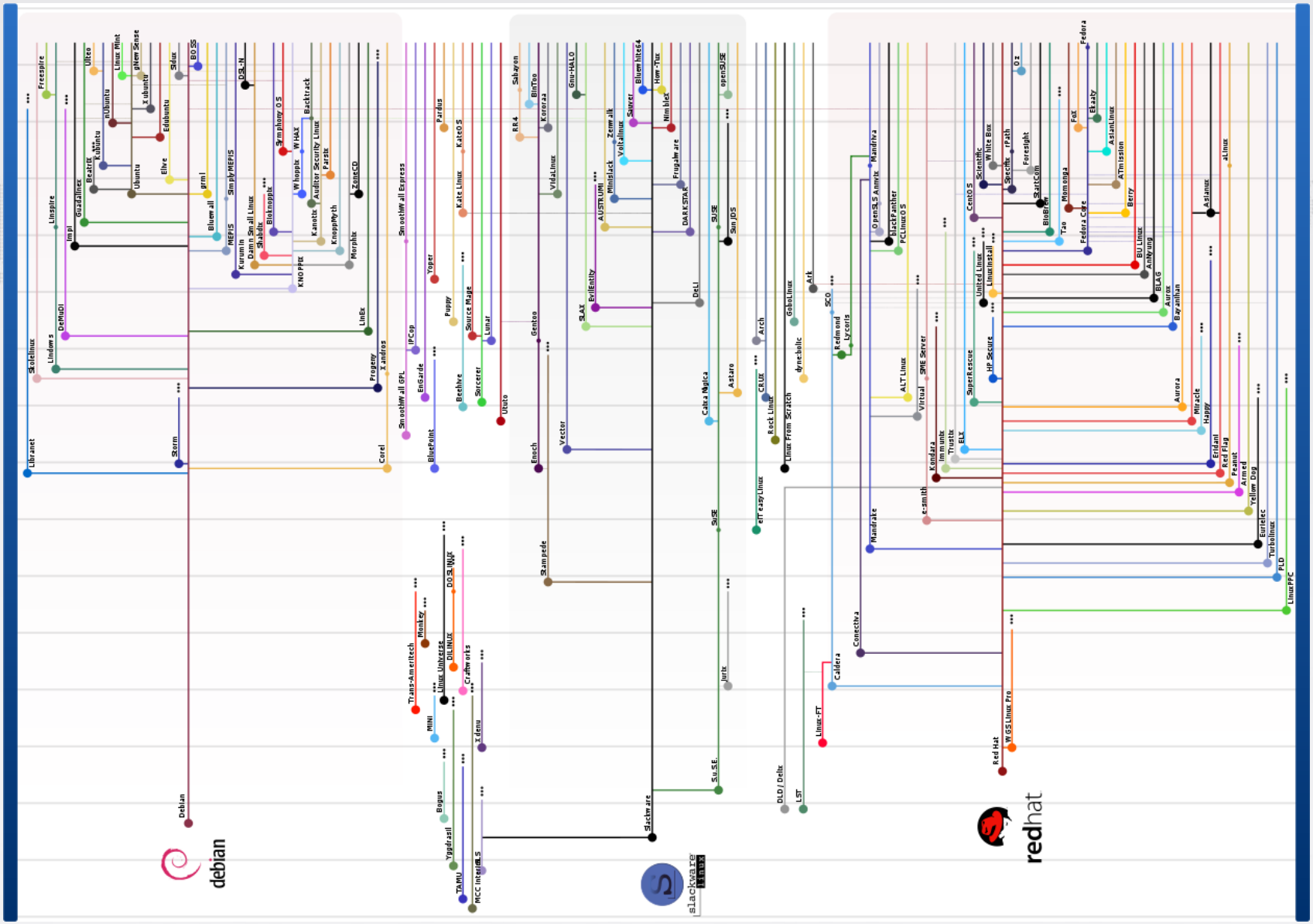
How do you compose a system by selecting components from tens of thousands developed independently?

A new idea from FOSS: distributions as *intermediaries* between developers and their users



Key notions: package & package managers

The FOSS Distribution Forest



FOSS is special:

1. *"reuse, reuse, reuse"* $\Rightarrow$
   - tons of small components
   - sharing: avoiding code duplication is importance for long-term maintenance
   - packages are heavily interconnected in various ways
2. *"release early, release often"* $\Rightarrow$
   - systems—as wholes—change on a daily basis
   - In debian, hundreds of packages change every day

Packages and their metadata

$$\text{Package} = \begin{cases} \text{some files} \\ \text{some scripts} \\ \text{metadata} \end{cases}$$

- Identification
- Inter-package rel.
  - Dependencies
  - Conflicts
- Feature declarations
- Other
  - Package maintainer
  - Textual descriptions

Example

```
Package: aterm
Version: 0.4.2-11
Depends: libc6 (>= 2.3.2.ds1-4),
  libice6 | xlibs (> 4.1.0), ...
Conflicts: suidmanager (< 0.50)
Provides: x-terminal-emulator, ...
Section: x11
Installed-Size: 280
Maintainer: Göran Weinholt ...
Architecture: i386
```

- a package is the *elemental component* of a distribution
- a working *system* is deployed by installing a package set (≈ 1000/2000 for GNU/Linux distributions)

A look at package installation

Phase	Trace
User request	`# apt-get install aterm`

Constraint resolution	`Reading package lists... Done`
	`Building dependency tree... Done`
	`The following extra packages will be installed:`
	`  libafterimage0`
	`The following NEW packages will be installed`
	`  aterm libafterimage0`
	`0 upgraded, 2 newly installed, 0 to remove and 1786 not upgraded.`
	`Need to get 386kB of archives.`
	`After unpacking 807kB of additional disk space will be used.`
	`Do you want to continue [Y/n]? Y`

Package retrieval	`Get: 1 http://debian.ens-cachan.fr testing/main libafterimage0 2.2.8-2 [301kB]`
	`Get: 2 http://debian.ens-cachan.fr testing/main aterm 1.0.1-4 [84.4kB]`
	`Fetched 386kB in 0s (410kB/s)`

Pre-configuration	`{`

Unpacking	`Selecting previously deselected package libafterimage0.`
	`(Reading database ... 294774 files and directories currently installed.)`
	`Unpacking libafterimage0 (from .../libafterimage0_2.2.8-2_i386.deb) ...`
	`Selecting previously deselected package aterm.`
	`Unpacking aterm (from .../aterm_1.0.1-4_i386.deb) ...`

Configuration	`Setting up libafterimage0 (2.2.8-2) ...` `Setting up aterm (1.0.1-4) ...`

Package installation — when things go wrong

```
# sudo apt-get install debhelper

Reading Package Lists... Done
Building Dependency Tree... Done
The following extra packages will be installed:
armagetron armagetron-common autoconf bonobo-activation codebreaker debconf debconf-i18n debconf-utils
dialog esound-common fb-music-high fontconfig frozen-bubble-data grepmail gv intltool-debian libaiksaurus-
libaiksaurus0c102 libatk1.0-0 libatk1.0-dev libbonobo-activation4 libbonobo2-0 libbonobo2-common libdb3
libdbd-mysql-perl libdbi-perl libeel2-data libesd0 ... xlibosmesa4 xlibs xlibs-data xpdf-common
The following packages will be REMOVED:
autoconf2.13 frozen-bubble frozen-bubble-lib gconf2 gnomemeeting itk3.1-dev libbonoboui2-0 libbonoboui2-c
libdigest-md5-perl libforms0.89 libgconf2-4 libgnome2-0 libgnome2-common libgnomeui-0 libgnomevfs2-0
libgnomevfs2-common libgtk1.2-dev libgtk2.0-0png3 libgtk2.0-dev libmime-base64-perl libpango1.0-dev
libsdl-mixer1.2-dev libsdl-perl libsdl-ttf1.2-dev libsdl1.2-dev libsmpeg-dev libstorable-perl nautilus
tk8.3-dev tktable-dev x-window-system x-window-system-core xaw3dg-dev xlib6g xlib6g-dev xlibmesa-dev
xlibmesa3 xlibosmesa3 xlibs-dev xlibs-pic xpdf xpdf-reader
The following NEW packages will be installed:
armagetron-common debconf-i18n fb-music-high fontconfig intltool-debian libaiksaurus-data libaiksaurus0c1
libeel2-data libfilehandle-unget-perl libfontconfig1 libforms1 libgdbm3 libgnutls7 libgsf-1 libice-dev
libice6 libidl0 liblzo1 libmagick5.5.7 libmail-mbox-messageparser-perl libmysqlclient12 libncursesw5
libnet-daemon-perl libnewt0.51 libpaper1 libplrpc-perl libsdl-console ... xlibmesa-gl xlibmesa-gl-dev
xlibs-data
75 packages upgraded, 80 newly installed, 42 to remove and 858 not upgraded.
Need to get 67.1MB of archives. After unpacking 26.9MB will be used.
Do you want to continue? [Y/n] Abort.
```

Research Directions

Research direction #1: better installation tools

Foster the development of better tools for the SysAdmin for the installation of upgrade of package-based FOSS systems.

Research direction #2: quality assurance

Urgent need of viable (semi-)automatic tools for QA of both *individual packages* and the *distribution as a whole*

Why do we find this interesting?

Challenge: Scale

- Number of packages in the debian development branch for linux on popular architectures : > 30.0000
- Number of architecture/OS pairs in debian : 14

Challenge: Fast moving

The development branch of a distributions is updated daily, or more often.

And last, not least . . .

It helps improve Free Software that we are using.

EDOS, Mancoosi, and Aeolus

- EDOS European project (Jan 2004 $\longrightarrow$ Jun 2007)
- Mancoosi: Managing the Complexity of the Open Source Infrastructure
- European Research Project in the 7th Framework
- Duration: Feb 2008 $\longrightarrow$ Mai 2011
- Ongoing: ANR (national) research projet Aeolus

Concrete view of packages

A package has prerequisites:

- System resources (disk space, ...)
- A certain version of a certain operating system
- File system structure (existence of, and access rights to certain directories)
- Availability of software libraries in a specific version
- Executability of other stand-alone tools

EDOS/Mancoosi: Abstract view of packages

A package contains *metadata*:

- A package provides a certain functionality that is denoted by the name of the package, probably refined by the version number.
- A package may also provide an even more abstract functionality (*feature*, *virtual package*), i.e. `web-browser`
- All prerequisites are expressed through relations to other packages (or virtual packages), or possibly other meta-data i.e. space consumption of the package.

Model (simplified)

Names, Versions and Constraints

- Set N of names
- Set V of versions: total and dense order
- Set $\textsc{Con}$ of constraints : $= v$, $> v$, $< v$, ... where $v \in \mathrm{V}$

A *package* (n, v, D, C) consists of

- a $\mathrm{package\ name}$ n,
- a *version* v,
- a set of dependencies $D \in \mathcal{P}(\mathcal{P}(\mathrm{N} \times \textsc{Con}))$,
- a set of conflicts $C \in \mathcal{P}(\mathrm{N} \times \textsc{Con})$,

A repository

is a set of packages, such that no two different packages carry the same name and version.

An R-installation

is a set $I \subseteq R$ with:

abundance For each element $d \in p.D$ there exists $(n, c) \in d$ and a package $q \in I$ such that $q.n = n$ and $p.v \in [[c]]$.

peace For each $(n, c) \in p.C$ and package $q \in I$, if $q.n = n$ then $q.v \notin [[c]]$.

flatness For all $p, q \in I$: if $p \neq q$ then $p.n \neq q.n$

Installability

$p \in R$ is R-installable if there exists an R-installation I with $p \in I$.

Example: Is *a* installable in *R*?

Repository *R*

```
Package: a
Version: 1
Depends: b, c|d

Package: b
Version: 17

Package: c
Version: 42
Conflicts: b > 15

Package: d
Version: 87
Depends: b < 20
```

Propositional Modeling without conflicts

Closed World Assumption

- (Package,version) = Propositional variable
  (package installed = value true)
- Complete installation = propositional model

Dependencies

- Modeling dependencies: $p \rightarrow \phi$ where ϕ is a positive formula
- Package p is not available: $\neg p$.
- Dependency theory D: *dual Horn theory*:
  Models are closed under union
- p is installable w.r.t. D : $D \wedge p$ satisfiable.
- Since D is dual Horn: p, q co-installable iff p installable and q installable (so far).

Propositional Modeling with conflicts

Conflicts

- A package p may be in conflict with several other packages $q_1, q_2, \ldots$.
- Conflict theory C: $\{\neg(p \wedge q_1), \neg(p \wedge q_2), \ldots\}$ (neither Horn nor dual Horn)
- p is installable: $p \wedge P \wedge C$ is satisfiable.

A result from EDOS [ASE 2006]

Installability of packages (measured in the number of packages) is NP-complete.

Limitation of this modelisation

- Closed world assumption: we assume complete knowledge of all packages.
- Does not capture the dynamic nature of distributions.

Special cases that are polynomial

- No conflicts (and installations may be non-flat):
  - Theory is dual Horn
  - Construction of a maximal model in PTime
- No disjunctions in dependencies (and no features, no multiple versions of packages)
  - all clauses are binary:

    $$a \rightarrow (b \wedge c \wedge d)$$

    is equivalent to

    $$\{a \rightarrow b, a \rightarrow c, a \rightarrow d\}$$

  - Construction of all consequences in PTime

EDOS-debcheck

- Written by Jérôme Vouillon in 2005, using SAT-solver technology
- Computes, for a complete distribution, *all* non-installable packages with explanation.
- And it does this in *a few seconds*.
- Integration into `pkglab`, an interactive system to explore package repositories of package-based software distributions.
- Paper at ASE 2006.

Co-installability

The Problem

- A set P of packages is co-installable (w.r.t. R) if there is an R-installation containing P.
- Generalisation of strong conflicts.
- There is a hugh number of co-installable sets.

Result

- Jérôme Vouillon and Roberto Di Cosmo. *On Software Component Co-Installability*. ESEC/FSE, September 2011.
- Drastical reduction of the problem by:
  - Elimination of irrelevant (for this problem) package relationships.
  - Collapsing the package graph by building a quotient.

Speculative analysis

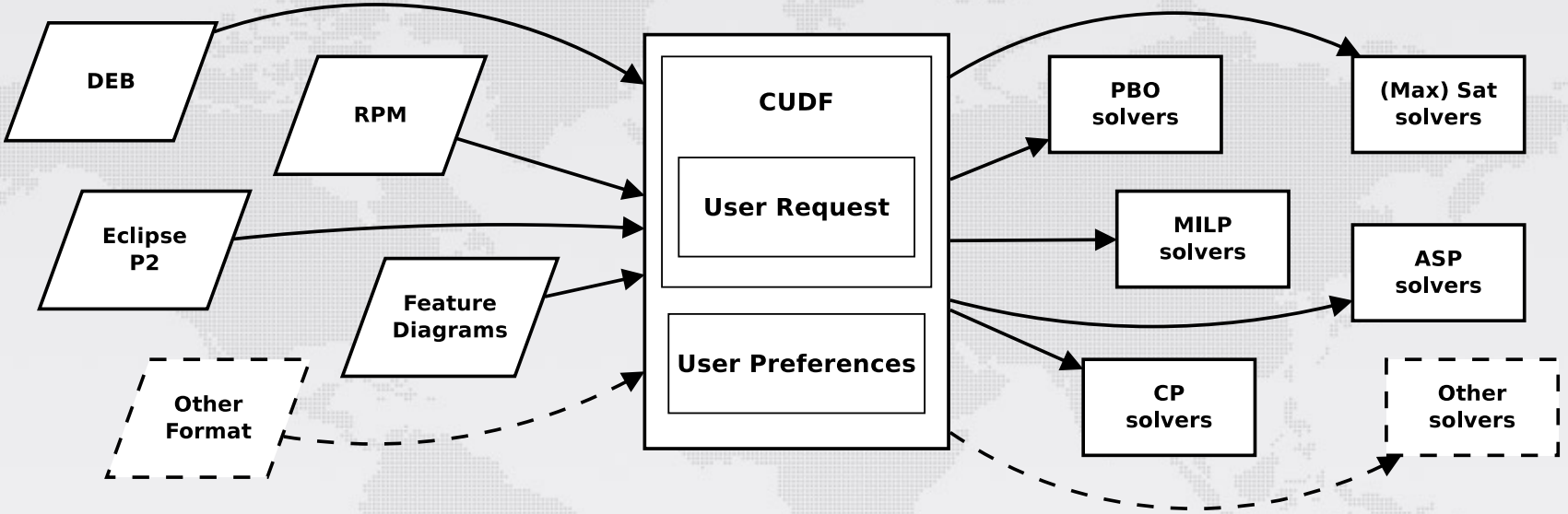
Finding outdated packages

- p is outdated if p is not installable, and it remains uninstallable how matter how the *other* packages evolve.
- Efficient calculation by reduction to the installability problem.

Approach

- Construct a (finite) repository containg representatives of relevant future versions of packages.
- Installations w.r.t. this repository are representative for installations in any possible future evolution.

The CUDF format



- CUDF as data interchange format
- Pietro Abate, Roberto Di Cosmo, Ralf Treinen, Stefano Zacchiroli. *MPM: A Modular Package Manager*, CBSE 2011.

MISC: The Mancoosi Solver Competition

What a good solver should be:

- Correct: do not propose a wrong solution
- Complete: find a solution if there exists one
- Flexible: allow for a rich language of user preferences
- Efficient

Organizing a Competition

- We (at IRILL) are no experts in building such solvers
- Our role is to foster the development of better solvers by organizing a yearly international competition.
- `www.mancoosi.org/misc`

MISC competitions

Past and Future Competitions

- Competitions associated with the LoCoCo workshop (see later)
- 1st MISC: FLoC 2010, Edinburgh, UK
- 2nd MISC: CP 2011, Perugia IT
- 3rd MISC, 2012: upcoming

Optimisation Criteria

- Several mesurements : number of removed packages, new packages, ...
- Lexicographic combinations of mesurements
- *Paranoid track* : as less changes as possible
- *Trendy track* : a system as up to date as possible
- *User track* : user-defined criteria

Towards a better apt

- Outcome of MISC: Currently three solvers of the MISC 2010 competition are packaged in debian.
- The experimental branch of apt is now containing an experimental interface to external CUDF solvers.
- Ongoing discussion with FOSS community about the evolution of the CUDF format.

Ongoing

- Better languages for describing user preferences for system upgrades
- Installation and upgrade problems not on a single machine but on a cluster (cloud) with shared ressources and services.
- Analyses of scripts that are executed at package installation: from the abstract view of packages to a concrete view.