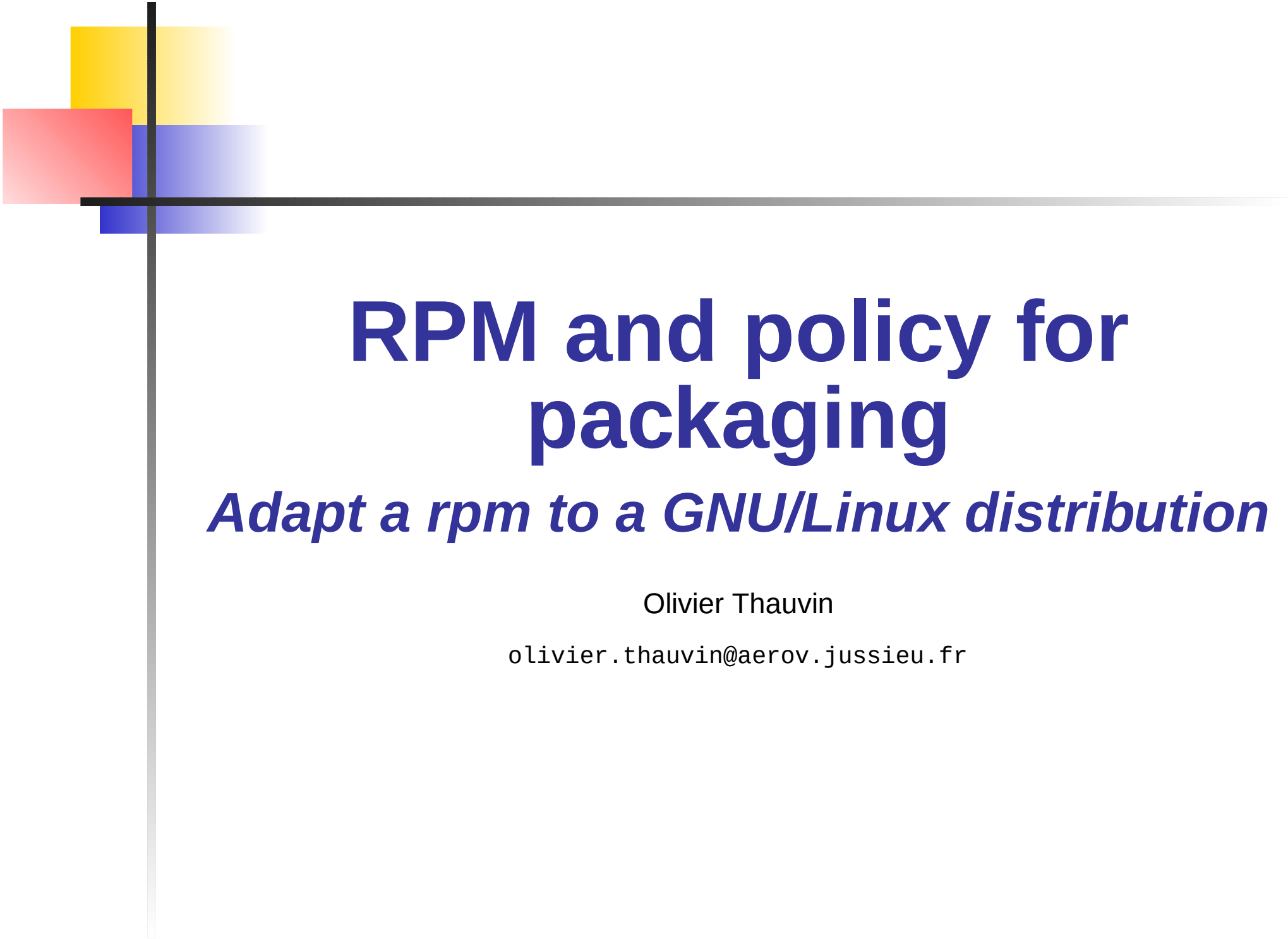
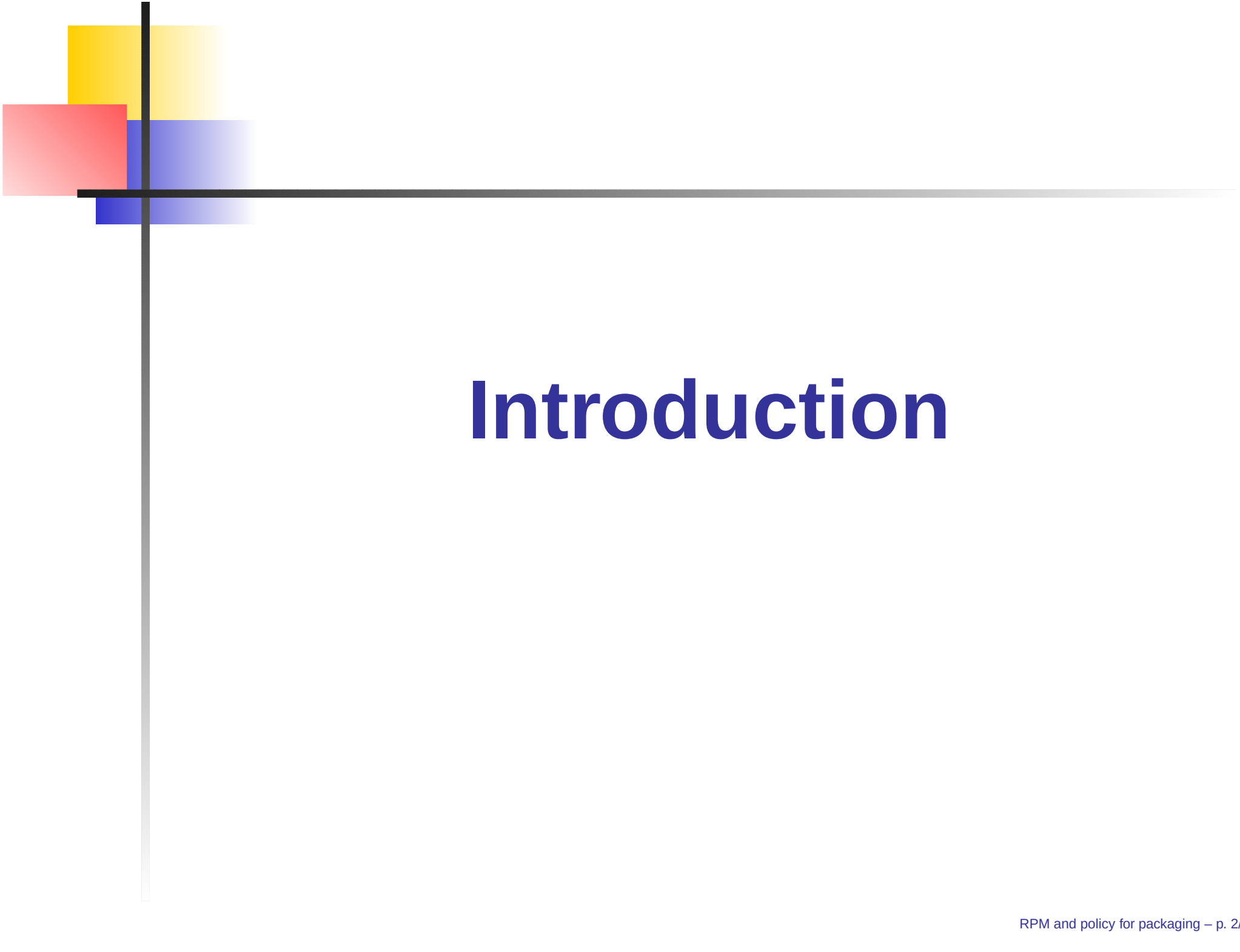


RPM and policy for packaging

Adapt a rpm to a GNU/Linux distribution

Olivier Thauvin

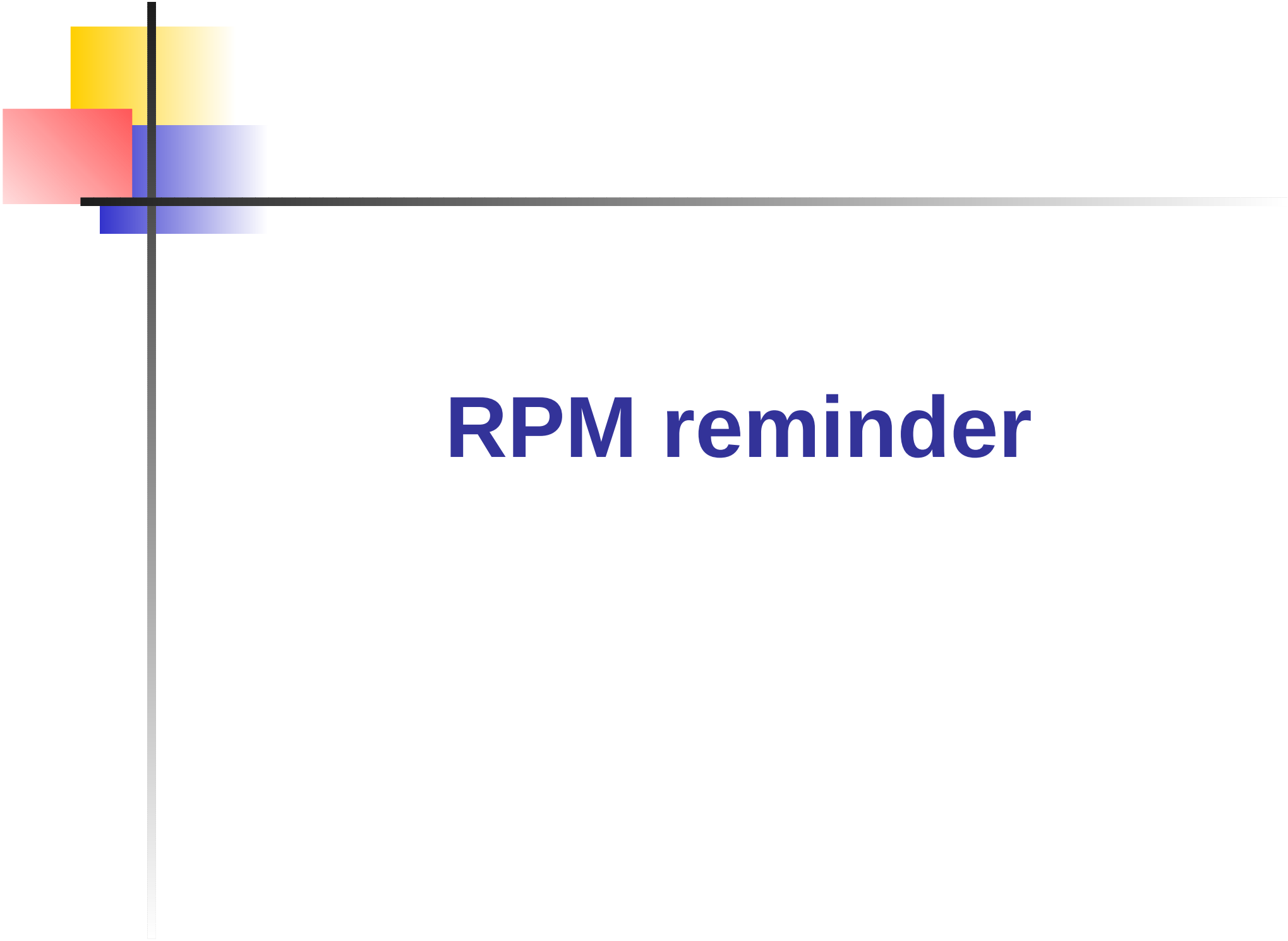
olivier.thauvin@aerov.jussieu.fr



Introduction

Plan

- RPM reminder
- Policy : introduction
- Matter of fact case : Mandriva



RPM reminder

RPM : Rpm Package Manager

What is it ?

- a set of tools
- an archive format (.rpm)
- files in rpm format

RPM : Rpm Package Manager

What is it ?

- a set of tools
- an archive format (.rpm)
- files in rpm format

Roles :

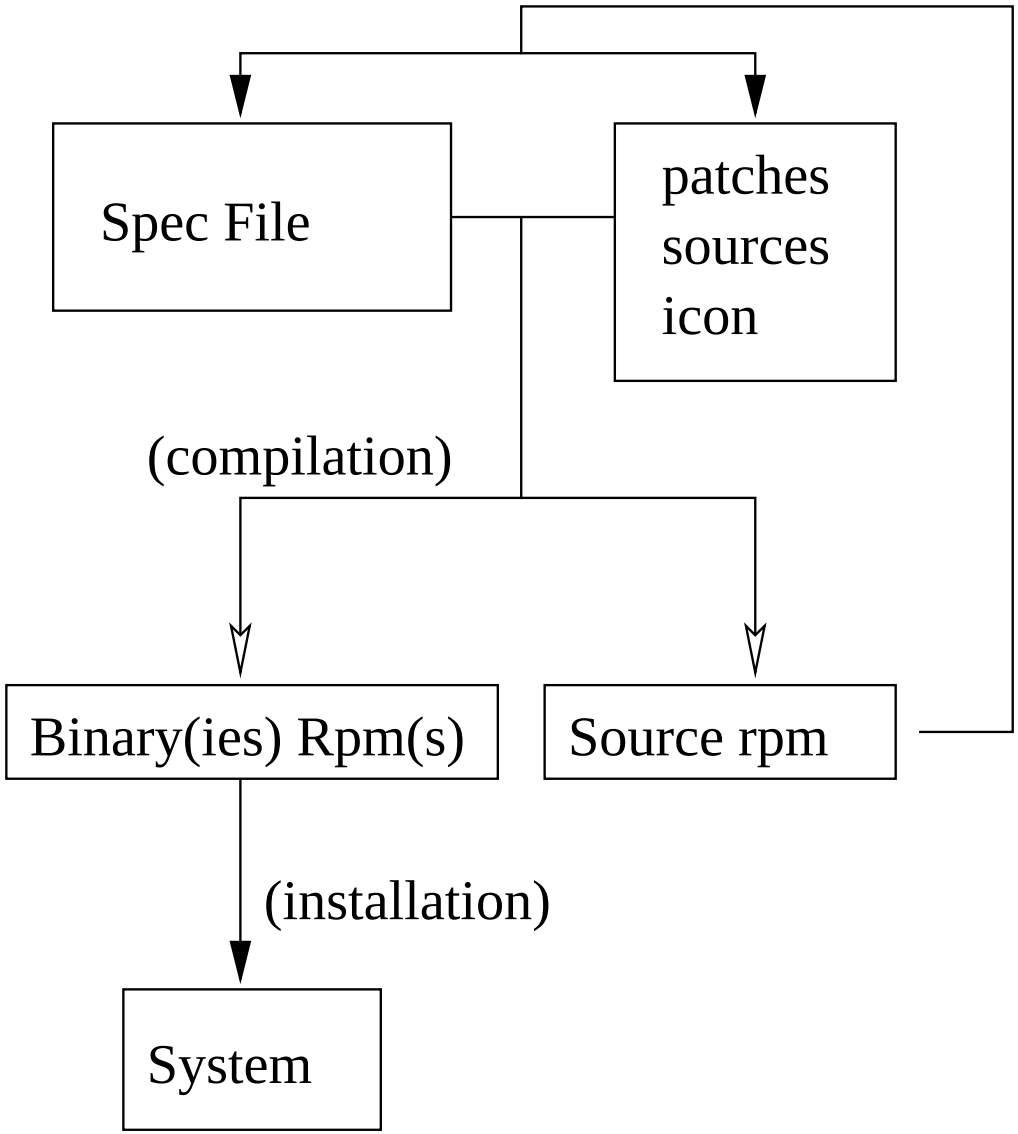
- install
- list
- control

The package

An .rpm file containing :

- entête informations about the rpm
  - practical informations about the software (name, version, url, . . .)
  - to make the rpm
  - a list of dependances
  - a list of files provided
  - history of the package (changelog)
- an archive containing the files

Making an rpm



The spec file

It describes :

- the rpm(s) produced(s)
- the compiling of the software

There are the following sections :

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

Specfile : example

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

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

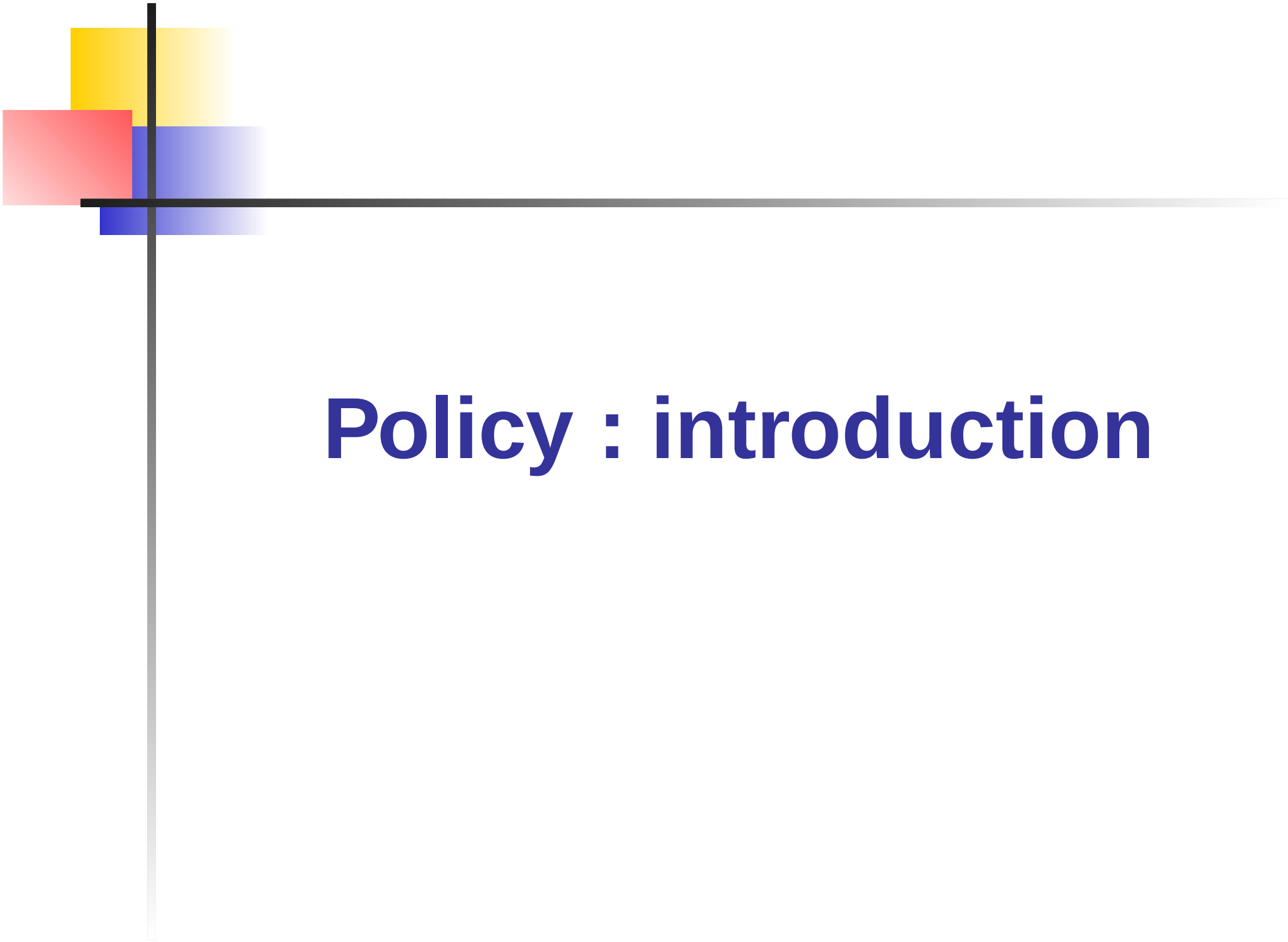
Macros

RPM specific variables.

Origine :

where	provider
system macros files	rpm
system macros files	distribution
`~/.rpmmacros`	user
spec file	packager

They provide values or functions.



Policy : introduction

What is a policy

To define rules.

What is a policy

To define rules.

Goal :

- homogeneity
- work factorisation
- quality
- ease the work

In brief, that’s good (tm)

Root is evil

Do not build as root,
Do not build as root,
And do not build as root.

Root is evil

Do not build as root,
Do not build as root,
And do not build as root.

Risks :

- pollute your system
- skip errors unwillingly
- compromission, corruption of your system
- cannot be done on the compile-server
- it gives buttons

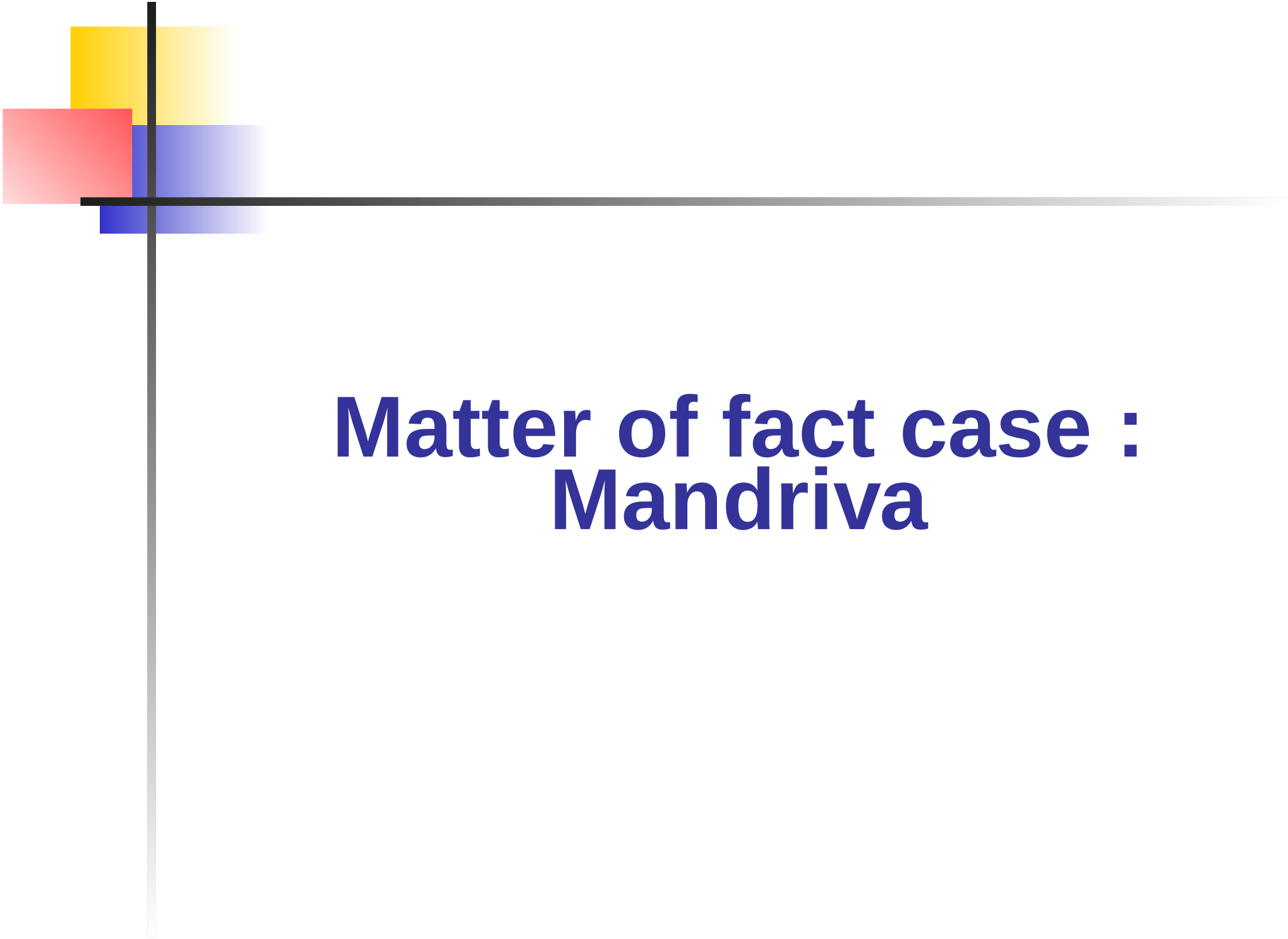
Do not change the config

The environment provided with rpm adapts the rpms one to each other.
You notably find :

- default paths
- dependances generation
- post-installation

Summary of basic rules

- document yourself
- ask people used to making packages for their distribution
- understand rather than work-around
- a package well done for a distribution often becomes uncompatible with others
- get your package integrated to the distribution (it’s worth doing for other things than rpm)



Matter of fact case : Mandriva

User configuration

- define your rpm environment
- define the packager tag

User configuration

- define your rpm environment
- define the packager tag

.rpmmacros :

```
1   %_topdir      /home/users/olivier/RPM
    %_tmppath     /home/users/olivier/RPM/tmp/
    %packager     Olivier Thauvin <nanardon@zar
    %distribution Mandriva
5
    %_signature   gpg
    %_gpg_name    Olivier Thauvin
    %_gpg_path    ~/.gnupg
```

Package naming policy

- package name are lowercase (there are some exceptions)
- package name should match software name
- language modules have package name prefixed by language name

Example :

- perl
- rpm
- perl-POE

%mkrel

Problems with backport :

	cooker		stable
without	1mdv	==	1mdv
with	1mdk	>	0.1.20060mdk
	1mdv2007.0	>	1mdv2006.0

%mkrel

Problems with backport :

	cooker		stable
without	1mdv	==	1mdv
with	1mdk	>	0.1.20060mdk
	1mdv2007.0	>	1mdv2006.0

Release : 1mdk
becomes :
Release : %mkrel 1

Path for files

- Prefix has a specific meaning, do not use it
- respect the standard paths
- prefer macros
- /usr/local is reserved for softwares installed without rpm

Standard path

path	macro	usage
/usr	%_prefix	general path
/etc	%_sysconfdir	configuration
/usr/bin	%_bindir	application
/usr/sbin	%_sbindir	app. for root
/usr/lib /usr/lib64	%_libdir	libraries
/usr/share	%_datadir	data
/usr/share/man	%_mandir	man pages

Compilation and installation

Compiling a software :

- use the appropriate options (%optflags)
- take care to paths
- install files in %buildroot

Simple case with autotools

```
%build
%configure
%make

%install
%makeinstall_std
```

Spec extracts

Example of a complicated case :

```
export CFLAGS="%optflags"
./configure \
    --prefix %_prefix \
    --bindir %_bindir \
    --libdir %_libdir
# parallel make do not work
make

%install
make install DEST=%buildroot

#install one file:
cat > %buildroot%_sysconfdir/%name.cfg <<EOF
blabla
EOF
```

Include the files in the package

- all files must be listed

Include the files in the package

- all files must be listed
- use macros

Include the files in the package

- all files must be listed
- use macros
- software-specific directories must be included

Include the files in the package

- configuration files must not be replaced

  %config(noreplace) %_sysconfdir/foo

Include the files in the package

- configuration files must not be replaced

  %config(noreplace) %_sysconfdir/foo

- documentations are included with %doc %doc README . . .

Include the files in the package

- configuration files must not be replaced

  %config(noreplace) %_sysconfdir/foo

- documentations are included with %doc %doc

  README . . .

- use %find_lang for translations

  %install

  %find_lang %name

  . . .

  %files -f %name.lang

Dependances

For binary packages : usually, nothing to do

For sources :

- identical for all architectures
- suficient to build the packages
- no redundancy

Example :

```
BuildRequires: libfoo-devel
```

Libidification

Goal :

- have many versions of a library
  Name packages using major number
- have many architectures
  Name package including a prefix

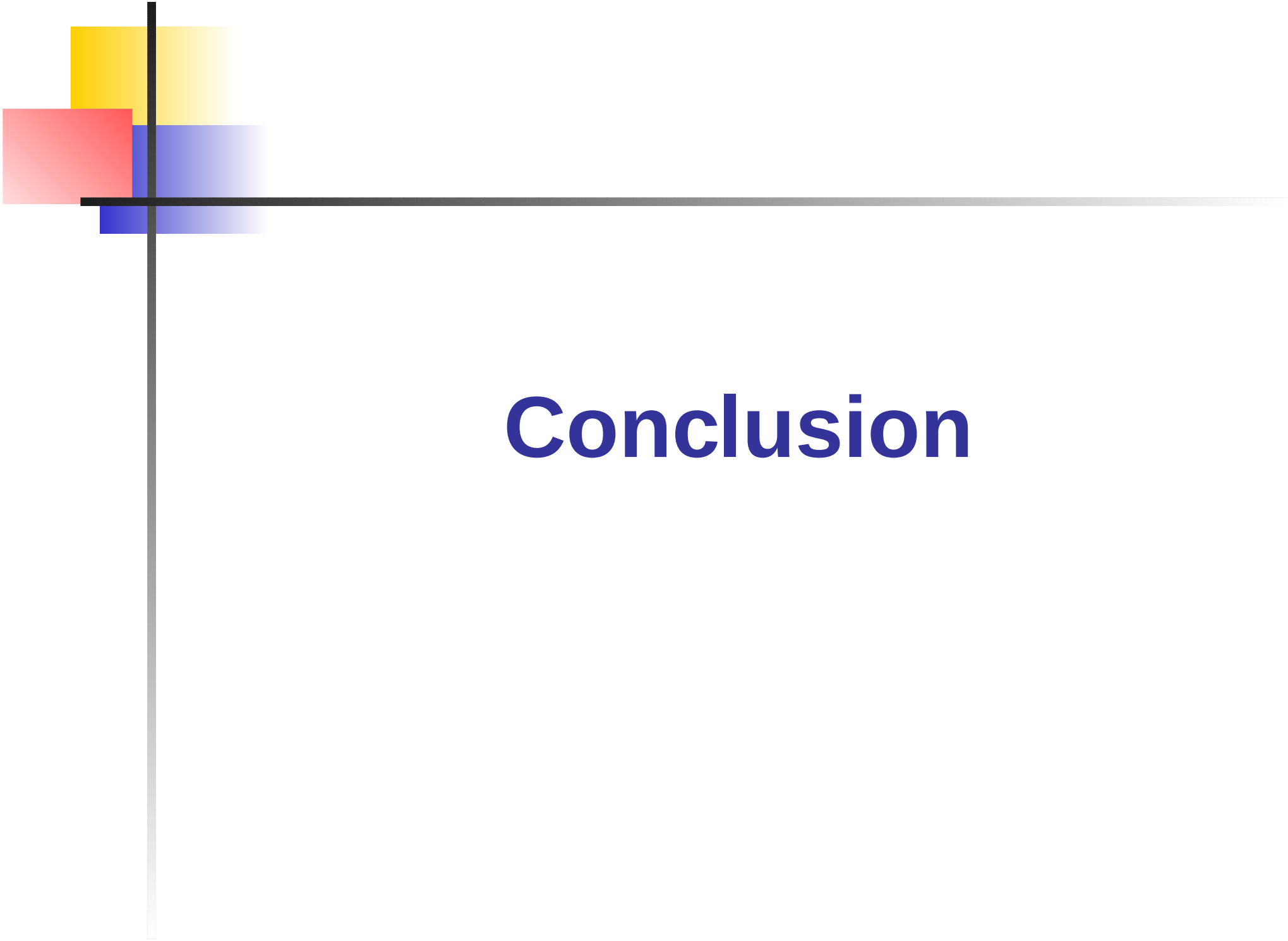
majeur :	0	1
32bits	libfoo0 lib/foo.so.0	libfoo1 lib/foo.so.1
64bits	lib64foo0 lib64/foo.so.0	lib64foo1 lib64/foo.so.1

Libidification : use case

```
%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

Tool to check packages complies to the policy

- maintained by the wonderful Michaël Scherer
- available at `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

This document is there :

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

Mandriva's rpmhowto :

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

Thanks :

- Guillaume Rousse
- Benoît Audouard
- Eric Villard
- CNRS/ISPL for web hosting

Questions ?