RPM et politique de packaging

Adapter un paquet à sa distribution

Olivier Thauvin

olivier.thauvin@aerov.jussieu.fr

Introduction

Plan

- RPM rappel
- Politique : généralité
- cas concret : Mandriva

RPM rappel

RPM : Rpm Package Manager

Qu’est ce ?

- un ensemble d’outils
- un format d’archive (.rpm)
- des fichiers au format rpm

RPM : Rpm Package Manager

Qu’est ce ?

- un ensemble d’outils
- un format d’archive (.rpm)
- des fichiers au format rpm

Rôle :

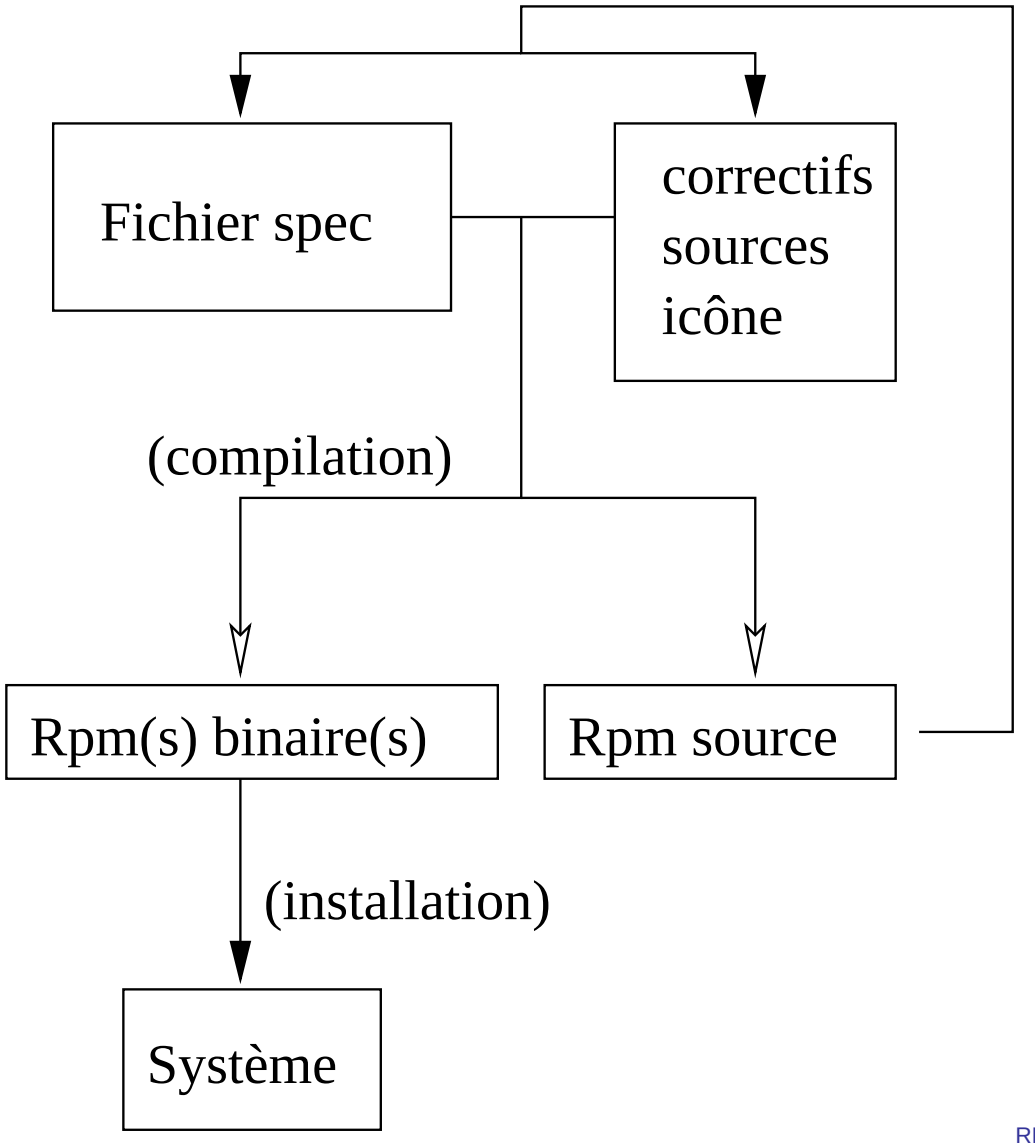
- installer
- inventorier
- controler

Le paquet

Un fichier .rpm contenant :

- entête les informations sur le rpm
  - des informations pratiques sur le logiciel (nom, version, url, . . .)
  - sur la fabrication du rpm
  - une liste de dépendances
  - la liste des fichiers contenus
  - l'historique du paquet (changelog)
- une archive contenant les fichiers

Faire un rpm


Fichier spec
correctifs
sources
icône
(compilation)
Rpm(s) binaire(s)
Rpm source
(installation)
Système

Le fichier spec

Il décrit :

- le (ou les) rpm(s) produit(s)
- le moyen de compiler le logiciel

Il est composé de sections :

- En-tête, %description
- %package
- %prep, %build, %install, %check
- %files
- %verify, %pre, %preun, %post, %postun, %posttrans, . . .
- %changelog

Specfile : exemple

```
Summary: A simple package
Name: simple
Version: 1
Release: 1
License: GPL
Group: MyGroup
%description
A simple package

$ rpm -ba simple.spec
Ecrit: /home/users/olivier/RPM/SRPMS/simple-1
\o/
```

Les macros

Variable propres à rpm.
Origine :

localisation	fournisseur
fichiers macros systèmes	rpm
fichiers macros systèmes	distribution
`~/.rpmmacros`	utilisateur
fichier spec	packager

Elles fournissent des valeurs ou des fonctions.

Politique : généralité

Qu'est-ce qu'une politique de packaging

Définitions d'un ensemble de règles.

Qu'est-ce qu'une politique de packaging

Définitions d'un ensemble de règles.

But :

- homogénéité
- factorisation du travail
- qualité
- faciliter le travail

Bref, c'est bien (tm)

Règles générales : Root est méchant

On ne construit pas sous root,
On ne construit pas sous root,
Et on ne construit pas sous root.

Règles générales : Root est méchant

On ne construit pas sous root,
On ne construit pas sous root,
Et on ne construit pas sous root.

Risques :

- polluer son système
- passer à coté d'erreurs
- compromission, corruption du système
- c'est impossible sur la machine de compilation finale
- ça donne des boutons

Règles générales : ne pas changer la config

L'environnement de rpm fourni adapte les rpms les uns aux autres.

On trouve notamment :

- les chemins par défaut
- la génération des dépendences
- post-installation

Règles de base résumées

- se documenter
- demander aux gens habitués à faire des packages pour leur distribution
- comprendre plutôt que contourner
- un package bien fait pour une distribution est souvent incompatible avec les autres
- faire intégrer son paquet dans la distribution (ça ne vaut pas que pour rpm)

cas concret : Mandriva

Configuration utilisateur

- definir son environnement rpm
- définir le packager tag

Configuration utilisateur

- definir son environnement rpm
- définir le packager tag

.rpmmacros :

```
%_topdir      /home/users/olivier/RPM
%_tmppath     /home/users/olivier/RPM/tmp/
%packager     Olivier Thauvin <nanardon@zar
%distribution Mandriva

%_signature   gpg
%_gpg_name    Olivier Thauvin
%_gpg_path    ~/.gnupg
```

Politique de nommage des paquets

- nom de paquet en minuscules (il y a quelques exceptions)
- le nom du paquet correspond au nom logiciel
- les module pour langage sont préfixé par le nom du langage

Exemples :

- perl
- rpm
- perl-POE

%mkrel

Problématique des backport :

	cooker		stable
sans	1mdv	==	1mdv
avec	1mdk 1mdv2007.0	> >	0.1.20060mdk 1mdv2006.0

%mkrel

Problématique des backport :

	cooker		stable
sans	1mdv	==	1mdv
avec	1mdk	>	0.1.20060mdk
	1mdv2007.0	>	1mdv2006.0

Release : 1mdk
devient :
Release : %mkrel 1

Chemin des fichiers

- Prefix a une signification particulière, ne pas l'utiliser
- respecter les emplacements standards
- préférer les macros
- /usr/local est réservé aux logiciels installé sans rpm

Chemin standard

chemin	macro	usage
/usr	%_prefix	chemin général
/etc	%_sysconfdir	configuration
/usr/bin	%_bindir	application
/usr/sbin	%_sbindir	app. pour root
/usr/lib /usr/lib64	%_libdir	bibliothèques
/usr/share	%_datadir	données
/usr/share/man	%_mandir	pages de man

Compilation et installation

Compilation du soft :

- utiliser les bonne options (%optflags)
- faire attention aux chemins
- installer les fichiers dans %buildroot

Cas simple avec autotools

```
%build
%configure
%make

%install
%makeinstall_std
```

Extraits de spec

Example d’un cas compliqué :

```
export CFLAGS="%optflags"
./configure \
    --prefix %_prefix \
    --bindir %_bindir \
    --libdir %_libdir
# parallèle make don’t work
make

%install
make install DEST=%buildroot

#installation d’un fichier:
cat > %buildroot%_sysconfdir/%name.cfg <<EOF
blabla
EOF
```

Intégrer les fichiers dans le paquet

- tout les fichiers doivent être listés

Intégrer les fichiers dans le paquet

- tout les fichiers doivent être listés
- utiliser les macros

Intégrer les fichiers dans le paquet

- tout les fichiers doivent être listés
- utiliser les macros
- les répertoires propres au logiciels doivent être intégrés

Intégrer les fichiers dans le paquet

- les fichier de configuration ne doivent pas être remplacés

  %config(noreplace) %_sysconfdir/foo

Intégrer les fichiers dans le paquet

- les fichier de configuration ne doivent pas être remplacés

  %config(noreplace) %_sysconfdir/foo

- les documentation sont intégrées avec %doc

  %doc README . . .

Intégrer les fichiers dans le paquet

- les fichier de configuration ne doivent pas être remplacés

  %config(noreplace) %_sysconfdir/foo

- les documentation sont intégrées avec %doc

  %doc README . . .

- utiliser %find_lang pour les traductions

  %install

  %find_lang %name

  . . .

  %files -f %name.lang

Les dépendences

Pour les paquets binaires : rien à faire en général

Pour les sources :

- identiques pour toutes les architectures
- suffisants pour builder les paquets
- non redondants

Exemple :

```
BuildRequires: libfoo-devel
```

Libidification

But :

- avoir plusieur versions de la bibliothèque
  Nomage des paquet en utilisant le numéro majeur
- avoir plusieur architectures
  Nomage en incluant un prefix

majeur :	0	1
32bits	libfoo0 lib/foo.so.0	libfoo1 lib/foo.so.1
64bits	lib64foo0 lib64/foo.so.0	lib64foo1 lib64/foo.so.1

Libidification : dans la pratique

```
%define major 0
#similar to %_lib%name%major
%define libname %name %major
...
%package -n %libname
Provide: lib%name = %version-%release
...
%files -n %libname
%defattr(-, root, root, -)
%_libdir/*.so.*
```

Conclusion

rpmlint

Outil pour vérifier que les paquets sont conformes à la politique

- maintenu par le merveilleux Michaël Scherer
- disponible sur `http://rpmlint.zarb.org/`

```
$ rpmlint -i rpm-mandriva-setup-1.24-1mdv2007.0.i586.rpm
E: rpm-mandriva-setup no-binary
The package should be of the noarch architecture because it
doesn’t contain any binaries.

E: rpm-mandriva-setup only-non-binary-in-usr-lib
There are only non binary files in /usr/lib so they should
be in /usr/share.
```

The End

Ce document sera là :

`http://forge.ipsl.jussieu.fr/docipsl/`.

Le rpmhowto de mandriva :

`http://qa.mandriva.com/twiki/bin/view/Main/RpmHowTo`.

Merci :

- Guillaume Rousse
- Benoît Audouard
- Eric Villard
- CNRS/ISPL pour l'hébergement

Questions ?