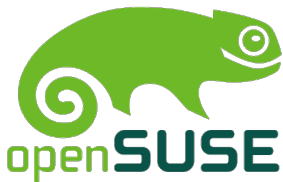


OBS introductory course for SUSE people



Tomáš Chvátal
tchvatal@suse.com
L3-Packaging/Packaging

2016/09/11

Introduction

Explanation of few shortcuts used in the slides

- `isc` = alias for `osc -A IBS`
- `ibs` = `build.suse.de`
- `obs` = `build.opensuse.org`
- `SR#` = submit request in any of the above instances
- `bnc` = bug on `bugzilla.novell.com`

What is buildservice

- provider for reproducible build results
- VCS for our packages
- collaboration tools using SR#s

How do I communicate with the tool

- Using web interface on the ibs or obs locations
- Using commandline interface called osc (osc is provided on most distributions as package too)

CLI Configuration tips

Configuration suggestions

`$HOME/.osrcrc`

```
build-jobs = 16 # value for buildjobs
extra-pkgs = vim gdb strace mc less unzip # some tools
no_verify = 1 # local verification only
```

Configuration suggestions - cont.

```
[https://api.suse.de]  
user = bugzillausername  
pass = bugzillapw  
email = guessyourself  
aliases = ibs
```

```
[https://api.opensuse.org]  
user = bugzillausername  
pass = bugzillapw  
email = guessyourself  
aliases = obs
```


Configuration suggestions - cont.

`$HOME/.bashrc`

As few commands will be called quite often it is easier to alias them.

```
export COMP_WORDBREAKS=${COMP_WORDBREAKS/./}  
alias isc="osc -A ibs"  
alias oscb="osc build --ccache"  
alias oscsd="osc service localrun download_files"
```

Basic CLI usage

Sample workload

```
$ osc co Archiving/rsnapshot
$ cd Archiving/rsnapshot
$ vi rsnapshot.spec # hackyhacky
$ osc vc -m "Fix something. Resolves bnc#1234"
$ oscb
$ osc ci
```

Some handy tips

osc build options

Build debuginfo packages locally:

```
$ osc build --debuginfo/-d
```

Skip the pesky post-build-checks:

```
$ osc build --nochecks
```

Add extra rpms from local system rather than from O/IBS:

```
$ osc build --prefer-pkgs /my/directory/withrpms/
```

Add extra packages on the chroot environment:

```
$ osc build -x gdb
```

osc tips continued

Do not init the chroot just go ahead with build:

```
$ osc build --no-init
```

Get the buildlog from the local build:

```
$ osc lbl <project eg. openSUSE_13.2>
```

Avoid password request prior each build:

```
echo "yourlogin ALL = (root) NOPASSWD: /usr/bin/build" >> /etc/sudoers
```

Collaboration/maintenance

openSUSE Factory

`https:
//progress.opensuse.org/workflow/factory-proposal.html

https://en.opensuse.org/openSUSE:
Factory_development_model`

openSUSE Factory cont.

```
$ osc develproject openSUSE:Factory rsnapshot
Archiving
$ osc branch Archiving rsnapshot
A working copy of the branched package can be checked out with:
osc co home:scarabeus_iv:branches:Archiving/rsnapshot
$ osc co home:scarabeus_iv:branches:Archiving/rsnapshot
$ cd home:scarabeus_iv:branches:Archiving/rsnapshot
$ vi rsnapshot.spec # hackyhacky
$ osc vc -m "Fix something. Wrt bnc#2345"
$ oscb
$ osc ci
$ osc sr -m "Finally got around to fix bnc#2345 so enjoy"
```

openSUSE Releases

- Forwarding fixes from devel project to released distributions are always up to maintainer (your) decision
- You should forward only smaller things (ie. no huge version updates that could break too much)
- One should be MORE careful when creating the maint. update than when submitting to Factory itself
- https://en.opensuse.org/openSUSE:Build_Service_Concept_Maintenance
- In some cases packages in Leap come over from SLE and thus need to be updated there first

openSUSE Leap package origin

In Leap one must first determine where the package comes from by using various scripts:

```
$ /mounts/work/src/bin/is_maintained.rb curl
```

```
Product                                Codestream  
SLE-DEBUGINFO_11-SP1-TERADATA  SUSE:SLE-11-SP1:Update  
[...]
```

```
Leap package comes from SUSE:SLE-12:Update  
[...]
```

```
$ osc cat openSUSE:Leap:42.1:Update/00Meta/lookup.yml | grep "^curl:"  
curl: SUSE:SLE-12:Update
```

openSUSE Releases cont.

```
$ osc maintained rsnapshot
openSUSE:12.3:Update/rsnapshot
openSUSE:13.1:Update/rsnapshot
$ osc mbranch rsnapshot
A working copy of the maintenance branch can be checked out with:
osc co home:scarabeus_iv:branches:OBS_Maintained:rsnapshot
$ cd home:scarabeus_iv:branches:OBS_Maintained:rsnapshot
$ ls
rsnapshot.openSUSE_12.3_Update/  rsnapshot.openSUSE_13.1_Update/
$ vi rsnapshot.openSUSE_12.3_Update/rsnapshot.spec
$ vi rsnapshot.openSUSE_13.1_Update/rsnapshot.spec
% for each folder now
$ osc vc -m "I did fancy maintenance changes, I am great"
$ oscb
$ osc ci
% back one level
$ osc mr -m "Maintenance update wrt bnc#something"
created request id 1234
```

SLE

- For SLE all updates are requested by Maintenance DPT and you receive notification with deadline
- All of the submissions are to adhere to Maintenance SLA (ask your boss to provide it to you)
- If you do something wrong people from QAM will report bugs to you and frown :)
- You always receive list of bnc#s you are supposed to fix, note that you can nominate additional fixes
- SLE does use new maintenance model from SLE-11 as openSUSE

SLE cont.

```
$ isc maintained vsftpd
SUSE:SLE-10-SP3:Update:Test/vsftpd
SUSE:SLE-11:Update:Test/vsftpd
$ isc mbranch vsftpd
$ isc co home:scarabeus_iv:branches:OBS_Maintained:vsftpd
$ cd home:scarabeus_iv:branches:OBS_Maintained:vsftpd
$ ls
vsftpd.SUSE_SLE-10-SP3_Update_Test/  vsftpd.SUSE_SLE-11_Update_Test/
% for each folder
$ vi vsftpd.spec
$ vi vsftpd.changes
$ oscb
$ osc ci
$ osc sr <RESPECTIVE_PROJECT>/<EXACT_PKG_NAME> -> osc sr SUSE:SLE-10-SP3:Update:Test/vsftpd
```

Webinterface

Practical example

Lets go over to `http://build.opensuse.org`

Endnote

Further reading/Contact points

- https://en.opensuse.org/openSUSE:Packaging_guidelines
- <http://l3ms.suse.de/packagers>
- opensuse-packaging@opensuse.org
- pack@suse.cz
- [#opensuse-factory@freenode.net](https://freenode.net/#opensuse-factory)
- [#pack@irc.suse.cz](https://irc.suse.cz/#pack)

Thanks/Questions

Thank you for your attention.
Are there any questions?