Linux Alternatives Framework

A New Binary Alternatives System

Dipl.-Math. Hans-Georg Eßer
Editor-in-chief
LinuxUser / EasyLinux
http://www.linux-user.de
http://www.easylinux.de

Power of Open Source: Alternatives

- *Edit a file?*
  vi, emacs, XEmacs, nedit, Kate, kwrite, joe, grep/sed/awk, ... (fm: 201 projects)
- *Write an e-mail?*
  mutt, elm, Mozilla, KMail, Evolution, Sylpheed, pine, ... (fm: 188 projects)
- *Browse the web?*
  Mozilla, Opera, w3m, lynx, links, Galeon, Konqueror, ... (fm: 182 projects)

Problem with choice: *choice*

- Imagine a start menu
- Imagine you´re new to Linux
- Try to pick a web browser, a mail client, or an editor from the menu...

"I just want to view a web page"

“Open this clickythingy in editor window”

- Some web browsers open page sources in an *editor* (for viewing)
- Some mail clients open external *editors* for writing a new mail
- `crontab -e` will open your crontab file in an *editor*

Competition: How many different *editors* can you open this way?

“Help” or “About”

- Many programs open a *web browser* to display the help (html) files
- Mail clients open a *web browser* when you click a link
- Your next “about” dialog might also launch a *web browser* and go to the project homepage

Competition: How many different *web browsers* can you open this way?

So you choose... and choose...

- Opera: *File/Preferences*, *Applications/ Program to view source code*
- mutt: Edit *~/.muttrc*:
  `set editor=...`
  or set $EDITOR environment variable
- Konqueror (file manager): *Preferences/ Setup Konqueror*, *File associations*, *text/plain*, *General*, *Order of programs*

Too much choice ...

- ... can confuse users
- ... is inconvenient when different UIs need to be understood
- ... requires configuration effort (time!) to make all programs behave equally

What to do about it?

The dumb solution: Remove choice

- Create a reduced distribution with one application per task only
- Force users to install only one application out of a choice of similar programs (remove others when a new one is installed)

→ loss of variety, one of the original strengths of Open Source software

The better solution: Define standard applications

- Keep all applications available
- Define standard tools for tasks
- Make all programs follow the definitions unless otherwise requested

→ How to do it?

Old solutions

- Environment variables:
  $EDITOR, $PAGER, ...
- Debian Alternatives:
  update-alternatives
- KDE configuration (only for KDE apps)
- “BlueCurve” desktop (remove choice)

Application classes (1/2)

- Program classes, e.g. “editor”, “browser”
- Fill with members (editor: vi, emacs, ...)
- Members have attributes:
  - isC (is a console application)
  - isX (is an X window application)
  - optC (extra options when started as console app)
  - optX (extra options when started as X app)

Application classes (2/2)

- Each class has
  - ... a default console application
  - ... a default X window application
- Auto-select default program, based on situation (we have X?)

LXA Class Definition

```
class editor {
  comment "Any kind of editors"
  defaultX emacs
  defaultC emacs
  member vi {
    comment "Visual editor"
    path /usr/bin/vi
    isX false
    isC true
  }
  member emacs {
    comment "Eight megs almost continuously swapping"
    path /usr/bin/emacs
    isX true
    isC true
    optC "-nw"
  }
}
```

lxa commands (1/2)

```
[a@b:/]# lxa create editor -c "Any kind of editors"
[a@b:/]# lxa add editor emacs -c "Eight megabytes" \
   -C -X --optX "-nw"
[a@b:/]# lxa add editor vi -c "Visual editor" -C
[a@b:/]# lxa defaultC editor vi
[a@b:/]# lxa defaultX editor emacs
[a@b:/]# lxa info editor
class editor [Any kind of editors]
 c-C-  vi      Visual editor
       /usr/lxa/alternatives/vi (/usr/bin/vi)
 cx-X  emacs   Eight megabytes
       /usr/lxa/alternatives/emacs (/usr/bin/emacs)
```

lxa commands (2/2)

Local configuration goes to ~/*.lxarc*

```
[a@b:/]$ lxa activate editor
[a@b:/]$ lxa deactivate editor

[a@b:~]$ lxa useglobal editor
```

Directory structure

- Member binaries move to
  /usr/lxa/alternatives/ (lxa add ...)
  (backup: e.g. */usr/bin/vi* → */usr/bin/.lxa/vi*)
- Old location links to
  /usr/lxa/bin/lxa-choose
- Chooser looks at class configuration
  /etc/lxa/classes.conf and local settings
  ~/.lxarc

Calling an application

- “nedit” is started
- Lookup: nedit is */usr/X11R6/bin/nedit*
- Link: → */usr/lxa/bin/lxa-choose*
- lxa-choose:
  - was called as “nedit”
  - looks up class database and local settings
  - determines X ($DISPLAY?) or console
  - finds: console, standard app: vi
  - launches vi

Implementation

- Quick and dirty Python implementation
- Almost feature-complete
- *lxa*: 626 lines, *lxa-choose*: 377 lines
- Download: http://lxa.hgesser.com

Integration with existing Linux distributions

- RPM/Debian: Pre/post-(un-)install *lxa* calls
- Behavior when installing: do not change standard X/console application
- Others: Use class sub-directories instead of class configuration file, e.g. */etc/lxa/classes/editor/vi*
- Distributor defines initial classes

Problems

- Integrity checks of *rpm*, *dpkg*, etc.
  → modify these tools
- Removing a package without removing a program´s class membership
- Ignorance of $PATH (standard application is called whether it was in path or not)
- Menu entries must be altered
- Unique names (two “vi”s?)

Future Extensions (1/2)

- More properties:
  Other than “X or console”, ask for things such as
  - Local execution (DISPLAY=remote?)
  - $HOME is NFS-mounted?
- Execution priorities:
  Give every application a priority instead of defining standard X/console apps

Future (2/2)

- Configuration GUI
  - root: create/modify classes
  - root: choose default applications
  - anyone: activate/deactivate class

Linux Alternatives Framework Contact

Speaker: Hans-Georg Eßer
Linux New Media AG
h.g.esser@linux-user.de

Source code: http://lxa.hgesser.com