Guix, the Computing Freedom Deployment Tool

Ludovic Courtès
ludo@gnu.org

GNU Hackers Meeting
22–25 August 2013, Paris

Salut !

Salut !

Salut !


GNU tower, Boston, MA


Services
Surface
Surface
team leader, GNU marketing dept.

Dependable. Hackable. Liberating.

Dependable.

per-user, transactional package installation etc.

```
alice@foo$ guix package --install=gcc
alice@foo$ guix gc --references `which gcc`
/nix/store/...-glibc-2.17
/nix/store/...-gcc-4.8.0
...

bob@foo$ guix package --install=gcc-4.7.3
bob@foo$ guix gc --references `which gcc`
/nix/store/...-glibc-2.13
/nix/store/...-gcc-4.7.3
...
```

demo!

transparent binary/source deployment

```
alice@foo$ guix package --install=emacs
The following package will be installed:
   emacs-24.3 out /nix/store/...-emacs-24.3

The following files will be downloaded:
   /nix/store/...-emacs-24.3
   /nix/store/...-libxpm-3.5.10
   /nix/store/...-libxext-1.3.1
   /nix/store/...-libxaw-1.0.11
```

transparent binary/source deployment

```
alice@foo$ guix package --install=emacs
The following package will be installed:
   emacs-24.3 out /nix/store/...-emacs-24.3

The following files will be downloaded:
   /nix/store/...-libxext-1.3.1
   /nix/store/...-libxaw-1.0.11
The following derivations will be built:
   /nix/store/...-emacs-24.3.drv
   /nix/store/...-libxpm-3.5.10.drv
```

transactional upgrades

```
$ guix package --upgrade
The following packages will be installed:
   emacs-24.3   out     /nix/store/...-emacs-24.3
   gdb-7.6      out     /nix/store/...-gdb-7.6
   geiser-0.4   out     /nix/store/...-geiser-0.4
   glibc-2.17   out     /nix/store/...-glibc-2.17
   guile-2.0.9  out     /nix/store/...-guile-2.0.9
...
```

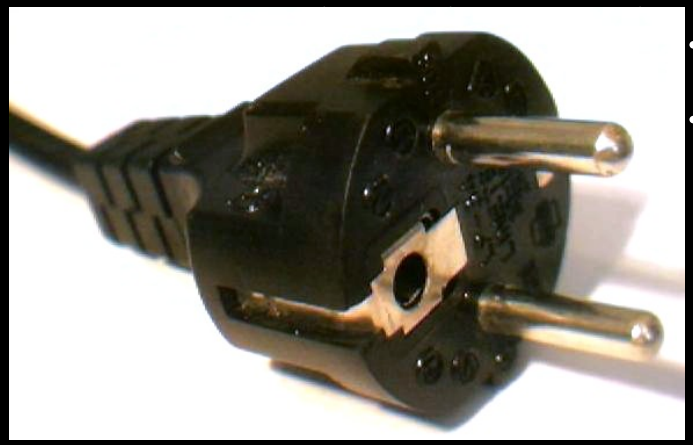
transactional upgrades

```
$ guix package --upgrade
The following packages will be installed:
   emacs-24.3   out     /nix/store/...-emacs-24.3
   gdb-7.6      out     /nix/store/...-gdb-7.6
   geiser-0.4   out     /nix/store/...-geiser-0.4
   glibc-2.17   out     /nix/store/...-glibc-2.17
   guile-2.0.9  out     /nix/store/...-guile-2.0.9
...
```

```
$ emacs --version ; guile --version
GNU Emacs 24.3.1
guile (GNU Guile) 2.0.9
```

transactional upgrades

```
$ guix package --upgrade
The following packages will be installed:
   emacs-24.3   out     /nix/store/...-emacs-24.3
   gdb-7.6      out     /nix/store/...-gdb-7.6
   geiser-0.4   out     /nix/store/...-geiser-0.4
   glibc-2.17                          ..-glibc-2.17
   guile-2.0.9                         ..-guile-2.0.9
...
```



transactional upgrades

```
$ guix package --upgrade
The following packages will be installed:
   emacs-24.3   out     /nix/store/...-emacs-24.3
   gdb-7.6      out     /nix/store/...-gdb-7.6
   geiser-0.4   out     /nix/store/...-geiser-0.4
   glibc-2.17   out     /nix/store/...-glibc-2.17
   guile-2.0.9  out     /nix/store/...-guile-2.0.9
...
```

(**interrupted right in the middle**)

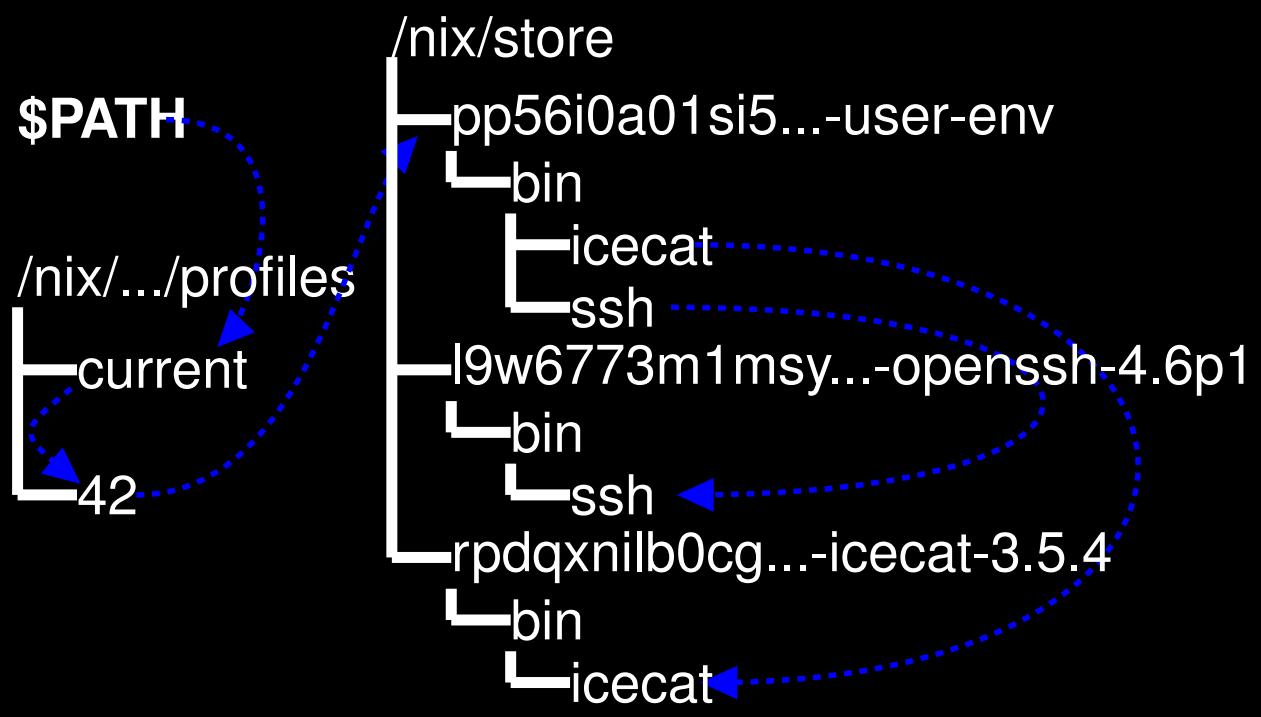
```
$ emacs --version ; guile --version
GNU Emacs 23.2
guile (GNU Guile) 1.8.8
```

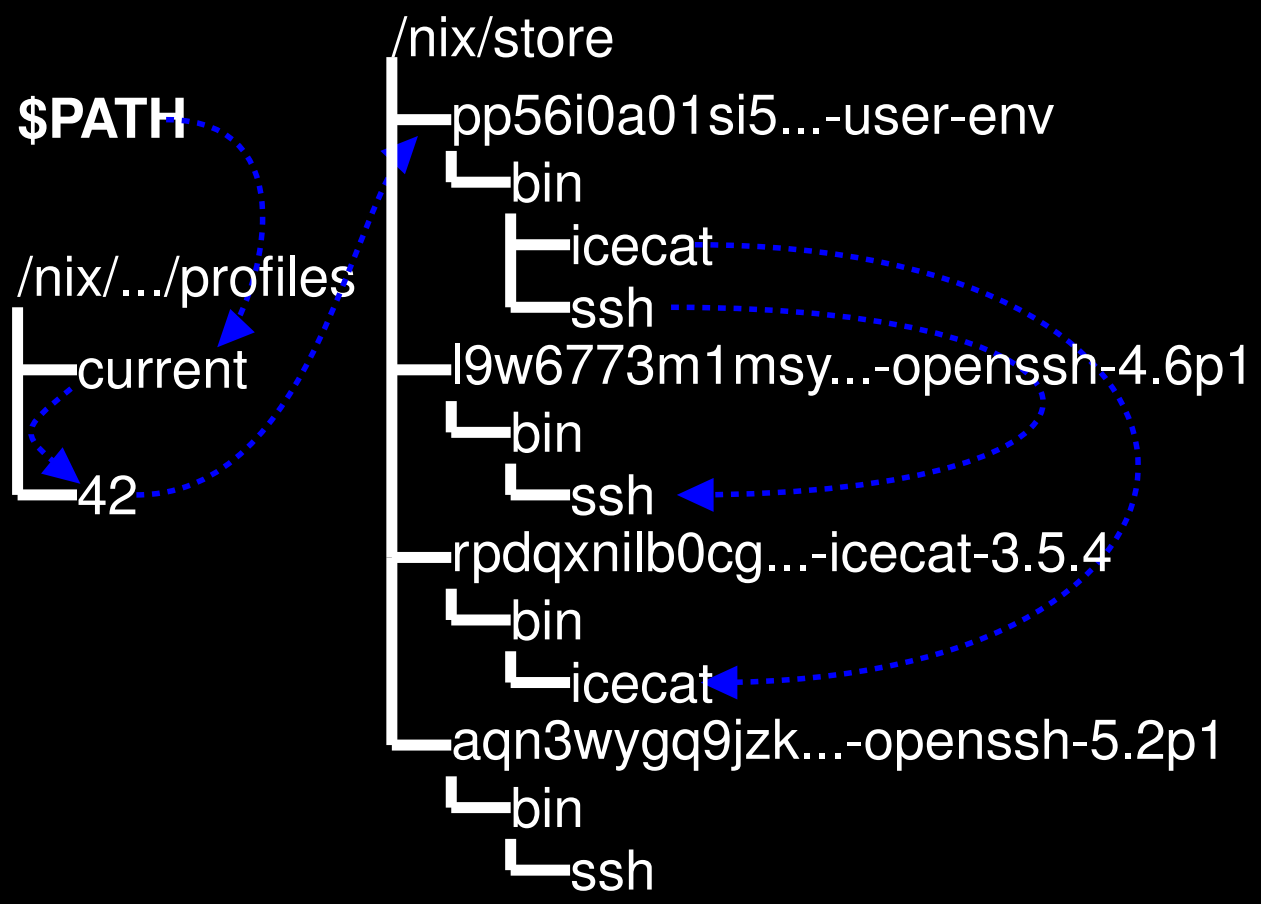
transactional upgrades

```
$ guix package --upgrade
The following packages will be installed:
   emacs-24.3   out     /nix/store/...-emacs-24.3
   gdb-7.6      out     /nix/store/...-gdb-7.6
   geiser-0.4   out     /nix/store/...-geiser-0.4
   glibc-2.17   out     /nix/store/...-glibc-2.17
   guile-2.0.9  out     /nix/store/...-guile-2.0.9
...
```

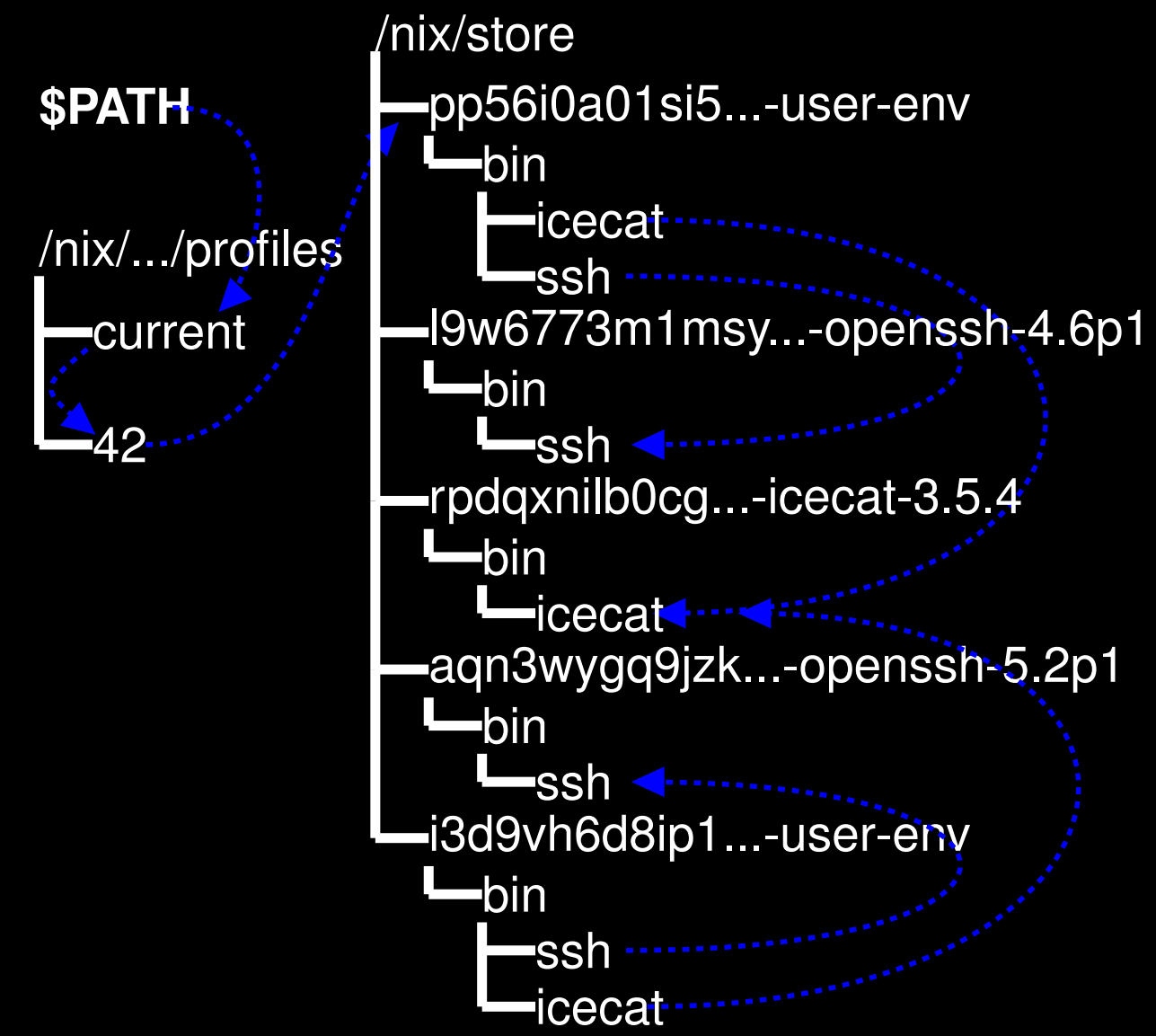
(**interrupted right in the middle**)

```
$ emacs --version ; guile --version
GNU Emacs 23.2
guile (GNU Guile) 1.8.8
```

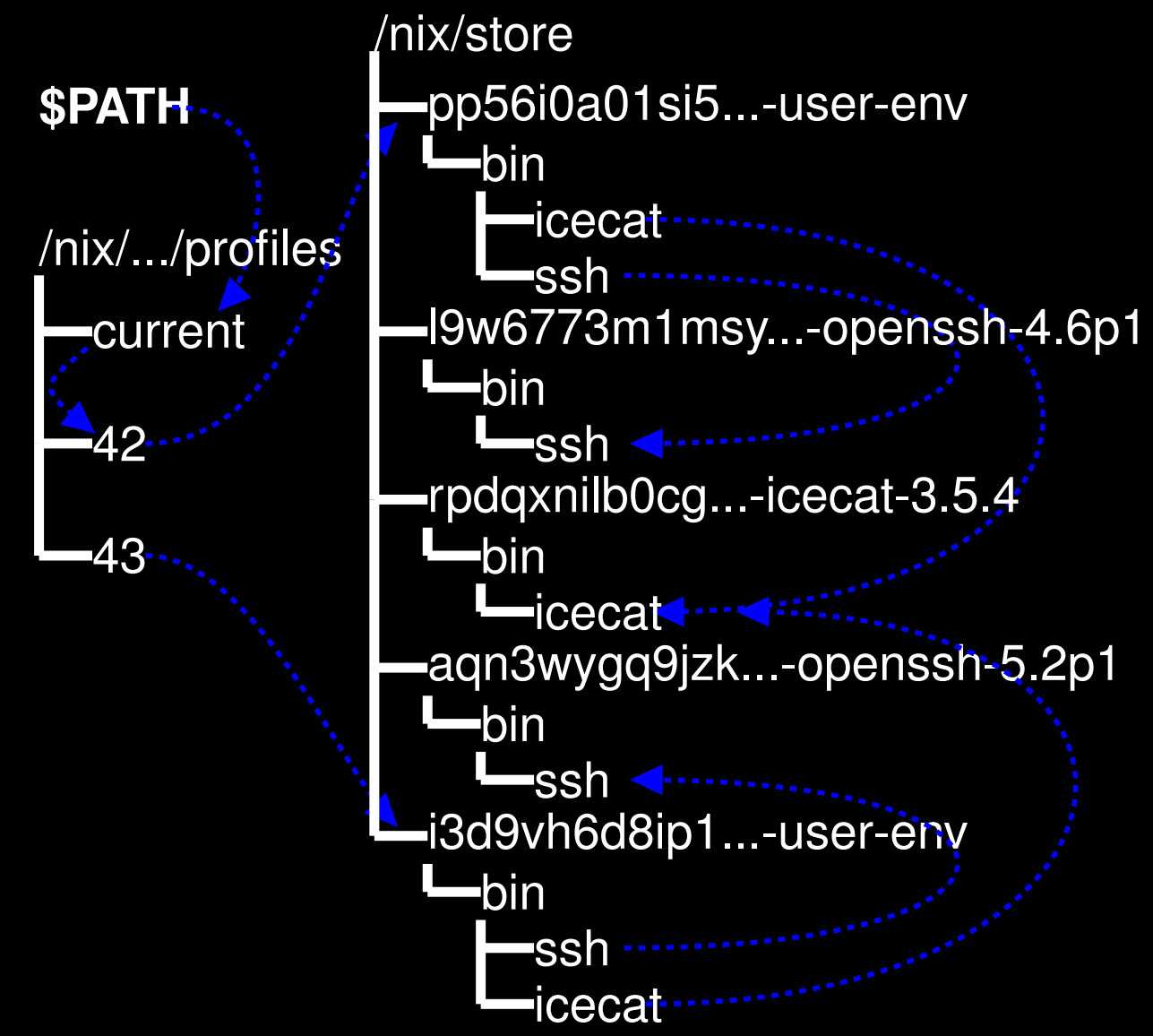

$PATH
/nix/.../profiles
current
42
/nix/store
pp56i0a01si5...-user-env
bin
icecat
ssh
l9w6773m1msy...-openssh-4.6p1
bin
ssh
rpdqxnilb0cg...-icecat-3.5.4
bin
icecat



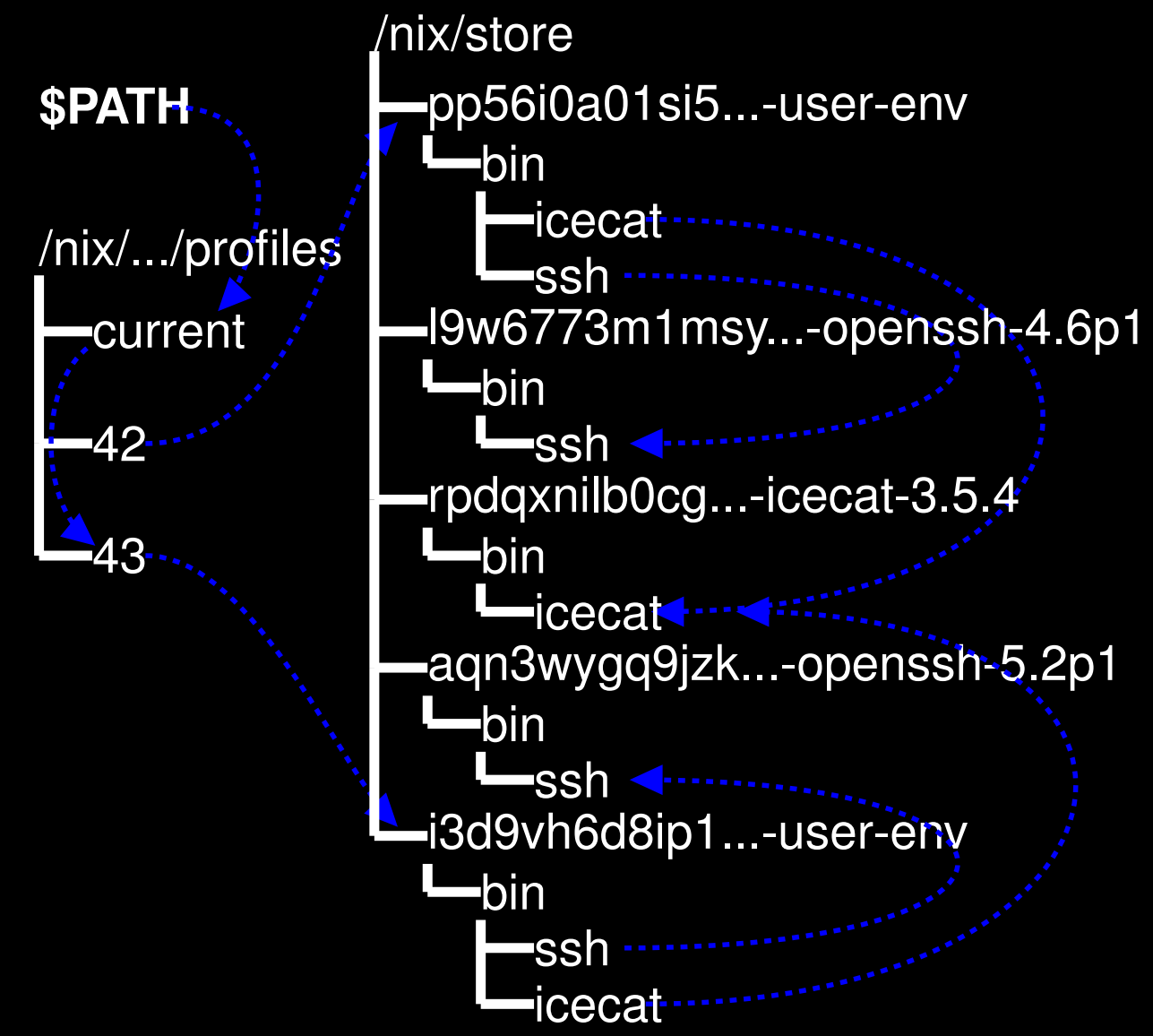

```
guix package --upgrade=openssh
```



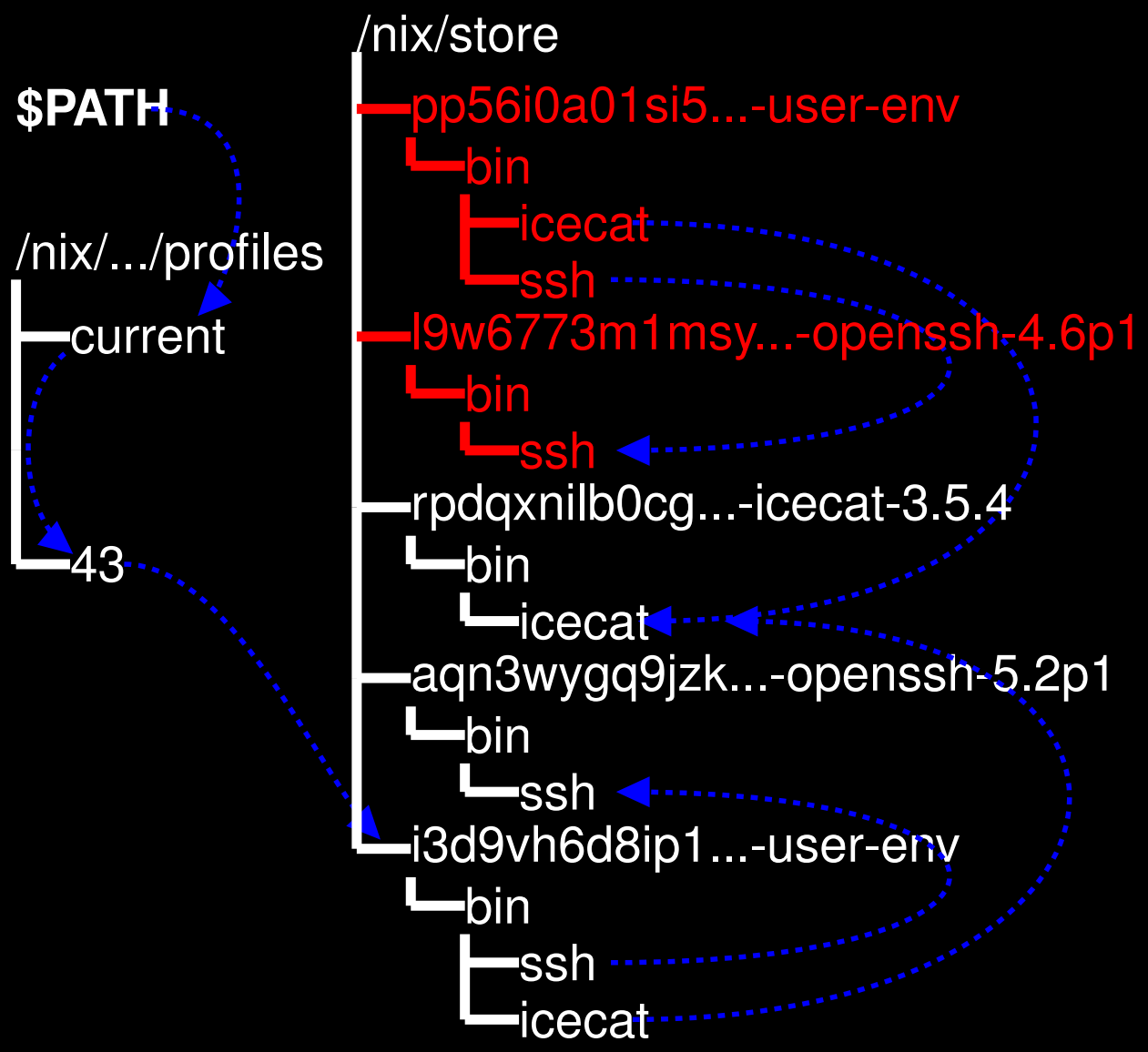

```
guix package --upgrade=openssh
```

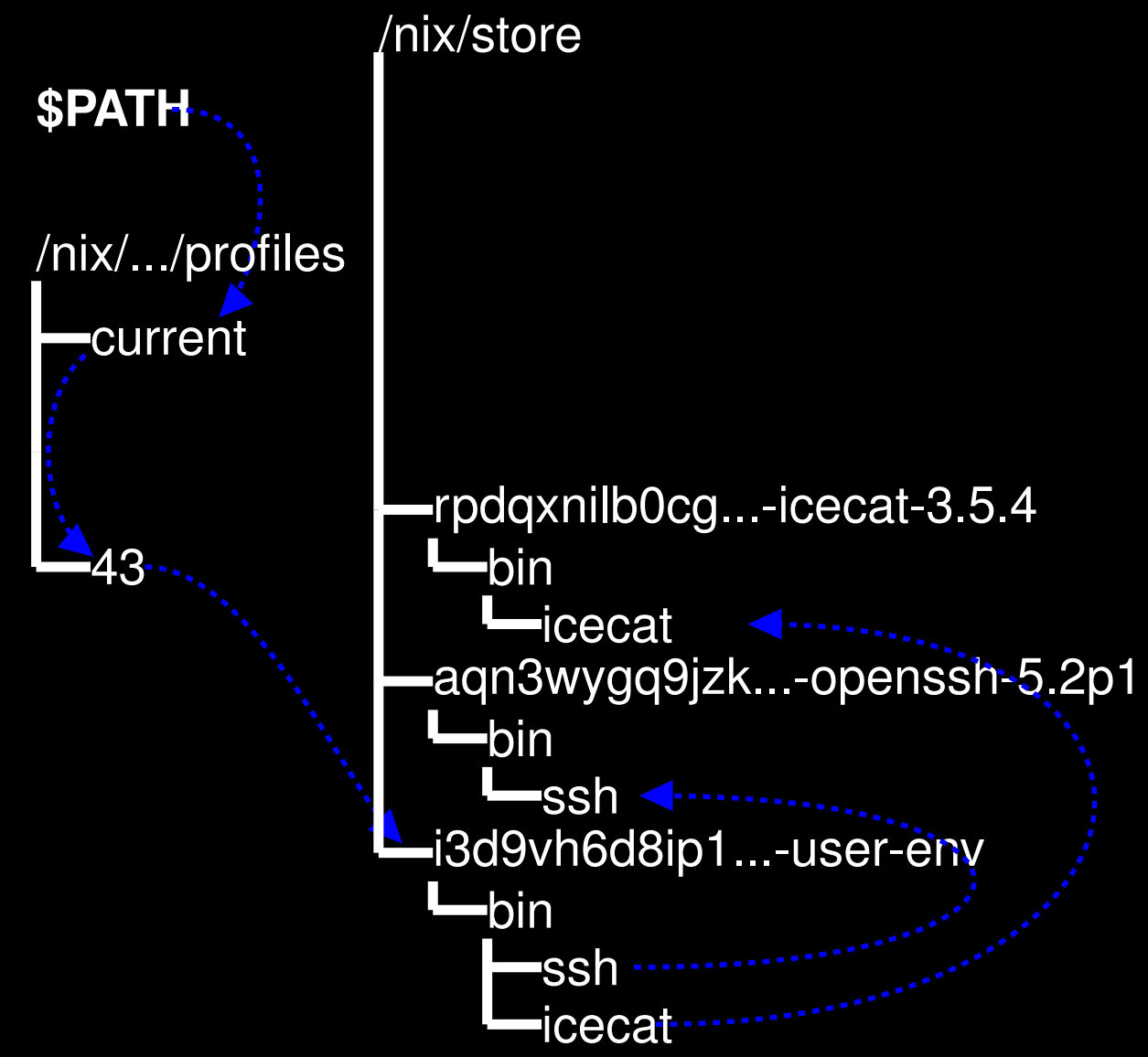



```
guix package --upgrade=openssh
```




```
guix package --upgrade=openssh
```


/nix/store
$PATH
pp56i0a01si5...-user-env
bin
icecat
ssh
/nix/.../profiles
current
l9w6773m1msy...-openssh-4.6p1
bin
ssh
rpdqxnilb0cg...-icecat-3.5.4
bin
icecat
43
aqn3wygq9jzk...-openssh-5.2p1
bin
ssh
i3d9vh6d8ip1...-user-env
bin
ssh
icecat


/nix/store
$PATH
/nix/.../profiles
current
43
rpdqxnilb0cg...-icecat-3.5.4
bin
icecat
aqn3wygq9jzk...-openssh-5.2p1
bin
ssh
i3d9vh6d8ip1...-user-env
bin
ssh
icecat

```
guix gc
```

rollback

```
$ emacs --version
GNU Emacs 24.2

$ guix package --upgrade=emacs
The following packages will be installed:
   emacs-24.3.1 out /nix/store/...-emacs-24.3.1
...

$ emacs --version
Segmentation Fault

$ guix package --roll-back
switching from generation 43 to 42

$ emacs --version
GNU Emacs 24.2
```

demo!

Hackable.

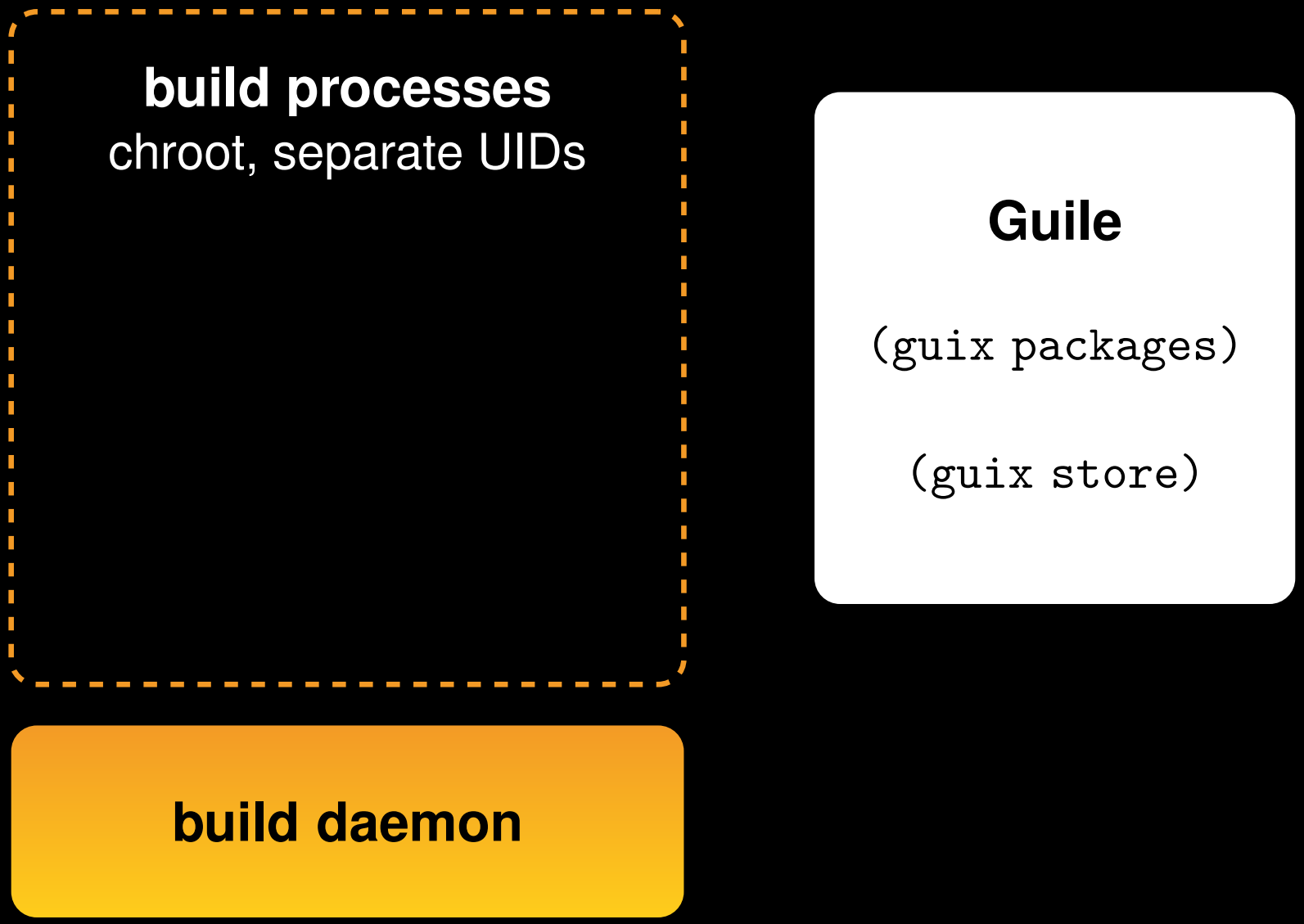
```
<project xmlns="http://guix.gnu.org/POM/0.0.1"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://guix.gnu.org/POM/0.0.1
                      http://guix.gnu.org/xsd/guix-0.0.1.xsd">
  <modelVersion>0.0.1</modelVersion>

  <!-- The Basics -->
  <groupId>...</groupId>
  <artifactId>...</artifactId>
  <version>...</version>
  <packaging>...</packaging>
  <dependencies>...</dependencies>
  <parent>...</parent>
  <dependencyManagement>...</dependencyManagement>
  <modules>...</modules>
  <properties>...</properties>

  <!-- Build Settings -->
  <build>...</build>
  <reporting>...</reporting>

  <!-- More Project Information -->
  <name>...</name>
  <description>...</description>
```

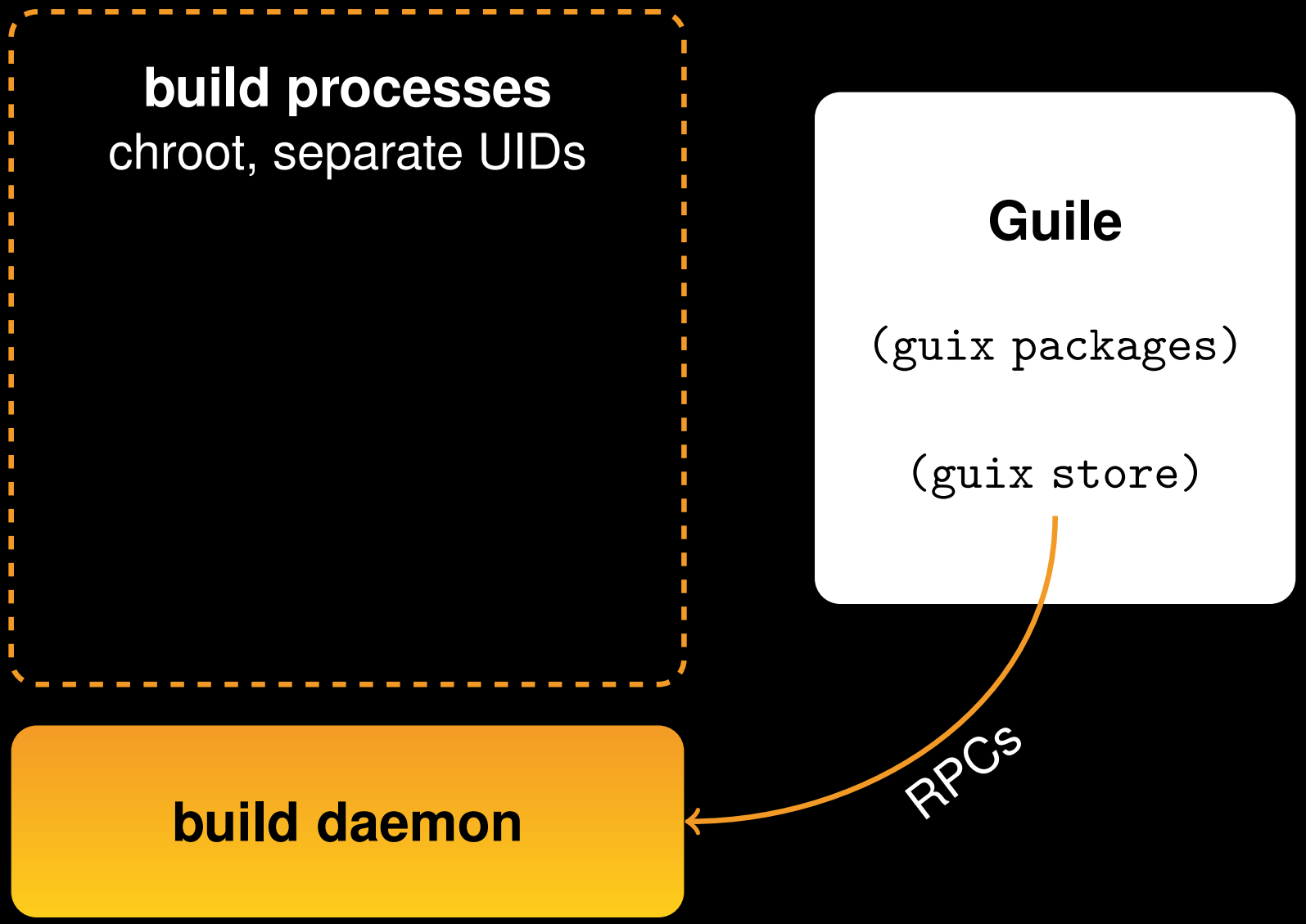
The truth is that Lisp is not the right language for any particular problem. Rather, Lisp encourages one to attack a new problem by implementing new languages tailored to that problem.

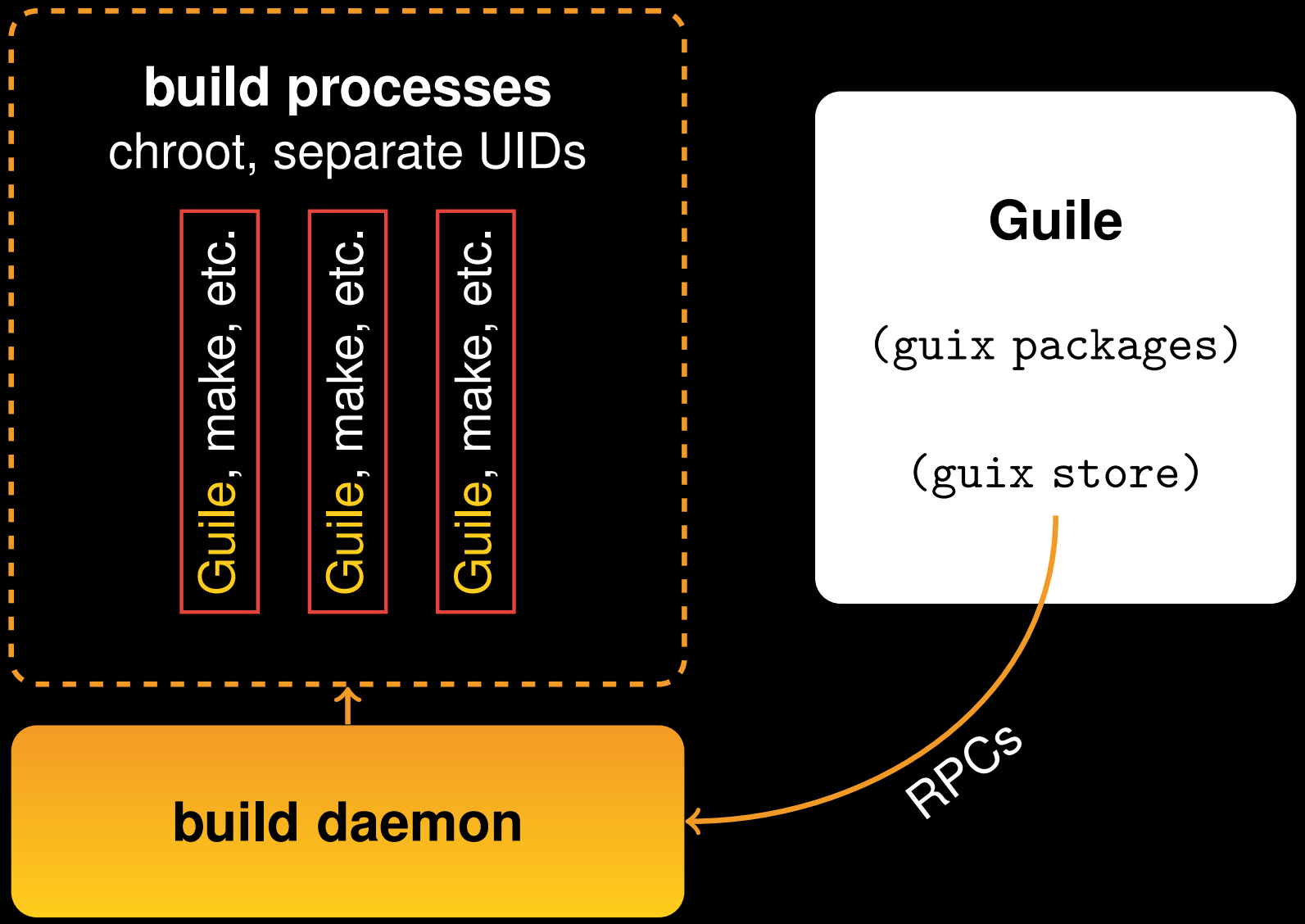
– Abelson & Sussman, 1987

```
(define hello
  (package
   (name "hello")
   (version "2.8")
   (source (origin
            (method url-fetch)
            (uri (string-append
                  "mirror://gnu/.../hello-" version
                  ".tar.gz"))
            (sha256 (base32 "0wqd...z6"))))
   (build-system gnu-build-system)
   (synopsis "Hello, GNU world: An example GNU package")
   (description "Produce a friendly greeting.")
   (home-page "http://www.gnu.org/software/hello/")
   (license gpl3+)))
```

Emacs + Geiser demo!


build processes
chroot, separate UIDs
Guile
(guix packages)
(guix store)
build daemon


build processes
chroot, separate UIDs
Guile
(guix packages)
(guix store)
RPCs
build daemon


build processes
chroot, separate UIDs
Guile, make, etc.
Guile, make, etc.
Guile, make, etc.
Guile
(guix packages)
(guix store)
RPCs
build daemon

```
(use-modules (guix packages) (guix store)
             (gnu packages base))

(define store
  (open-connection))

(package? hello)
=> #t

(define drv (package-derivation store hello))
drv
=> "/nix/store/xyz...-hello-2.8.drv"

(build-derivations (list drv))
```

... daemon builds/downloads package on our behalf...

```
=> "/nix/store/pqr...-hello-2.8"
```

demo!

copy fields from `hello` except for `version` and `source`

```
(package ( inherit hello)
  (version "2.7")
  (source
    (origin
      (method url-fetch)
      (uri "mirror://gnu/hello/hello-2.7.tar.gz")
      (sha256
        (base32 "7dqw3...")))))
```

```
(define (static-package p)
  ;; Return a statically-linked variant of P.
  (package (inherit p)
    (arguments
     `(#:configure-flags '("--disable-shared"
                           "LDFLAGS=-static")
       ,@(package-arguments p)))))
```

builder side of `gnu-build-system`

```
(define %standard-phases
  `((configure . ,configure)
    (build . ,build)
    ;; …
    ))

(define* (gnu-build #:key (phases %standard-phases)
                    #:allow-other-keys
                    #:rest args)
  ;; Run all the PHASES in order, passing them ARGS.
  (every (match-lambda
           ((name . proc)
            (format #t "starting phase `~a'~%" name)
            (let ((result (apply proc args)))
              (format #t "phase `~a' done~%" name)
              result)))
         phases))
```

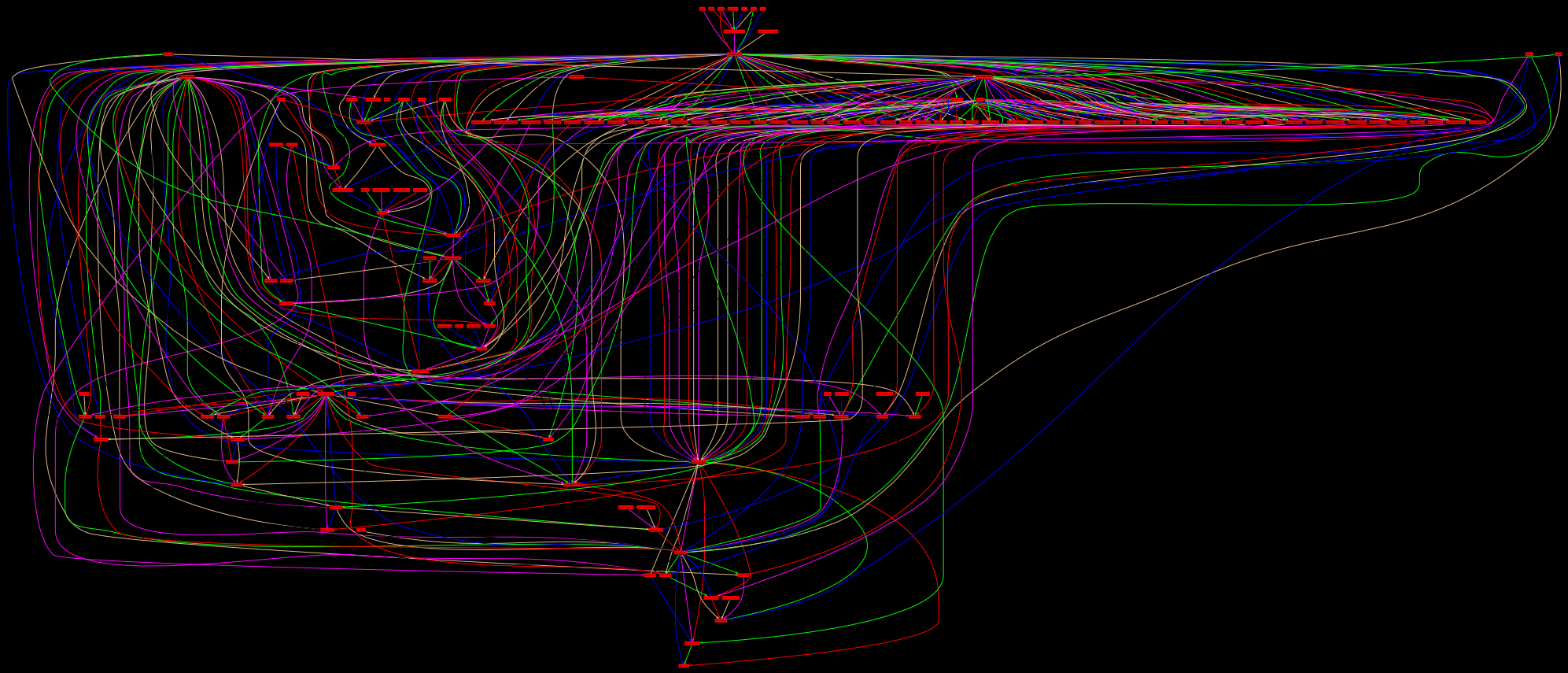
Liberating.

GNU system = 100% **libre** software

The "Corresponding Source" for a work in object code form means **all the source code needed to generate**, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities.

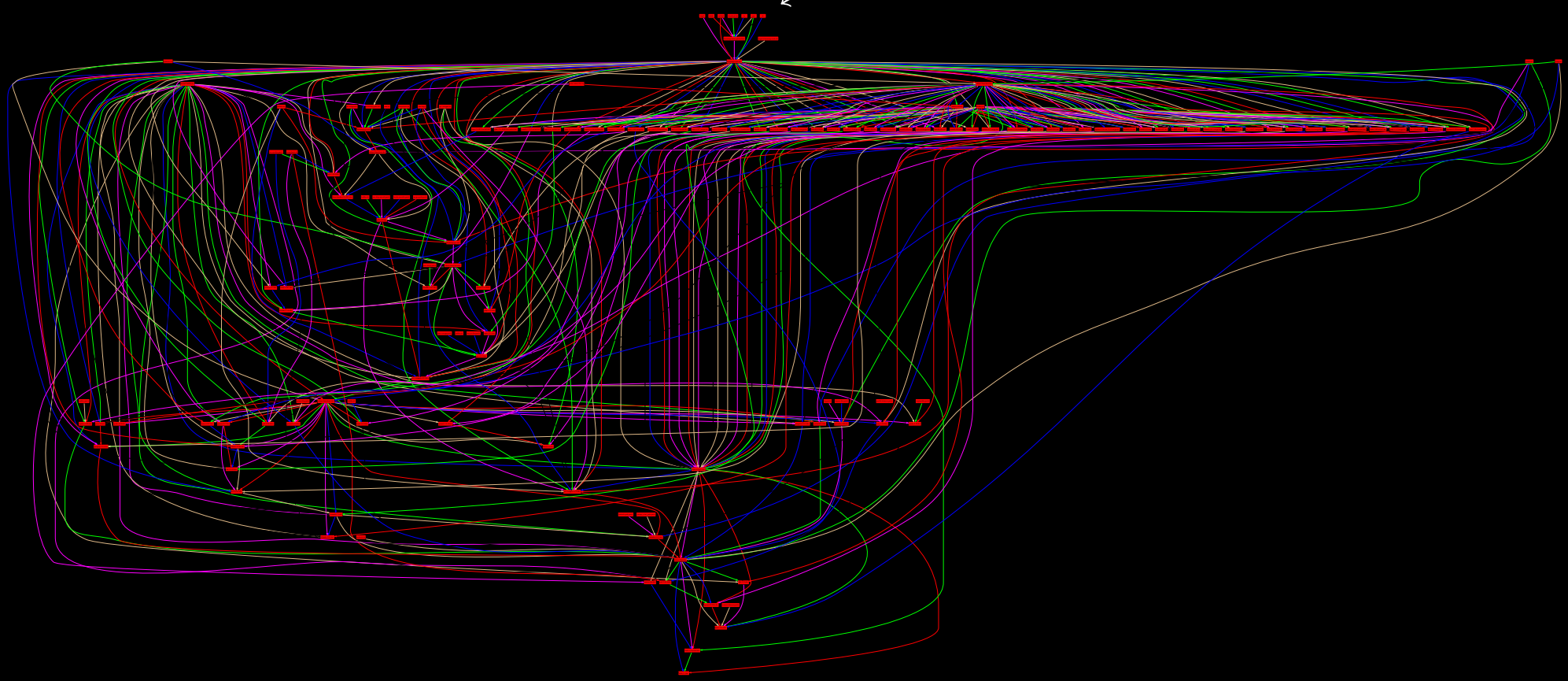
The "Corresponding Source" for a work in object code form means **all the source code needed to generate, install, and (for an executable work) run the object code** and to modify the work, including scripts to control those activities.

Guix users get the Corresponding Source



build-time dependencies of GNU Hello

bootstrap binaries



build-time dependencies of GNU Hello


boot strap

```
(origin
  (method  url-fetch )
  (uri (string-append "mirror://gnu/gcc/gcc-"
                      version "/gcc-" version
                      ".tar.bz2"))
  (sha256 (base32 "1hx9…")))
```

use Guile HTTP(S)/FTP client

```
(origin
  (method  url-fetch )
  (uri (string-append "mirror://gnu/gcc/gcc-"
                      version "/gcc-" version
                      ".tar.bz2"))
  (sha256 (base32 "1hx9...")))
```

use Guile HTTP(S)/FTP client

```
(origin
  (method  url-fetch )
  (uri (string-append "mirror://gnu/gcc/gcc-"
                      version "/gcc-" version
                      ".tar.bz2"))
  (sha256 (base32 "1hx9...")))
```

how is the very first tarball downloaded?

bootstrapping the distribution

0. statically-linked binaries of `mkdir`, `tar`, `xz`, `bash`, and Guile

bootstrapping the distribution

0. statically-linked binaries of `mkdir`, `tar`, `xz`, `bash`, and Guile
1. derivation runs Bash script to untar Guile

bootstrapping the distribution

0. statically-linked binaries of `mkdir`, `tar`, `xz`, `bash`, and Guile
1. derivation runs Bash script to untar Guile
2. use Guile to download statically-linked binaries of GCC, Binutils, libc, Coreutils et al., and Bash

bootstrapping the distribution

0. statically-linked binaries of `mkdir`, `tar`, `xz`, `bash`, and Guile
1. derivation runs Bash script to untar Guile
2. use Guile to download statically-linked binaries of GCC, Binutils, libc, Coreutils et al., and Bash
3. use that to build GNU Make

bootstrapping the distribution

0. statically-linked binaries of `mkdir`, `tar`, `xz`, `bash`, and Guile
1. derivation runs Bash script to untar Guile
2. use Guile to download statically-linked binaries of GCC, Binutils, libc, Coreutils et al., and Bash
3. use that to build GNU Make
4. build a tool chain independent of the bootstrap binaries

bootstrapping the distribution

0. statically-linked binaries of `mkdir`, `tar`, `xz`, `bash`, and Guile
1. derivation runs Bash script to untar Guile
2. use Guile to download statically-linked binaries of GCC, Binutils, libc, Coreutils et al., and Bash
3. use that to build GNU Make
4. build a tool chain independent of the bootstrap binaries

what led to the binaries in step 0?

```
$ guix build bootstrap-tarballs
/nix/store/...-bootstrap-tarballs-0
```

```
$ guix build bootstrap-tarballs
/nix/store/...-bootstrap-tarballs-0
```

porting to new arches:

```
$ guix build bootstrap-tarballs \
     --target=mips64el-linux-gnuabi64
```

Does this binary **correspond** to that source?

```
$ guix build guile
```

```
$ guix build guile
/nix/store/ h2g4sc09h4... -guile-2.0.9
```

hash of **all** the dependencies

```
$ guix build guile
/nix/store/ h2g4sc09h4... -guile-2.0.9

$ guix gc --references /nix/store/...-guile-2.0.9
/nix/store/4jl83jgzaac...-glibc-2.17
/nix/store/iplay43cg58...-libunistring-0.9.3
/nix/store/47p47v92cj9...-libffi-3.0.9
/nix/store/drkwck2j965...-gmp-5.0.5
...
```

```
$ guix build guile
/nix/store/ h2g4sc09h4... -guile-2.0.9

$ guix gc --references /nix/store/...-guile-2.0.9
/nix/store/4jl83jgzaac...-glibc-2.17
/nix/store/iplay43cg58...-libunistring-0.9.3
/nix/store/47p47v92cj9...-libffi-3.0.9
/nix/store/drkwck2j965...-gmp-5.0.5
...
```

nearly bit-identical for everyone

controlled build environment

1. one directory **per installed package**
2. **immutable** installation directories
3. undeclared dependencies **invisible** to the build process
4. **isolated build**: chroot, separate UID, etc.

Lively!

Shipping is a feature.
A really important feature.

– Joel Spolsky

timeline

- July 2012 — GHM, Düsseldorf
- Nov. 2012 — dubbed GNU
- Jan. 2013 — 0.1
  - self-contained
  - ≈150 packages
- Feb. 2013 — Boot-to-Guile
- May 2013 — 0.2
  - package upgrade, search, update
  - binary substituter
  - ≈350 packages
- June 2013 — European Lisp Symposium
- July 2013 — 0.3
  - cross-compilation, debug info
  - ≈400 packages

status

- full-featured package manager
- self-contained distro, 450+ packages, 2 platforms
- binaries built & served at `http://hydra.gnu.org`
- tooling: auto-update, sync with GNU Womb, etc.
- l10n: 3 languages!

status

- community!
  - chances are your neighbor is a Guix dev!

status

- community!
  - chances are your neighbor is a Guix dev!
  - 4+ regular contributors

status

- community!
  - chances are your neighbor is a Guix dev!
  - 4+ regular contributors
- active repo, active mailing list, IRC channel
- `http://bugs.gnu.org/guix` + `guix-devel@gnu.org`

pushing the limits: booting to Guile

```
(expression->initrd
 '(begin
    (mkdir "/proc")
    (mount "none" "/proc" "proc")

    ;; Load Linux kernel modules.
    (let ((slurp (lambda (module)
                   (call-with-input-file
                     (string-append "/modules/" module)
                     get-bytevector-all))))
      (for-each (compose load-linux-module slurp)
                (list "md4.ko" "ecb.ko" "cifs.ko")))

    ;; Turn eth0 up.
    (let ((sock (socket AF_INET SOCK_STREAM 0)))
      (set-network-interface-flags sock "eth0" IFF_UP))

    ;; At last, the warm and friendly REPL.
    (start-repl)))
```

demo!

time to boot into a GNU system...

road map

1. QEMU image that boots to the DMD init system (real soon)

road map

1. QEMU image that boots to the DMD init system (real soon)
2. simple installer ISO image (fall 2013)

road map

1. QEMU image that boots to the DMD init system (real soon)
2. simple installer ISO image (fall 2013)
3. NixOS-style whole-system configuration EDSL (winter 2013)

you can help!

- **install it** atop your current distro
- **use it**, report bugs
- add your GNU & favorite **packages** to the distro
- **port** to other hardware
- help with the **infrastructure**
- share your **ideas** for the GNU system!

ludo@gnu.org

http://gnu.org/software/guix/

credits

- Boston skyline, CC-BY-SA 3.0, http://en.wikipedia.org/wiki/File:Boston_skyline_at_earlymorning.jpg
- “GNU marketing dept.” picture by the Free Software Foundation, http://www.fsf.org/news/gnu-comes-bearing-gifts-draws-shoppers-from-windows-store
- boots with a strap, CC-BY-SA 3.0, http://en.wikipedia.org/wiki/File:Dr_Martens,_black,_old.jpg
- birthday cake, http://openclipart.org/detail/5595/birthday-cake-by-dstankie
- GNU head, GFDL, http://www.gnu.org/graphics/agnuhead.html

The source of this document is available from http://git.sv.gnu.org/cgit/guix/maintenance.git.