Packaging Tools for OpenServer & UnixWare

Ron Record
rr@sco.com

John Wolfe
jlw@sco.com

Why Package ?

- Single software administration tool
- Ease of installation, update, and removal
- **Support for multiply loaded versions**
- Configuration during install/removal phases
- Automatic check for dependencies and verification
- Set permissions, ownership, and group
- Graphical user interface
- Support for remote network install/removal

Which packaging system?

- SCO OpenServer
    - Base system packages are in CDMT format
    - User base typically more familiar with custom
    - OpenServer also supports pkgadd format
    - Custom does not recognize pkgadd packages (however, can "wrap" pkgadd in custom)
- UnixWare
    - Pkgadd only, no support for CDMT
- Single pkgadd package can be used to install on both OpenServer and UnixWare
    - HBA & NIC drivers are pkgadd on OSR6 & UW7

Tips & Pits

- Know your package
    - Read the README/INSTALL/PORTING guides in the package's source tree
    - Find out which files/dirs need special permissions or ownership
- Examine other vendors packages if possible
    - Use rpm2cpio to extract a source rpm's spec file
    - See control scripts in /opt/K/SCO/<package>/<version>/cntl/
- Pick a unique package name

- Determine and specify any package dependencies
- Take special care to identify configuration files
    - Preserve configurations across updates
    - Specify configuration files as "variable"
- Test a variety of installation and removal scenarios
- Avoid file and directory name collisons
- Add SCO Help entries when possible
- Provide an accurate succinct description

- Provide graphical user interface desktop files for GUI applications where appropriate
- Careful not to overwrite permissions/ownership of a system directory or another package
    - Files and directories listed in a package's input files should be "owned" by that package
- Provide a System V style init script if appropriate and create /etc/rc?.d symlinks if desired

Packaging:
Custom Distribution Mastering
Toolkit (CDMT) for OpenServer

- Use cqs script for pre-installation system prep – for example, creation of new users/groups
- Do not use the cqs script for package configuration
- Use the cqs script to gather user input in order to avoid installation stops
- Use ccs script for install/removal config during phases
  - Adding/removing entries in SCO Help doc system
  - Setting system specific configuration parameters
  - Enabling/disabling automatic startup of service(s)
  - Initializing product configuration (e.g. Setup cache)

Control Scripts (continued)

- Source in the ccsSetup.sh standard functions library
- Arguments to ccs script are step, keywords, and pkglist
  - The step argument indicates which phase we are in
  - The keywords argument can be used to determine if this is an upgrade or initial system load
  - The package list argument provides a list of packages in the component in the form *vendor:component:package*
- Can use SSO_SHARED_ROOT variable to access files not yet exported or otherwise only available in the SSO

Example CCS Script (Part 1)

```
#!/bin/sh
scriptname="$0"
step="$1"
keywords="$2"
Pkglist="$3"
. ccsSetup.sh
doPostExport() {
    [ -x /usr/bin/doctool ] && {
        /usr/bin/doctool --add $DOC/SDK_qt3.desktop
    }
}
```

```
doPreUnExport() {
    [ -x /usr/bin/doctool ] && {
        /usr/bin/doctool –remove $DOC/SDK_qt3.desktop
    }
}
DOC=DevSysDoc/SDKhome/SDKgroup/SDK_qt3
ccs_return_value=0
case "$step" in
        POST_EXPORT) doPostExport ;;
        PRE_UNEXPORT) doPreUnExport ;;
esac
exit $ccs_return_value
```

 Contents of the example file /usr/share/meta/doc/DevSysdoc/SDKhome/SDKgroup/ SDK_qt3/SDK_qt3.desktop

[Desktop entry]

Name=Qt Graphical User Interface Library

DocPath=/usr/lib/qt3/doc/html/index.html

X-COL-rewrite=SDK_qt3

X-COL-Weight=4.0

Many more examples in /usr/share/meta/doc/…

- Convert a list of relative pathnames to CDMT input files with */usr/bin/cdmtConvert*

- Create the SSO databases from the CDMT input files with */usr/bin/cdmtParse*

- Populate $CDMT_DIR/sso/ with compressed distribution files using *cdmtCompress*

- Generate custom-installable archives from the $CDMT_DIR/sso/ hierarchy with *cdmtArchive*

- Previously I used Skunkware's mkvol scripts but now I use metapkg

SCO®

- Separate Storage Section Object (SSO) per component / version
  - Good feature in general
  - Presents issues with software that uses **realpath()**
    - **Java – 3 packages -> 1 component**
      - **Other packages need to set CLASS_PATH to add jar files**
    - **Python**
      - **SSO contains archive of files**
      - **Extract SSO archive to "user" space**

14

Packaging:

SVR4 Package Datastream

- Package objects – files to be distributed
- **pkginfo file**

        PKG="MyProd"

        NAME="My Very Own Prod"

        VERSION="1.0.1"

        CATEGORY="application"

        CLASSES="base extra"

        ARCH="i386"

        BASEDIR=/

- **prototype file**
    - **Use pkgproto to construct**
    - **Add package "info" files and installation scripts**

- **copyright – printed at install time**

- **depend – details software dependencies**
  - **Prerequisite packages**
  - **Incompatible packages**
  - **Reverse dependency**

- **space – additional space for files dynamically created at install time.**

- **compver – defines previous (or future) compatible versions of this package**

- Must be Bourne SHELL scripts -
/bin/sh

- **request script**

  - **Administrator interaction prior to install**

  - **Query OS, MP, current configuration**

  - **Assigns or redefines package parameters**

    - **Ouput as a list parameters and their values**

    - **Append to request input param #1**

      cat >$1 <<!
      PARAM1="value 1"
      PARAM2="value 2"
      !

18

- **preinstall script**
  - **Patches (MP) archive files to be replaced**
- **postinstall script**
  - **Driver packages trigger kernel rebuilds**
  - **Establish links into related packages**
  - **Build application configuration, registry, etc.**
  - **Add created files / links to contents file**
- **preremove scripts**
  - **Reverse postinstall steps**
- **postremove script**
  - **Reverse preinstall steps**
  - **i.e. restore previous versions of patched files**

/var/[s]adm/install/contents

- **Master list of installed paths**
  - **File type**
  - **Mode, owner, group**
  - **Class**
  - **Size, cksum, last mod time**
  - **package(s) that have installed this file**

- ## /var/[s]adm/pkg/<name>

  - Entries for admin/pkginfo/install scripts
    - **install sub-directory contains package scripts**

- Populate distribution tree

- Correct permissions / owner / group

- Create pkginfo file
- Create optional information files
- Create installation scripts if necessary
  - preinstall / postinstall
  - preremove / postremove

- Create the prototype file

  pkgproto [-c <class>] [path1=[path2]...]

  pkgproto -c base usr=/usr opt=/opt
  > prototype

- Add entries for pkginfo, info & install scripts

  i pkginfo=pkginfo
  i copyright=copyright
  i request=request
  ...

- Run pkgmk

  pkgmk -o -c -d `pwd` -r `pwd`

- Run pkgtrans, if desired, to create a pkgadd datastream

  pkgtrans -s `pwd` `pwd`/MyProd.pkg MyProd

- Install and test newly created package

  pkgadd -d `pwd`/MyProd.pkg all

- **Essentially the same points as CDMT**
- **Check existing package scripts in**

  /var/[s]adm/pkg/<pkg_name>/info
  - **hba & nic drivers on OSR6 and UW7**
  - **J2SE x.x.x on UW7**
  - **J2SE 1.3.1 on OSR 5.0.7**
  - **UDK/OUDK set on OSR5 and UW7**
  - **UnixWare maintenance packs (MP)**

- **Install with pkgadd is additive**
  - **NO automatic replace**
  - **NO retaining previous versions**
  - **Files installed and pkginfo file updated**
  - **Use installation scripts to save / restore previous version, if needed**
  - **Use removef to delete now unneeded files – or remove old version first**

- Download and install the mkpkg package

    ftp://ftp2.sco.com/pub/skunkware/uw7/Packages/mkpkg-1.1.pkg

- Extract distribution tree into empty directory

- Set correct permissions/ownership

- Run

    mkpkg <package name> <package version>

- Create installation scripts if necessary

- If necessary, edit prototype file and run

    ./MakePKG

- Install and test newly created package

    pkgadd -d `pwd`/MyProd.pkg all

26

**Packaging:
Using Metapkg To Create
Custom And Pkgadd Installable
Packages**

- **Download and install the latest metapkg media images**
  ftp://ftp2.sco.com/pub/skunkware/osr6/vols/

- Metapkg documentation is installed in
  /usr/share/doc/packages/metapkg/

- Metapkg examples are installed in
  /usr/share/doc/packages/metapkg/examples/

- Metapkg convenience scripts are installed in
  /usr/share/doc/packages/metapkg/scripts/

- The metapkg and reman binaries as well as mkcdmt and mkpkgadd symbolic links are installed in */usr/bin/*

- Create input directory and populate dist directory
- Create dist/cntl/ scripts, if any
- Create <package name>.mkcdmt control file
  - Specify non-default permissions and ownership
  - Symbolic links specified as additional exports
  - Dependencies and updated versions listed here
- Run mkcdmt. For example:

    **mkcdmt -f -h -d `pwd` -P gimp **

    **-D "GNU Image Manipulation Program" **

    -V 2.2.7Sb -p `pwd`/gimp.mkcdmt

- Run **make**

Sample Metapkg Control File

```
prepare ("Checking and preparing distribution") {
  auto_compress_texinfo();
  auto_format_mansource();
  auto_strip(TRUE,TRUE);
}
package ("/", "${METAPKG_DESCRIPTION}", "QT3") {
  file ("/usr/lib/qt3/mkspecs/unixware-cc/qmake.conf") {
    access (SERVER);
  }
  file ("/usr/lib/qt3/lib/libqt-mt.so.3.3.8") {
    addexport ("/usr/lib/qt3/lib/libqt-mt.so", normal);
  }
}
component ("qt3", "${METAPKG_VERSION}", "${METAPKG_DESCRIPTION}") {
  dependency ("SCO:gwxlibs");
  upgrades("^3.3.5*");
}
```

Shortcut Scripts

- See */usr/share/doc/packages/metapkg/scripts/*

- *Create a gzip'd tar archive of the distribution files relative to / and name it <package name>-<version>-dist.tar.gz*

- *Place the gzip'd tar archive in the /dist/ directory that the setup shell script points to*

- *Create an empty directory <package name> in the packaging directory and cd into it*

- *Run ../setup*

- *Run the listlinks script to create entries for symbolic links in the file /tmp/<package name>-symlinks*

- *Edit <package name>.mkcdmt adding the above file to the "package" section. Run "MakeCDMT" then "make"*

Guidance / Assistance

- CDMT documentation
  http://osr600doc.sco.com/en/manCDMT/CONTENTS.html

- SCO Software Manager (custom) documentation
  http://osr600doc.sco.com/en/man/html.ADM/custom.ADM.html

- Installing and Managing Software Components
  http://osr600doc.sco.com/en/INS_install/swaN.admin.html

- Installing/Managing Software Over a Network
  http://osr600doc.sco.com/en/INS_install/swnetN.netinstall.html

- Metapkg documentation
  ftp://ftp2.sco.com/pub/skunkware/osr5/devtools/metapkg/doc

- Porting Guide: http://www.sco.com/support/docs/openserver/600/porting/osr6portingTOC.html

- Upgrade Guide: http://www.sco.com/support/docs/openserver/600/upgrade/index.html

- Online Documentation and Late News http://www.sco.com/support/docs/openserver/

- OpenServer 6 Support Download Page:
    - http://www.sco.com/support/update/download/product.php?pfid=12&prid=20
- SCO "Legend" Mailing List: Public
    - Legend-subscribe@list.sco.com
    - legend@sco.com
- Porting/Migration Alias:
    - osr5to6@sco.com
- Knowledge base:
    - http://wdb1.sco.com/kb/search