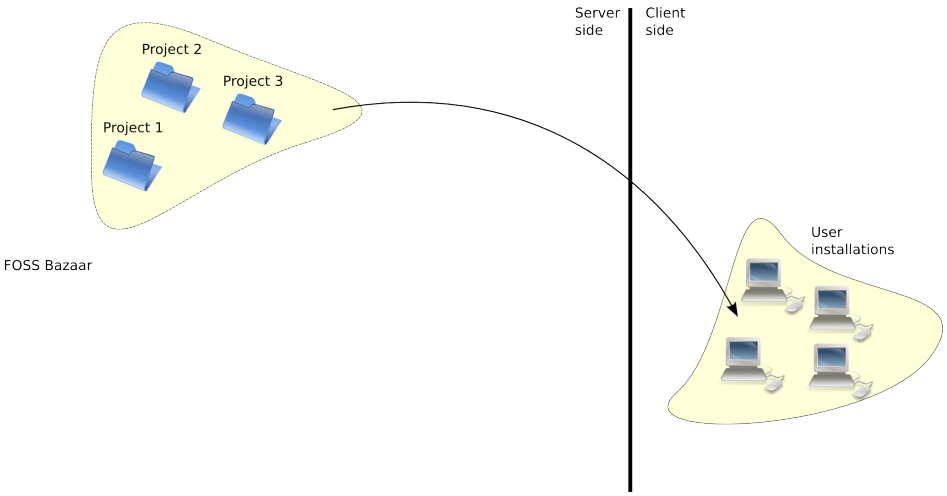
Avant l'arrivée des *distributions*, le seul moyen d'installer du logiciels sur les postes clients était:

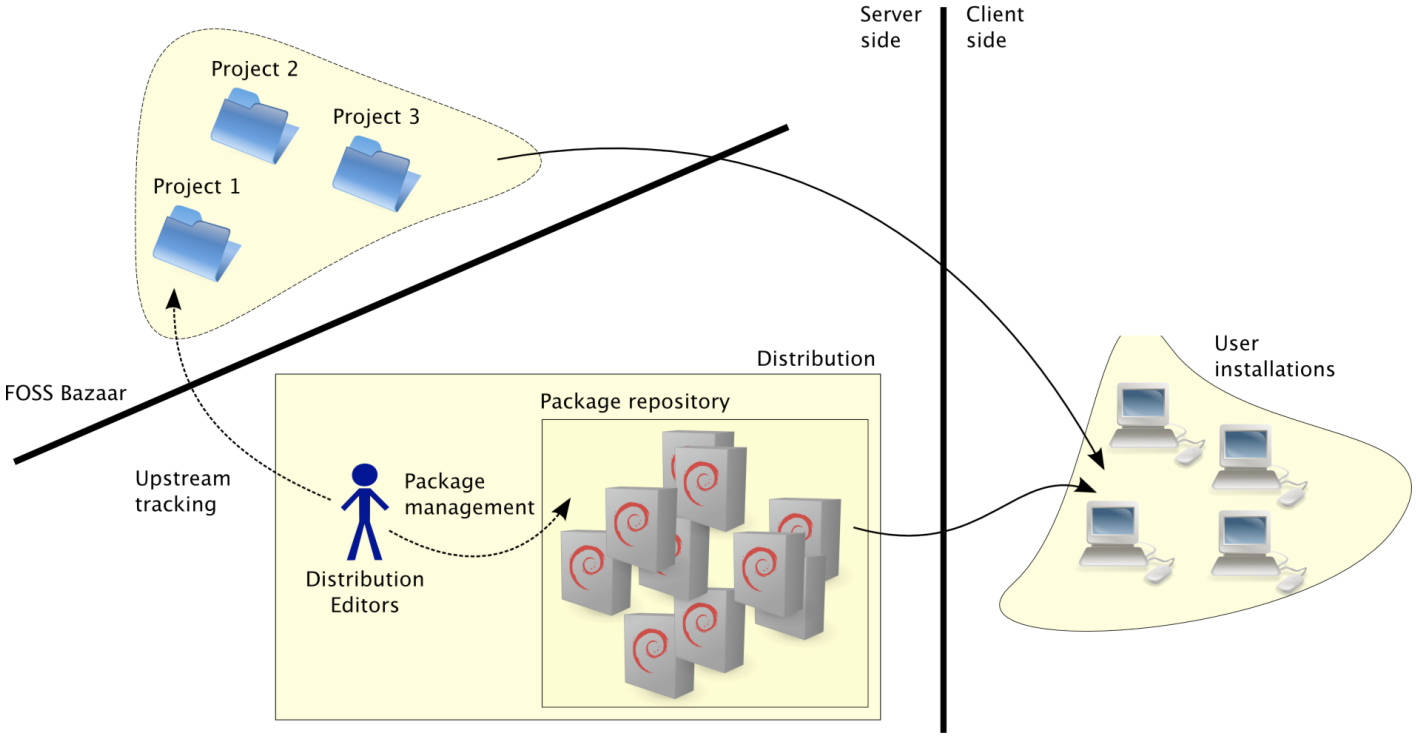


Mais:

- pas de *version standard* du poste client
- besoin de recompiler, . . . et reconfigure assez souvent
- trop compliqué!

Le cas des *distributions* Linux

Cela a fait naître les *distributions* comme intermédiaires entre les projets et les utilisateurs



La notion centrale est celle de *paquet*, avec ses outils d'administration. . .

Le role d'un éditeur de distribution:

upstream tracking : suivre l'évolution des sources
the developer is almost never the packager!

integration : offrir une collection coherente[33] de paquets
Pour cela, on doit traiter correctement des *dependances*

testing :

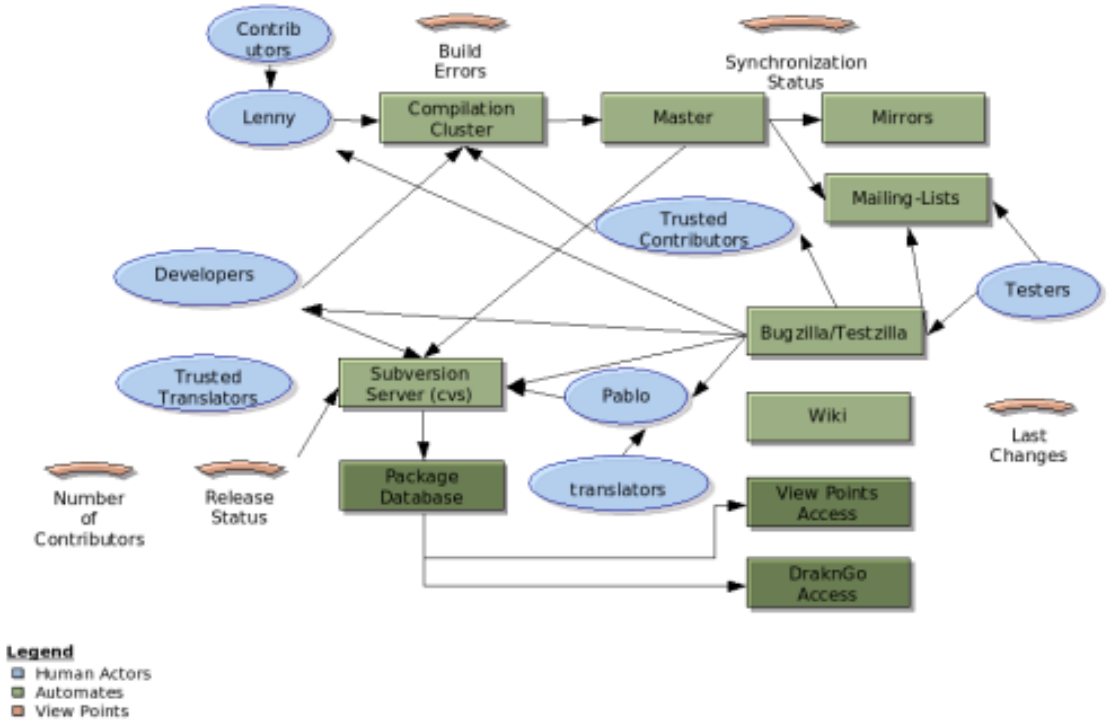
distribution : diffusion rapide sans casser l'éxistant

Ce n'est pas facile!:

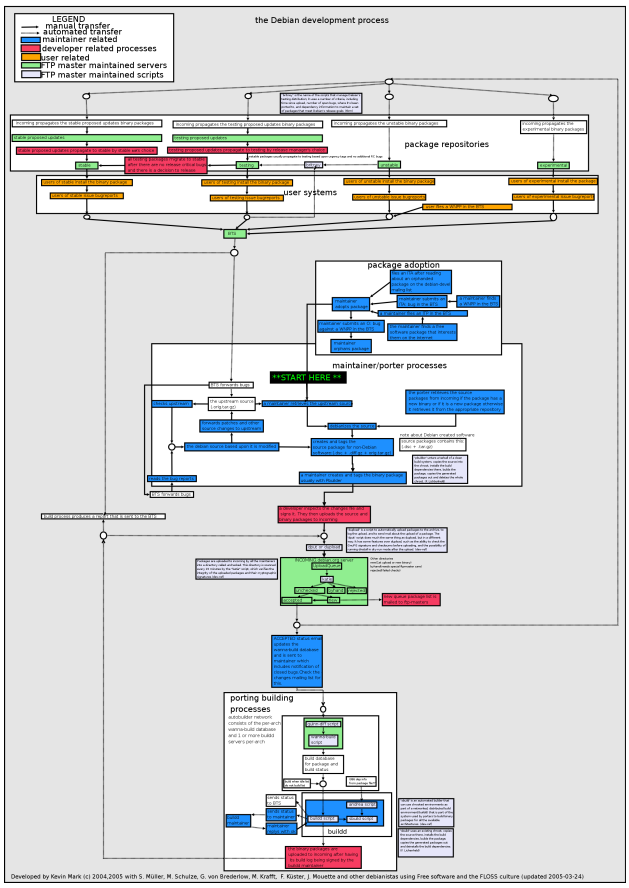
Le cycle de 6 mois de Mandriva nécéssite 30 années-homme.

[33] more formally?

An overview of Mandriva's lifecycle



An overview of Debians's lifecycle



Paquetages/composants: facteurs *industriels* cle

L'utilisation de composants, paquetages est nécessaire pour l'automatisation et du processus industriel:

- mesures
- test
- qualité
- configuration et customisation
- ...

C'est une des innovations majeurs des distributions GNU/Linux par rapport aux systèmes Unix traditionnels: il permet d'envisager *de la configuration flexible et la gestion* de sytèmes complexes, grace à la *reutilisabilité* et la *modularité*.
Faire cela bien, cela peut être très difficile.

Paquets, metadonnées, installations

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

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

Example

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

- le paquet est le *composant élémentaire* des systèmes de distributions modernes (pas spécifique à GNU/Linux)
- un *système* est obtenu en installant un ensemble de paquets ($\approx$ 1000/2000 pour une distribution GNU/Linux)

Exemple d'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) ...`

- *chaque* phase peut échouer (cela arrive souvent . . .)
- on doit essayer de capturer les erreurs le plus tôt possible

Une installation est quelque chose de compliqué

Dans les distributions d'aujourd'hui:

1. le problème de savoir si une installation termine avec succès est indecidable en général : les scripts de configuration sont Turing complets. . . *aujourd'hui*
2. aucune garantie que les composants dans une distribution (DVD ou en ligne) contiennent seulement paquets installables
3. beaucoup d'erreurs rencontrés par les utilisateurs sont liès aux dependences
   - plus de 20,000 paquets (et 10,000 sources) dans Debian
   - plus de 200.000 interdependences

   et cela grandit chaque jour!

Paquets dans une distribution Linux/*BSD:

Mandriva	$\approx$ 9.000
*BSD	$\approx$ 10.000
Debian	$\approx$ 20.000
	. . .

Changements journaliers, et pourtant . . . tout "doit fonctionner"

Outils pour l'analyse des dependances

coherence : la distribution *ne doit pas* contenir

dangling dependencies : packages depending on non-existent packages

unsatisfiable dependencies : conflicts between packages that are in the transitive closure of the dependencies of a same package

Facile?

Pas du tout: *versions*, *virtual packages*, *exclusive-or constraints*, *conflict constraints*, etc.

Coherence en pratique

Installation d'une Mandrake Community edition, version 10.1, à partir du DVD

```
              libc.so.6(GLIBC_2.0) is needed by libreadline4-4.3-7mdk
              libc.so.6(GLIBC_2.1.3) is needed by libreadline4-4.3-7mdk
              libc.so.6(GLIBC_2.3) is needed by libreadline4-4.3-7mdk
* rpm transactions start
* getFile libreadline4-4.3-7mdk.i586.rpm:Installation DVD (1:cdrom)
* retrying installing package libreadline4-4.3-7mdk.i586 alone in a transaction
* opened rpmdb for writing in /mnt
* opened rpm database for retry transaction of 1 package only
* check failed : libc.so.6 is needed by libreadline4-4.3-7mdk
              libc.so.6(GLIBC_2.0) is needed by libreadline4-4.3-7mdk
              libc.so.6(GLIBC_2.1.3) is needed by libreadline4-4.3-7mdk
              libc.so.6(GLIBC_2.3) is needed by libreadline4-4.3-7mdk
* rpm transactions start
* getFile libreadline4-4.3-7mdk.i586.rpm:Installation DVD (1:cdrom)
* retrying installing package libreadline4-4.3-7mdk.i586 alone in a transaction
* opened rpmdb for writing in /mnt
* opened rpm database for retry transaction of 1 package only
* check failed : libc.so.6 is needed by libreadline4-4.3-7mdk
              libc.so.6(GLIBC_2.0) is needed by libreadline4-4.3-7mdk
              libc.so.6(GLIBC_2.1.3) is needed by libreadline4-4.3-7mdk
              libc.so.6(GLIBC_2.3) is needed by libreadline4-4.3-7mdk
* rpm transactions start
* getFile libreadline4-4.3-7mdk.i586.rpm:Installation DVD (1:cdrom)
* bad package libreadline4-4.3-7mdk.i586 unable to be installed
_
```

Metadata in common *binary* package formats

dependencies package A needs another package B to work properly.

conflicts package A that cannot be installed when package B is.

virtual packages and provides several packages can say they provide a "virtual package"; other packages can depend on the virtual packages (ex: web browser, mta...).

versioned dependencies and conflicts dependencies or conflicts can mention package versions.

complex boolean dependencies package A can depend on package B AND (package C OR package D).

feature dependencies a package can require some other package - any other package - providing feature F (ex: need file /bin/sh).

grouping package can be assigned to a group (eg. web browsers).

recommendations package A will almost always need package B.

suggestions package A may sometimes work better if package B is installed.

Un exemple

```
Package: binutils
Priority: standard
Section: devel
Installed-Size: 5976
Maintainer: James Troup <james@nocrew.org>
Architecture: i386
Version: 2.15-6
Provides: elf-binutils
Depends: libc6 (>= 2.3.2.ds1-21)
Suggests: binutils-doc (= 2.15-6)
Conflicts: gas, elf-binutils, modutils (<< 2.4.19-1)
Filename: pool/main/b/binutils/binutils_2.15-6_i386.deb
Size: 2221396
MD5sum: e76056eb0d6a0f14bc267bd7d0f628a5
Description: The GNU assembler, linker and binary utilities
The programs in this package are used to assemble, link and ma
binary and object files. They may be used in conjunction with
```

WorkPackage 2 in the EDOS Project

Our goal:

"to build new generation tools for managing large sets[34] *of software packages, using formal methods"*

Our focus:

check *distribution repositories*, not *user installations*

Why this focus:

an *essential* problem, yet an *invisible* one

[34] like those found in free software distributions

First phase: checking packagewise installability

We focused on the *package installation problem*

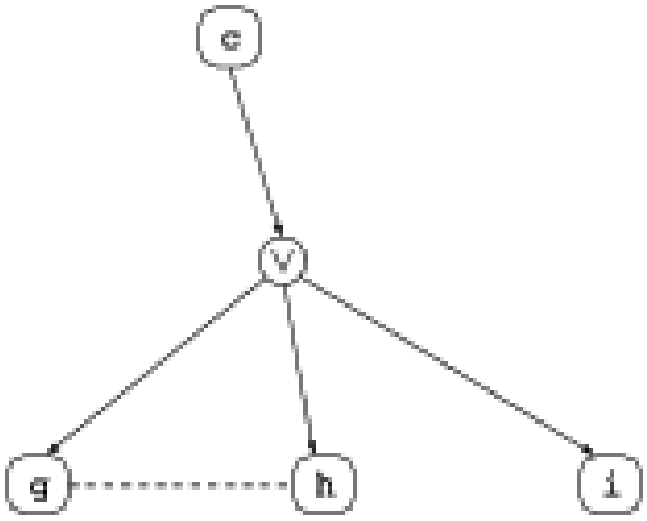
"given a repository R, can I install a package P =(u,v)?"

Solving this problem is central to:

- analyse a repository
- allow distribution maintainers to discover early problemsdue to the changes in the package versions

The package installability problem is complex

- Packages may *require* other packages,
- Packages may *disjunctively* require other packages,
- Packages may *conflict* with each other.



Can we use the existing depsolvers?

- APT: incomplete, surprising behavior
  - `apt-get install abiword-gnome=2.2.7-3` fails
  - `apt-get install abiword-gnome=2.2.7-3 abiword-common=2.2.7-3` succeeds
  - `apt-get install abiword-common=2.2.7-3 abiword-gnome=2.2.7-3` succeeds, but install one package more!
- URPMI:incomplete, good heuristics;Portage: incomplete,
- Smart: complete?,non optimal, explosive!

Can we use the existing depsolvers?

```
/usr/bin/time smart --data-dir=/ext/tmp/smart install --urls php3
Loading cache...
Updating cache... ######################################## [100%]
Computing transaction...
http://www.pps.jussieu.fr/ boender/pool/main/libg/libgcrypt11/libgcryp
http://www.pps.jussieu.fr/ boender/pool/main/a/apache/apache-common_1.
http://www.pps.jussieu.fr/ boender/pool/main/o/openssl/libssl0.9.7_0.9
...

23576.80user[35] 8.71system 13:09:13elapsed 49%CPU (0avgtext+0avgdata 0m
0inputs+0outputs (822major+36209minor)pagefaults 0swaps
```

(see EDOS WP2 Deliverable 2.2 for detailed information).
None of these tools can be used to check for installability all of the 40.000+ packages in Debian!
No criticsm implied: these tools try to do *more* than checking installability.

[35] 6h30

Package installation as boolean constraint solving

- DEB and RPM both use unary constraints
  - u meaning "any version of unit u" [36]
  - u *op const* with *op* being =,>>,<<,>=,=< meaning "any version v of unit u such that v *op const* is true".

  they can be encoded as boolean constraints
  - For each package (u, v) in the repository R, add the variable I_u^v
  - Convert unary constraint into a disjunction of boolean constraints as:
    - ★ u becomes $I_u^{v_1} \vee \ldots \vee I_u^{v_k}$ [37]
    - ★ u *op const* becomes $I_u^{v_1} \vee \ldots \vee I_u^{v_k}$ [38]

[36] can be seen as an abbreviation for $u > 0$
[37] $v_1, \ldots, v_k$ are the available versions of P in the repository
[38] $v_1, \ldots, v_k$ are the available versions of u in the repository that satisfy v_i *op const*

Package installation as boolean constraint solving

- dependencies and conflict relations become logical implications
  If R contains a dependency for (u, v) of the form

$$\begin{aligned} &\mathit{Depends} : (u_1^1\ op_1^1\ v_1^1 \vee \cdots \vee u_1^{r_1}\ op_1^{r_1}\ v_1^{r_1}) \\ &\wedge \cdots \wedge (u_s^1\ op_s^1\ v_s^1 \vee \cdots \vee u_s^{r_s}\ op_s^{r_s}\ v_s^{r_s}). \end{aligned}$$

  we introduce the formula

$$\begin{aligned} &I_u^v \Rightarrow (\bigvee_{w\ op_1^1 v_1^1} I_{u_1}^w \vee \cdots \vee \bigvee_{w\ op_1^{r_1} v_1^{r_1}} I_{u_{r_1}}^w) \\ &\wedge \ldots \wedge (\bigvee_{w\ op_s^1 v_s^1} I_{u_s}^w \vee \cdots \vee \bigvee_{w\ op_s^{r_s} v_s^{r_s}} I_{u_{r_s}}^w) \end{aligned}$$

  If R contains a conflict for (u, v) of the form

$$\begin{aligned} &\mathit{Conflicts} : (u_1^1\ op_1^1\ v_1^1 \vee \cdots \vee u_1^{r_1}\ op_1^{r_1}\ v_1^{r_1}) \\ &\wedge \cdots \wedge (u_s^1\ op_s^1\ v_s^1 \vee \cdots \vee u_s^{r_s}\ op_s^{r_s}\ v_s^{r_s}). \end{aligned}$$

  we introduce the formula

$$\begin{aligned} &I_u^v \Rightarrow (\bigwedge_{w\ op_1^1 v_1^1} \neg I_{u_1}^w \vee \cdots \vee \bigwedge_{w\ op_1^{r_1} v_1^{r_1}} \neg I_{u_{r_1}}^w) \\ &\wedge \ldots \wedge (\bigwedge_{w\ op_s^1 v_s^1} \neg I_{u_s}^w \vee \cdots \vee \bigwedge_{w\ op_s^{r_s} v_s^{r_s}} \neg I_{u_{r_s}}^w) \end{aligned}$$

Installation as boolean constraint solving: an example

- a repository becomes the conjunction of the dependency and conflict relations
- for Debian repositories, we need also to model the fact that only one version of a unit u can be installed at a time:

$$\bigwedge_{\substack{v_1, v_2 \in R_u \\ v_1 \neq v_2}} \neg(I_u^{v_1} \wedge I_u^{v_2})$$

Installation as boolean constraint solving: an example

```
Package: libc6
Version: 2.2.5-11.8

Package: libc6
Version: 2.3.5-3

Package: libc6
Version: 2.3.2.ds1-22
Depends: libdb1-compat

Package: libdb1-compat
Version: 2.1.3-8
Depends: libc6 (>=
2.3.5-1)

Package: libdb1-compat
Version: 2.1.3-7
Depends: libc6 (>=
2.2.5-13)
```

becomes

$$\neg(I^{2.3.2.ds1-22}_{\texttt{libc6}} \wedge I^{2.2.5-11.8}_{\texttt{libc6}})$$
$$\wedge$$
$$\neg(I^{2.3.2.ds1-22}_{\texttt{libc6}} \wedge I^{2.3.5-3}_{\texttt{libc6}})$$
$$\wedge$$
$$\neg(I^{2.3.5-3}_{\texttt{libc6}} \wedge I^{2.2.5-11.8}_{\texttt{libc6}})$$
$$\wedge$$
$$\neg(I^{2.1.3-7}_{\texttt{libdb1-compat}} \wedge I^{2.1.3-8}_{\texttt{libdb1-compat}})$$
$$\wedge$$
$$I^{2.3.2.ds1-22}_{\texttt{libc6}} \rightarrow (I^{2.1.3-7}_{\texttt{libdb1-compat}} \vee I^{2.1.3-8}_{\texttt{libdb1-compat}})$$
$$\wedge$$
$$I^{2.1.3-7}_{\texttt{libdb1-compat}} \rightarrow (I^{2.3.2.ds1-22}_{\texttt{libc6}} \vee I^{2.3.5-3}_{\texttt{libc6}})$$
$$\wedge$$
$$I^{2.1.3-8}_{\texttt{libdb1-compat}} \rightarrow I^{2.3.5-3}_{\texttt{libc6}}$$

Installation as boolean constraint solving: end

Now, checking whether a particular version v of a unit u is installable boils down to finding a boolean assignment that makes v_u true, and satisfies the encoding of the repository.

Installation as boolean constraint solving: end

In our example, to test installability of `libc6` version `2.3.2.ds1-22` we get the *equivalent*[39] SAT problem

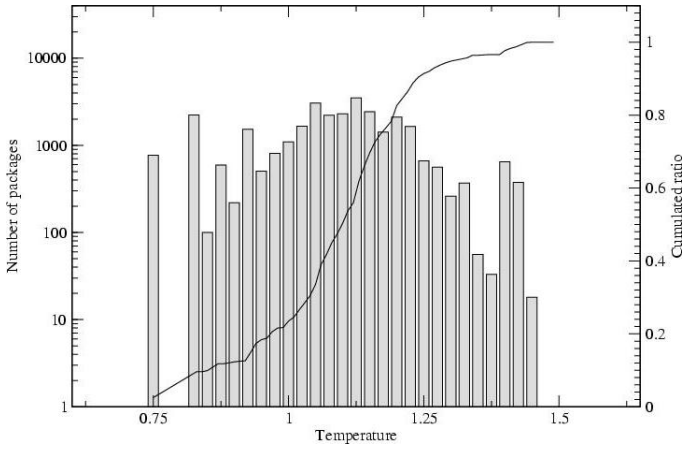
$$
\begin{aligned}
& I^{2.3.2.ds1-22}_{\texttt{libc6}} \\
& \wedge \\
& \neg(I^{2.3.2.ds1-22}_{\texttt{libc6}} \wedge I^{2.2.5-11.8}_{\texttt{libc6}}) \\
& \wedge \\
& \neg(I^{2.3.2.ds1-22}_{\texttt{libc6}} \wedge I^{2.3.5-3}_{\texttt{libc6}}) \\
& \wedge \\
& \neg(I^{2.3.5-3}_{\texttt{libc6}} \wedge I^{2.2.5-11.8}_{\texttt{libc6}}) \\
& \wedge \\
& \neg(I^{2.1.3-7}_{\texttt{libdb1-compat}} \wedge I^{2.1.3-8}_{\texttt{libdb1-compat}}) \\
& \wedge \\
& I^{2.3.2.ds1-22}_{\texttt{libc6}} \longrightarrow \\
& (I^{2.1.3-7}_{\texttt{libdb1-compat}} \vee I^{2.1.3-8}_{\texttt{libdb1-compat}}) \\
& \wedge \\
& I^{2.1.3-7}_{\texttt{libdb1-compat}} \longrightarrow \\
& (I^{2.3.2.ds1-22}_{\texttt{libc6}} \vee I^{2.3.5-3}_{\texttt{libc6}}) \\
& \wedge \\
& I^{2.1.3-8}_{\texttt{libdb1-compat}} \longrightarrow I^{2.3.5-3}_{\texttt{libc6}}
\end{aligned}
$$

i.e.[40]

```
p cnf 5 8
4 0
1 2 -4 0
-4 -5 0
-3 -5 0
-3 -4 0
-2 3 0
-1 3 4 0
-1 -2 0
```


Practical results

- The resulting formulas can be large (median formula size 400 litterals); luckily, their SAT-temperature is low.



- Some formulas can be harder[41].
- A serious SAT-solver is required.

This is incorporated in the EDOS *debcheck/rpmcheck* tool.

[41] the formula for `kde 5.45` has 32,000 litterals

Installation is NP-complete!

We just need NP-hardness: we reduce 3SAT to the Debian package installation problem.
Let $S = C_1 \wedge \ldots \wedge C_n$ be a 3SAT instance:

- $C_i = (l_{i,1} \vee l_{i,2} \vee l_{i,3})$ with $l_{i,k}$ literals[42].
- $A = \{a_1, \ldots, a_k\}$ be the set of atoms occurring in S.

Build the Debian repository R_S, with packages P_S[43], P_{C_i} for each clause C_i, and V_a, P_a, and $P_{\overline{a}}$ for each atom a, with constraints

1. P_S depends $P_{C_1}, \ldots, P_{C_n}, V_{a_1}, \ldots, V_{a_k}$
2. P_{C_i} depends $P_{\ell_{i,1}} | P_{\ell_{i,2}} | P_{\ell_{i,3}}$, for each $1 \leq i \leq n$
3. V_a depends $P_a | P_{\overline{a}}$ for each atom a
4. P_a conflicts $P_{\overline{a}}$ for each atom a
5. $P_{\overline{a}}$ conflicts P_a for each atom a

[42] either a propositional atom a or a negated propositional atom $\overline{a}$
[43] representing S itself

EDOS WP2: practical results

toolchain

Ceve : generic parser/converter
EdosLib : package metadata manipulation (cone extract, etc.)
Solver/Naive/Oz : interface to 3 different solvers

integrated tool

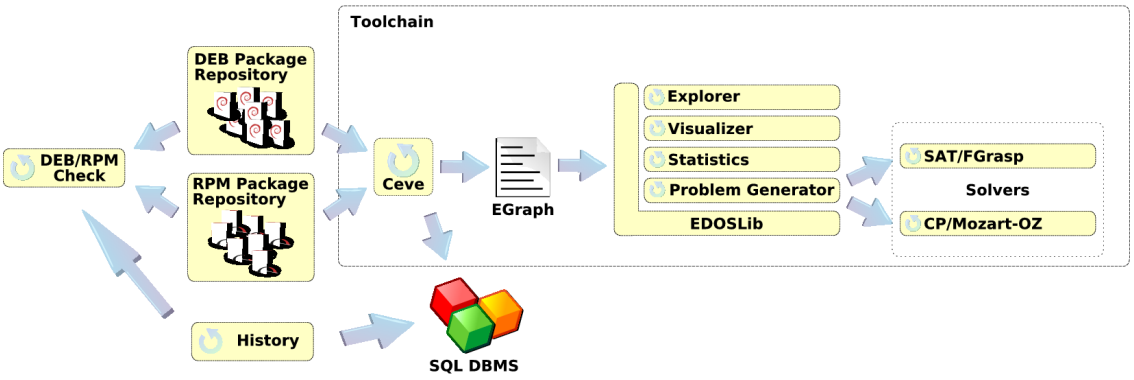
debcheck/rpmcheck : *fast* analysis, and *concise explanation* of errors

timeline

history, anla : timeline exploration of package bases, integrating the solver from debcheck/rpmcheck

All the software is available under subversion from `http://www.edos-project.org`, under GPL or LGPL licence.
Analysis of the full Debian Pool (over 39000 packages) can be performed on this laptop in 3[44] minutes!

[44] not a typo! it is really three



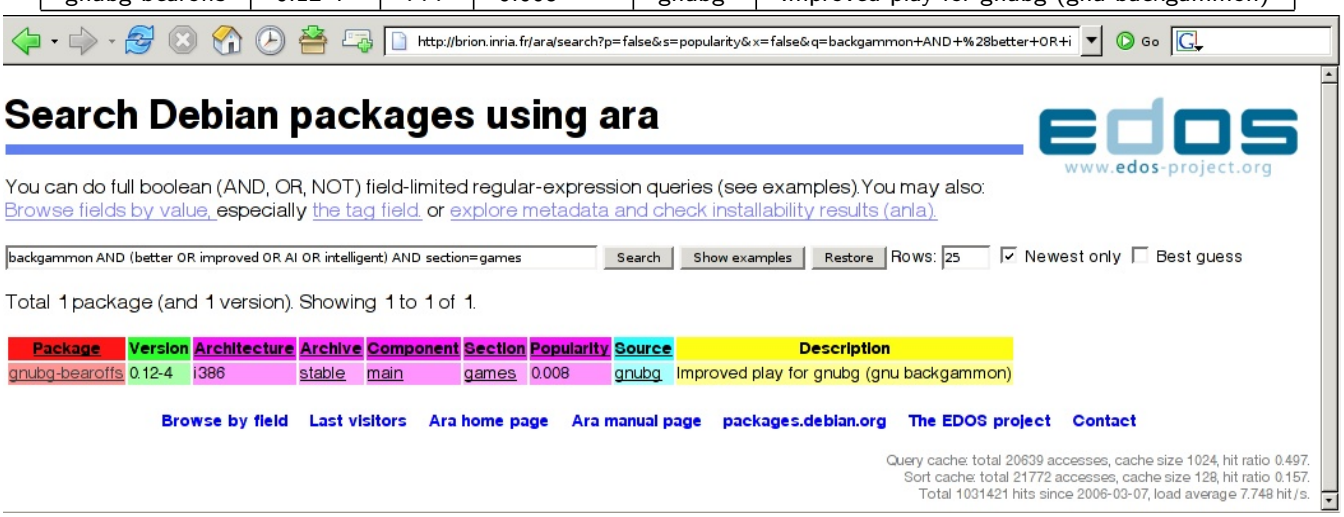
time for a first demo!

Example usage of the EDOS tools

We are playing backgammon but we are unsatisfied by the game performance. We want to know if there is a package with better gameplay. We do a search using `ara.edos-project.org` :

```
backgammon AND (better OR improved OR AI OR intelligent)
AND section=games
```

Package	Version	. . .	Popularity	Source	Description
gnubg-bearoffs	0.12-4	. . .	0.008	gnubg	Improved play for gnubg (gnu backgammon)



Nobody's perfect

Let's install gnubg-bearoffs.

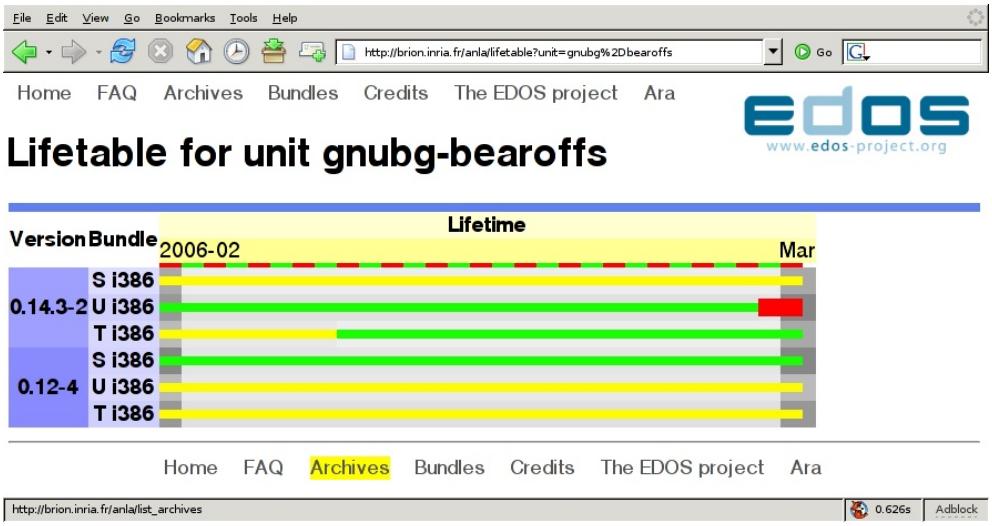
```
% sudo apt-get install gnubg-bearoffs
Reading package lists...  Done
Building dependency tree...  Done
Some packages could not be installed.  This may mean
that you have requested an impossible situation or if
you are using the unstable distribution that some
required packages have not yet been created or been
moved out of Incoming.
Since you only requested a single operation it is
extremely likely that the package is simply not
installable and a bug report against that package should
be filed.
The following information may help to resolve the
situation:
The following packages have unmet dependencies.
gnubg-bearoffs:  Depends:  gnubg but it is not going to
be installed
E: Broken packages
```


Why is gnubg-bearoffs not installable ?

1. First check Debian's Quality Assurance pages.
2. At http://packages.qa.debian.org/g/gnubg.html we see zero reported bugs.
3. At "debcheck" (http://qa.debian.org/debcheck.php): no gnubg-* package is listed.
4. Searching on gnubg or backgammon on the Debian lists yields nothing.
5. Searching the web yields the maintainers' blog entry stating that it is indeed broken.

anla, a Debian exploration web interface

- Integrates our dependency solver debcheck. It therefore finds **all** dependency and conflict-related installability problems.
- Knows of the historical evolution of package metadata.
- Can explain why packages are uninstallable in a given archive at a given date.

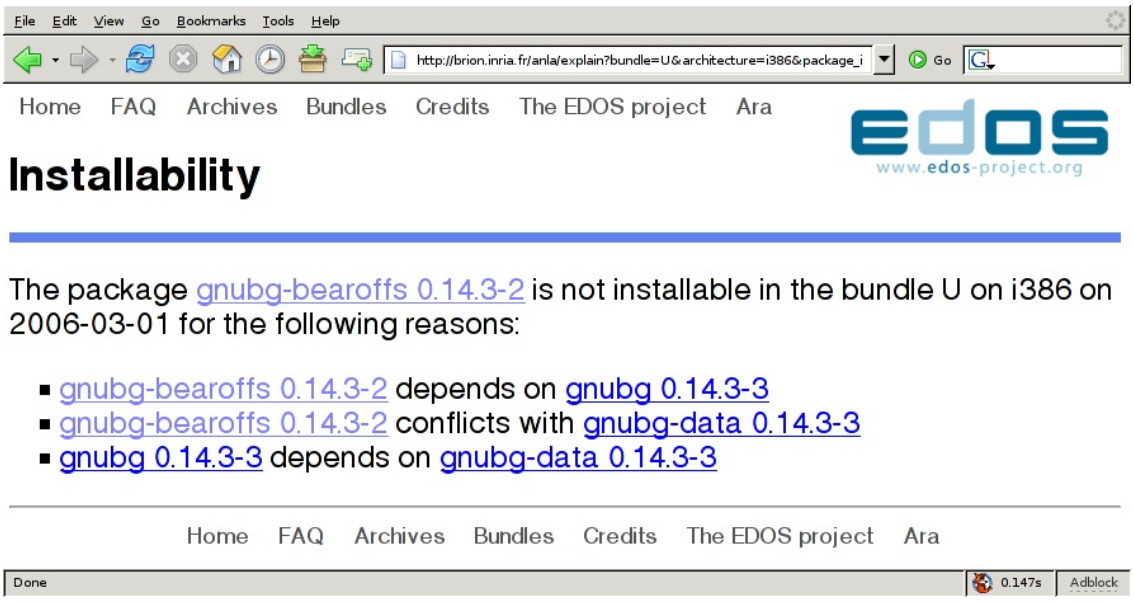


Diagnosing gnubg-bearoffs

Output of anla:

The package gnubg-bearoffs 0.14.3-2 is not installable in the bundle U on i386 on 2006-03-01 for the following reasons:

- *gnubg-bearoffs 0.14.3-2 depends on gnubg 0.14.3-3*
- *gnubg-bearoffs 0.14.3-2 conflicts with gnubg-data 0.14.3-3*
- *gnubg 0.14.3-3 depends on gnubg-data 0.14.3-3*



Mais l'upgrade aussi pose problème

upward closure :

si on part d'une configuration coherente C, pour tout ensemble de nouveaux paquets N, on peut arriver à une configuration coherente CN avec les fonctionnalités de C et N.
La mise à jour de C à CN peut seulement mettre à jour des paquets, ou remplacer des paquets avec d'autres qui les remplacent[45].

[45] plue formellement?

Upgrade en pratique

```
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-data libaiks
libatk1.0-0 libatk1.0-dev libbonobo-activation4 libbonobo2-0 libbonobo2-common libdb3 libdbd-mysql-perl libdbi-
libeel2-data libesd0 libfilehandle-unget-perl libfontconfig1 libforms1 libfreetype6 libfreetype6-dev libgcc1
...perl perl-base perl-modules perlmagick pkg-config pm-dev po-debconf render-dev tcl8.4 tcl8.4-dev tktable tra
ucf whiptail x-dev xaw3dg xbase-clients xfig xfree86-common xlibmesa-dri xlibmesa-gl xlibmesa-gl-dev xlibosmesa
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-common
libdigest-md5-perl libforms0.89 libgconf2-4 libgnome2-0 libgnome2-common libgnomeui-0 libgnomevfs2-0 libgnomevf
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 libaiksaurus0c102 lib
libfilehandle-unget-perl libfontconfig1 libforms1 libgdbm3 libgnutls7 libgsf-1 libice-dev libice6 libidl0 liblz
libmagick5.5.7 libmail-mbox-messageparser-perl libmysqlclient12 libncursesw5 libnet-daemon-perl libnewt0.51 lib
libplrpc-perl libsdl-console libsdl-gfx1.2 libsdl-ttf2.0-0 libsm-dev libsm6 libssl0.9.7 libstartup-notification
libt1-5 libtext-charwidth-perl libtext-wrapi18n-perl libtiff-tools libwmf0.2-7 libx11-6 libx11-dev libxcursor1
libxext-dev libxext6 libxft1 libxft2 libxi-dev libxi6 libxmu-dev libxmu6 libxmuu-dev libxmuu1 libxp-dev libxp6
libxpm-dev libxpm4 libxrandr-dev libxrandr2 libxrender-dev libxrender1 libxt-dev libxt6 libxtrap-dev libxtrap6
libxtst-dev libxtst6 libxv-dev libxv1 lyx-common lyx-xforms pm-dev po-debconf render-dev tcl8.4 tcl8.4-dev ucf
x-dev xlibmesa-dri 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]n
Abort.
```

The Mancoosi project [http://www.mancoosi.org]

project strep FP7 (2008–2011)

focus *coherence* and maintenance of a FOSS distribution installation (user point of view)

- Tools for user-side package management (package managers / meta-installer)
  - installation
  - removal
  - upgrade
  - downgrade
  - ...

Goals, approach and strategy

Mancoosi's goal:

> *enable safe, efficient maintainability of the Open Source software infrastructure built out of software packages*

A twofold *approach* to reach this objective:

- design algorithms and tools to tackle the upgradeability problem; this allows to find good upgrade paths w.r.t. the user's needs.
- conceive and implement a transactional update process; this allows to *roll-back* an unsatisfactory upgrade.

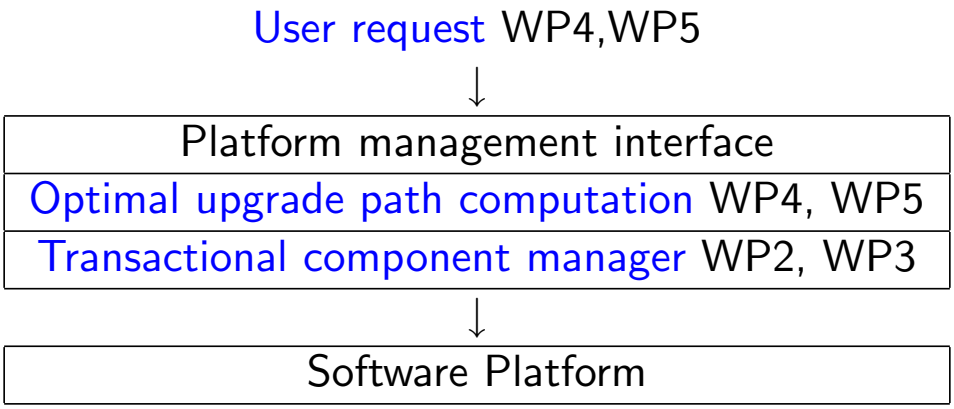
A long term, *ecosystem* strategy to reach this objective:

- bring users, distributions, developers and researchers together!
- set up virtous cycles

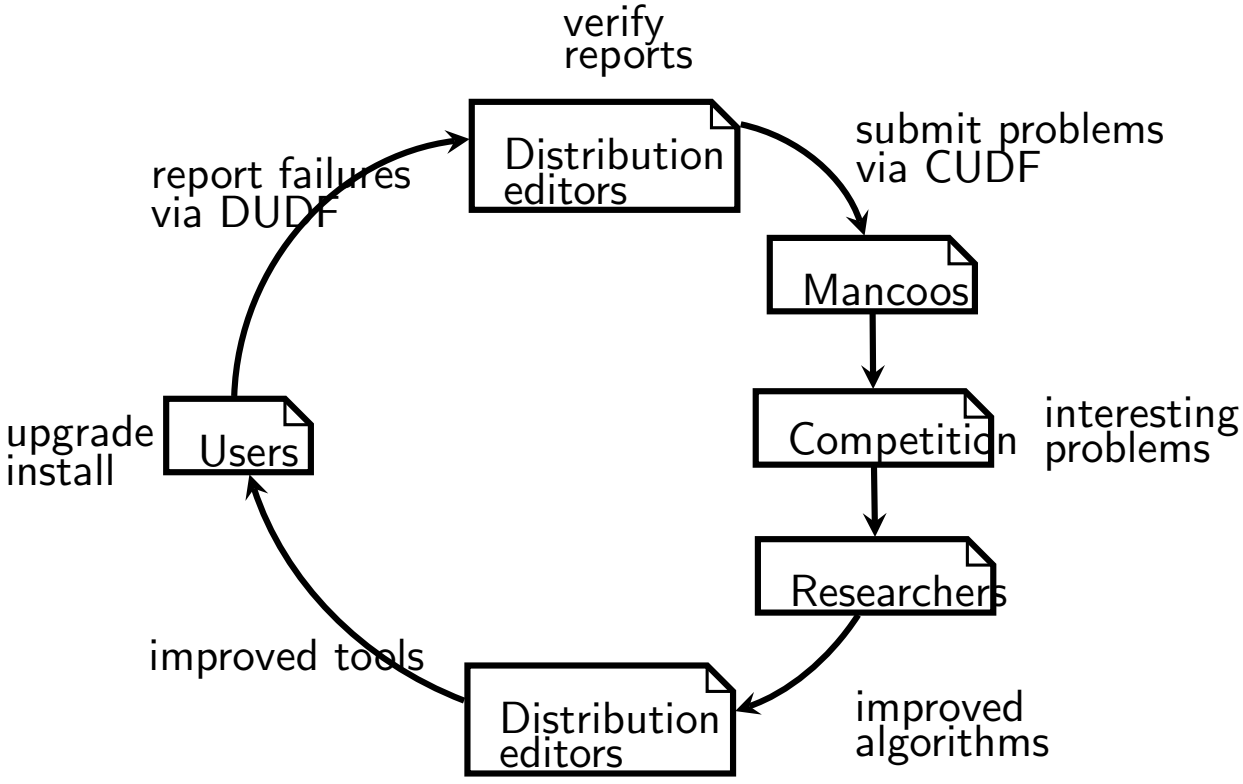
All in a picture

Mancoosi architecture

In Mancoosi, there are really two main activities, corresponding to the architecture of our design, and they are loosely coupled

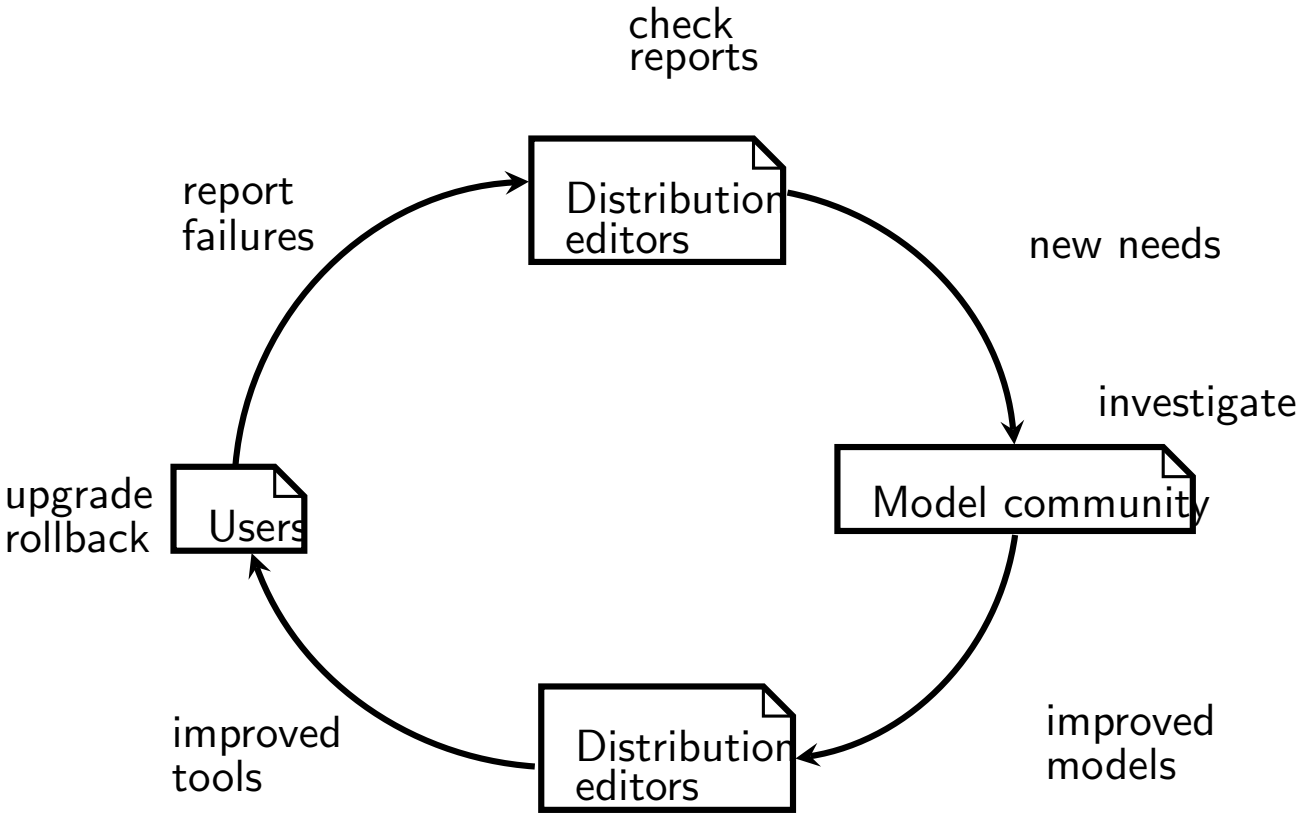


Upgrade resolution



Abstract view of packages through the metadata lens.

Transactional updates



Concrete view of packages through the maintainer scripts lens.

Concluding remark

Everything developed in Mancoosi is *already* relevant for some, and will be increasingly relevant *to all* component based technologies

Linux : packages

Eclipse : plugins

Firefox : extensions

Java : beans

Web services : predictable service assembly

... ...

Project overview

Transactional updates

WP2 build a formal model of the effects of the installation/removal of a software package on a system
Leader: UDA, Prof. Alfonso Pierantonio

WP3 conceive and implement tools, using the model, to perform a transactional update process
Leader: CXA, Paulo Trezentos

Project overview

Upgradeability

WP4 conceive and improve algorithms for optimal upgrade path construction
Leader: UNSA, Prof. Michel Rueher

WP5 collect data from actual user problems, and fuel an *international competition* of specialised algorithms
Leader: UPD, Prof. Ralf Treinen

Project overview

Ancillary WPs

WP1 Coordination
Leader: UPD, Prof. Roberto Di Cosmo

WP6 Dissemination
Leader: Edge-It, Arnaud Lapreuvote

WP7 Collaboration (ongoing)

Building momentum

Sat4J : Daniel Le Berre collaborates tightly

Eclipse P2 : agreement to share the CUDF format

Maven : interest in sharing the CUDF format

RPM : Jeff Johnson got involved in the WP5 lists, `rpm5` will natively support CUDF

Wikipedia : somebody (not us) noticed Mancoosi already!