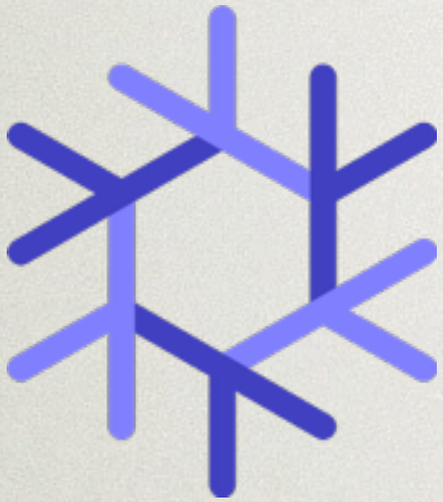
The Nix/Hydra System and hep-nix-overlay

Ian-Woo Kim (PH-TH)
ian-woo.kim@cern.ch

PH-SFT group meeting
CERN, 22 Sep 2014

NIX PACKAGE MANAGER



Nix logo (nixos.org)

Eelco Dolstra

- Created by Eelco Dolstra
- Started in 2004

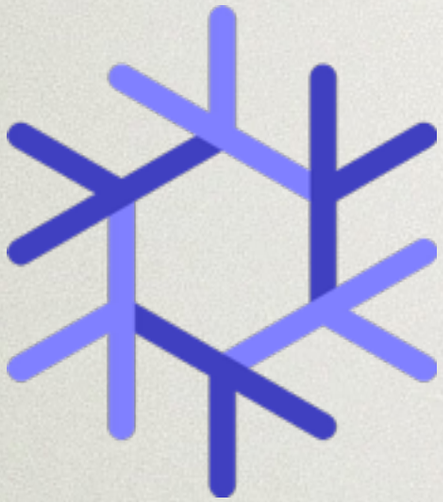
Nix: A Safe and Policy-Free System for Software Deployment

Eelco Dolstra, Merijn de Jonge, and Eelco Visser – Utrecht University

- Ph.D. Thesis, 2006

The Purely Functional Software Deployment Model

NIX PACKAGE MANAGER



Nix logo (nixos.org)

Eelco Dolstra

- Created by Eelco Dolstra
- Started in 2004

Nix: A Safe and Policy-Free System for Software Deployment

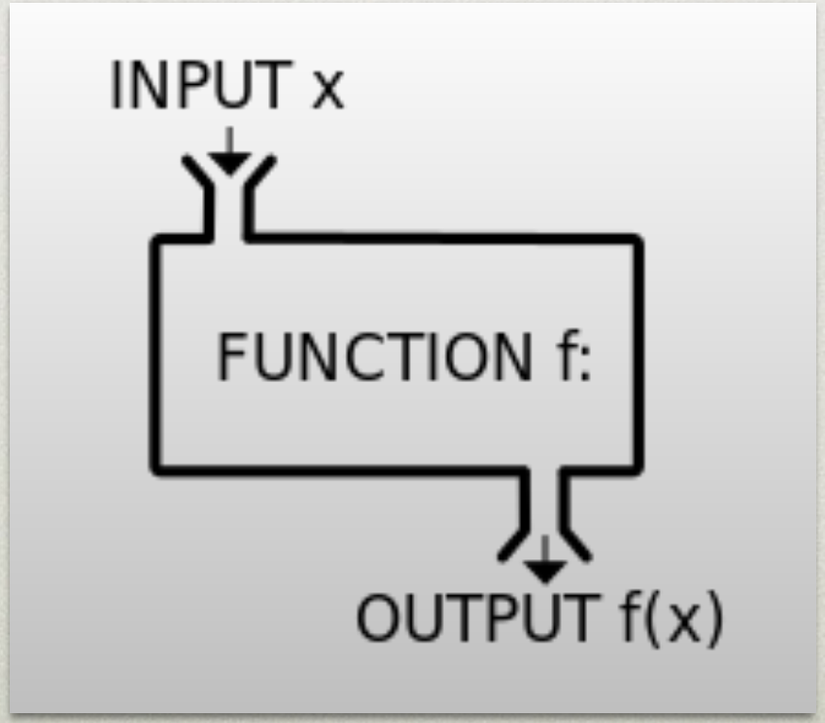
Eelco Dolstra, Merijn de Jonge, and Eelco Visser – Utrecht University

- Ph.D. Thesis, 2006

The Purely Functional Software Deployment Model

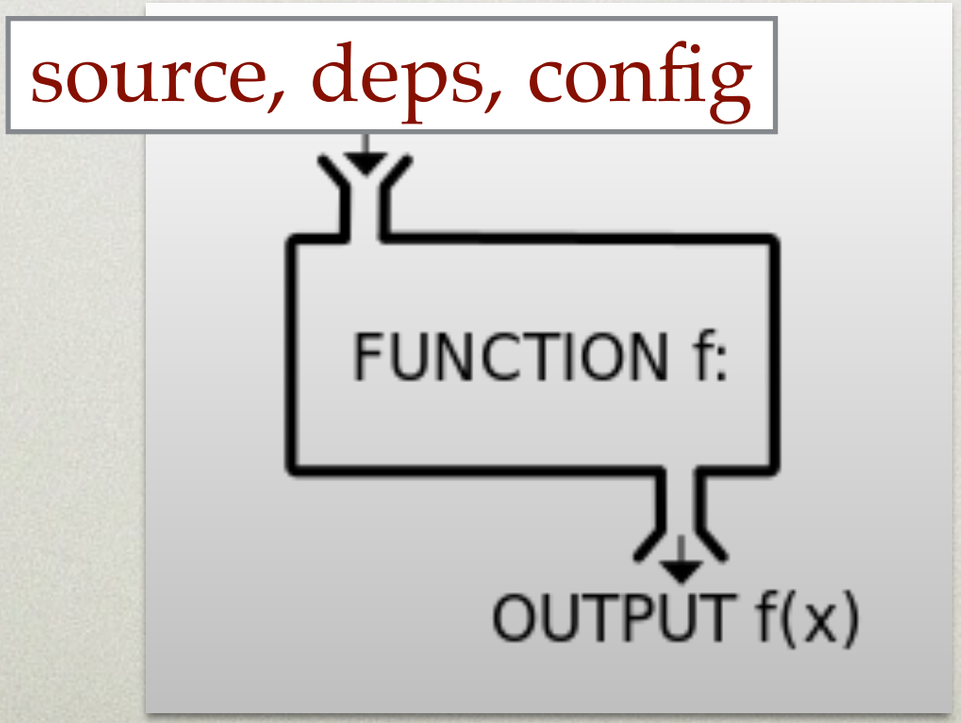
Purely Functional?

- A pure function = a function in mathematical sense.
  - The same inputs give the same outputs.



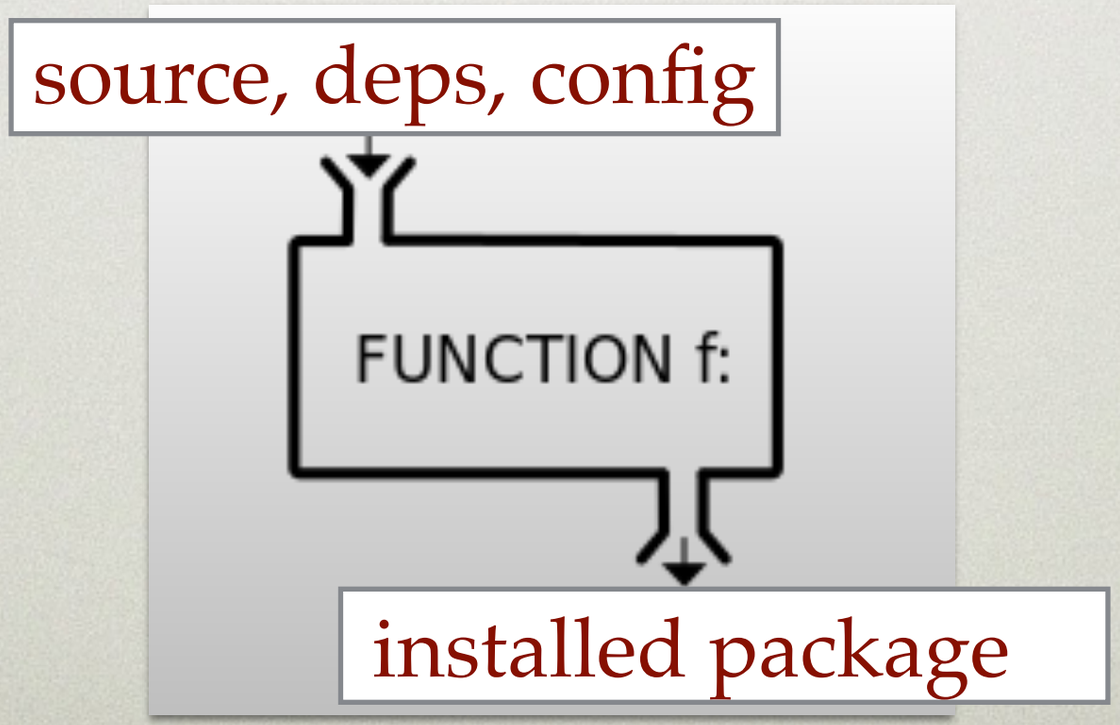
Purely Functional?

- A pure function = a function in mathematical sense.
  - The same inputs give the same outputs.



Purely Functional?

- A pure function = a function in mathematical sense.
  - The same inputs give the same outputs.

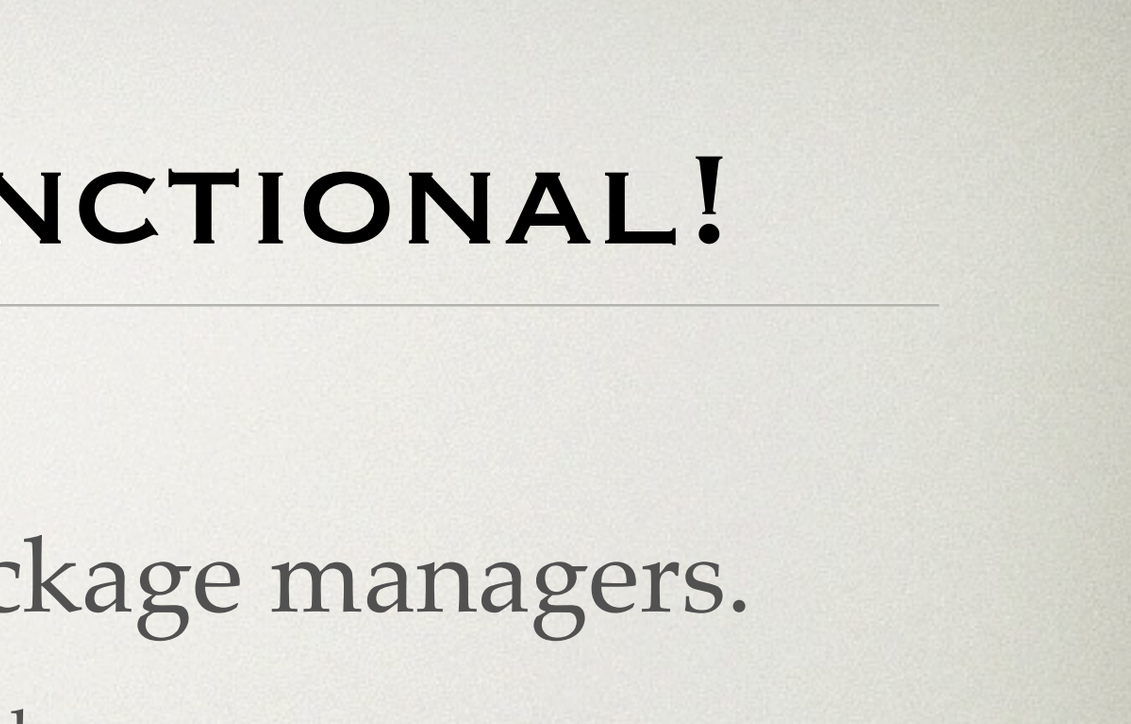


Purely Functional!

- Dream of build/package managers.
  - Reproducibility guaranteed
  - Can identify build problems
  - Share across systems

Many package managers try to achieve this by fixing *the same build environment.*

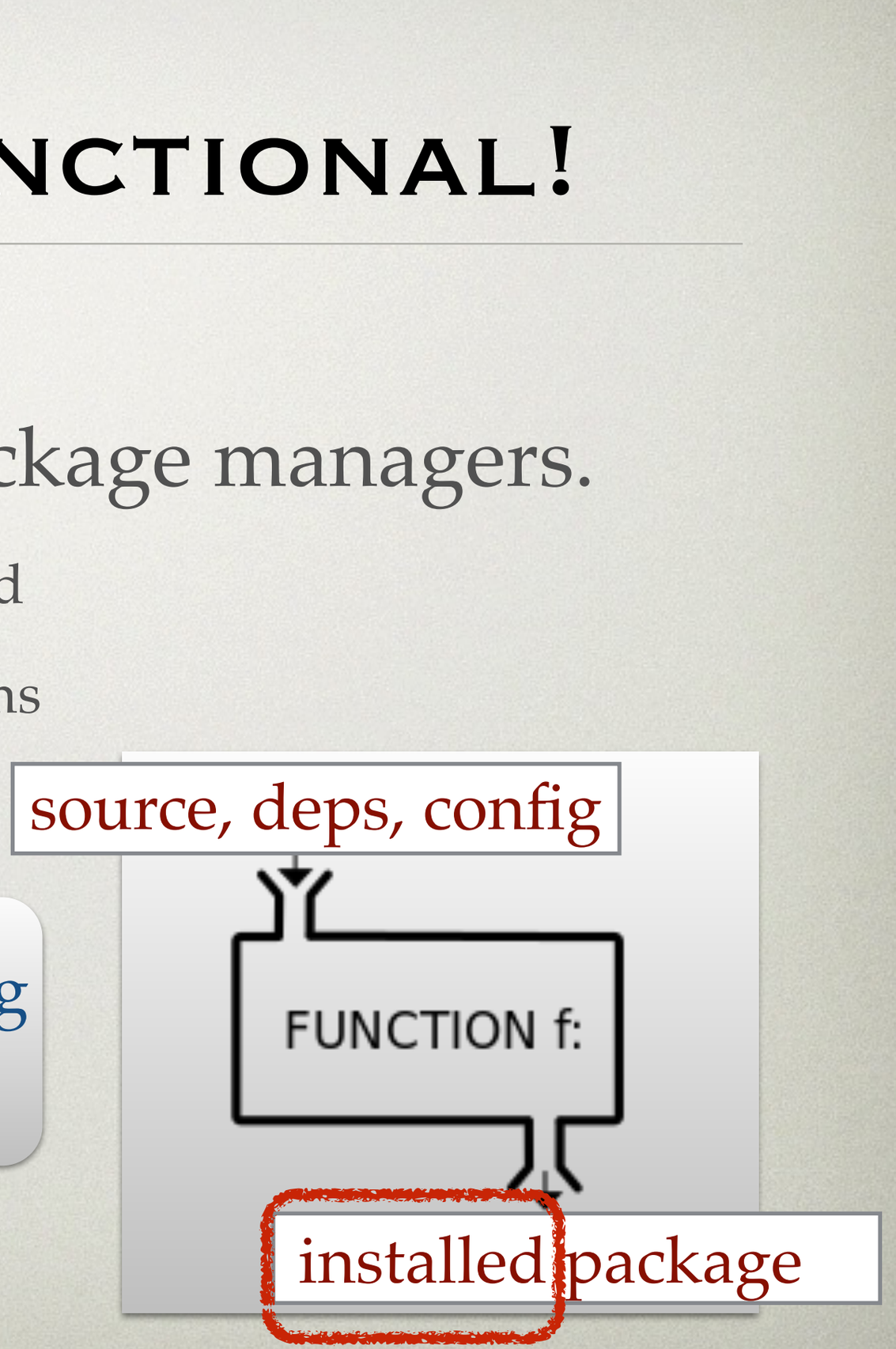
The resultant package file can be the same but the resultant system still differs by package installation history…



Purely Functional!

- Dream of build/package managers.
  - Reproducibility guaranteed
  - Can identify build problems
  - Share across systems

Nix achieve this by **NOT** allowing *to overwrite previous installation.*



Everything is hashed

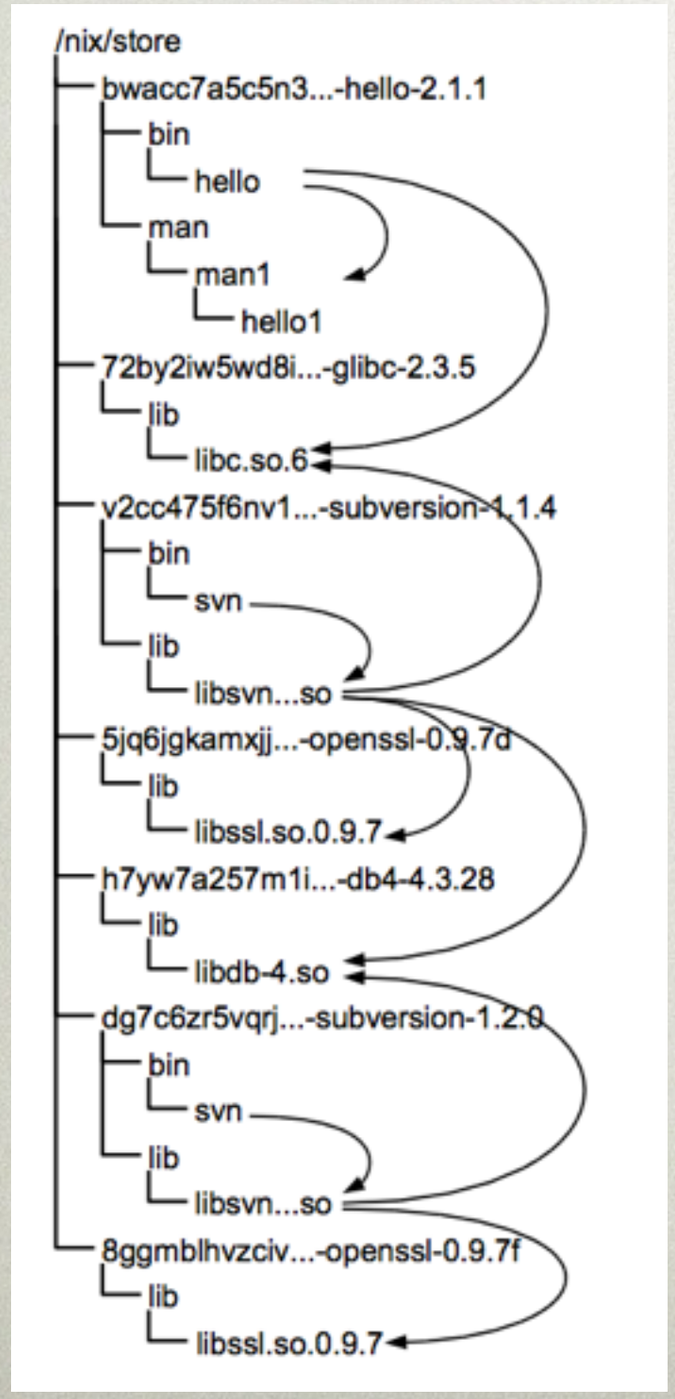
- Nix packages are installed in nix store with hash. ex) GNU hello

```
/nix/store/whi5gpcqxx29dg89ffjdi0dg9rb5i0s2-hello-2.9
```

Nix store location | Hash | Package name, version, flavor

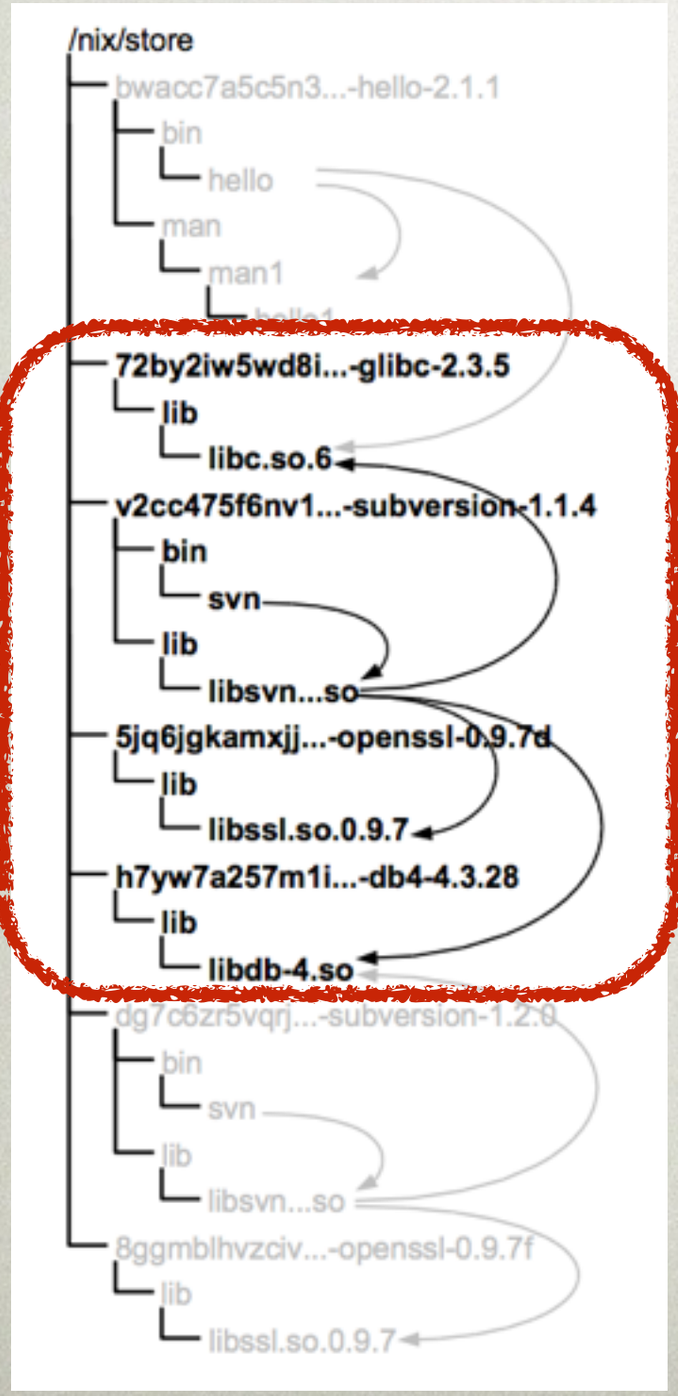
- The same packages with different version / dependency / configuration can coexist.
- User's profile is a selection of packages aggregated into a common prefix.

Everything is hashed



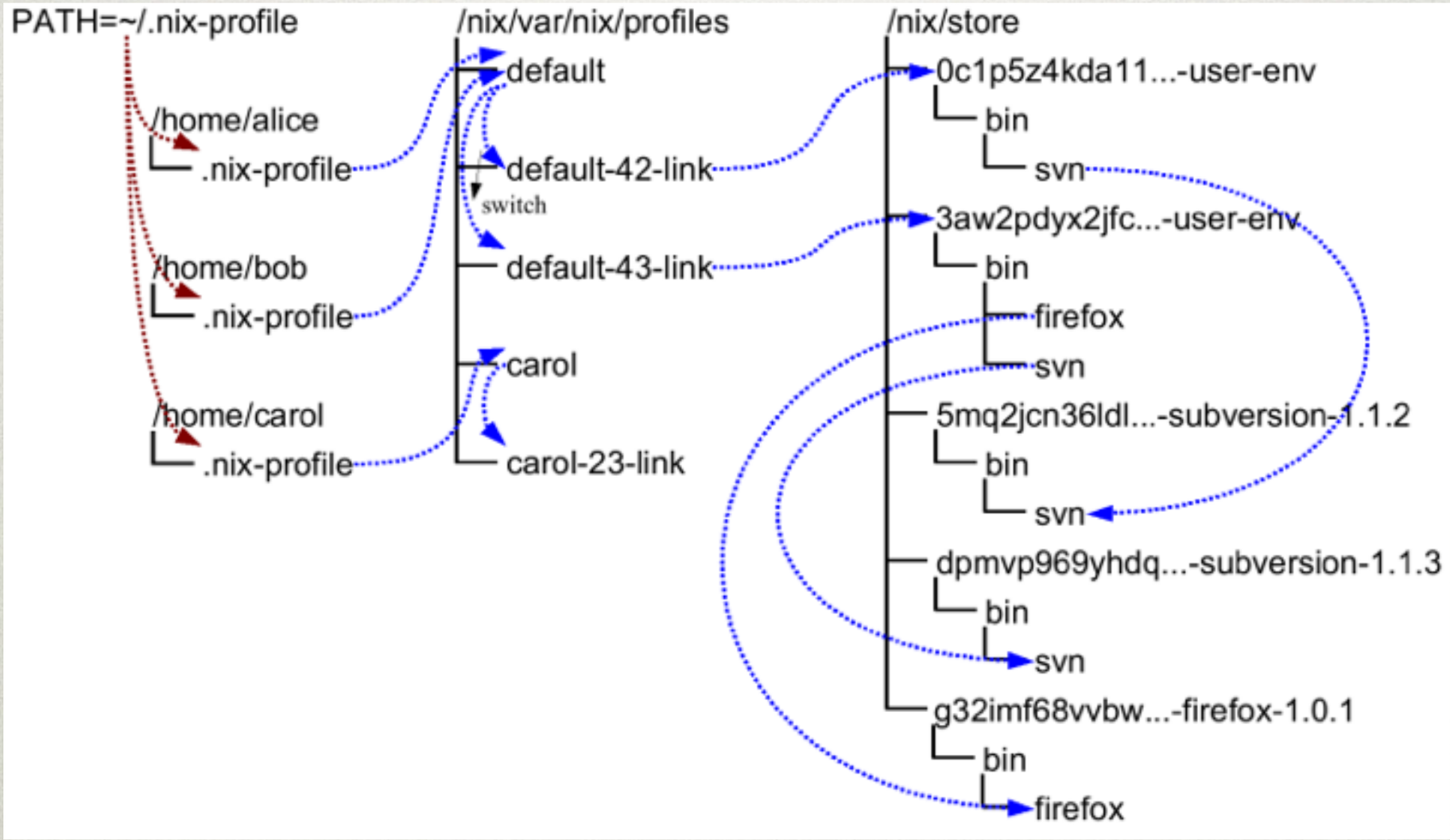
- Dependencies are explicitly specified with hashes.
  - Explicit filepath, use RPATH for DLL
- Share packages with the same hash

Well-defined closure



- Subset of nix store that is closed under the dependency graph
  - Everything needed is self-contained.
- This can be even distributed to other machines with Nix.

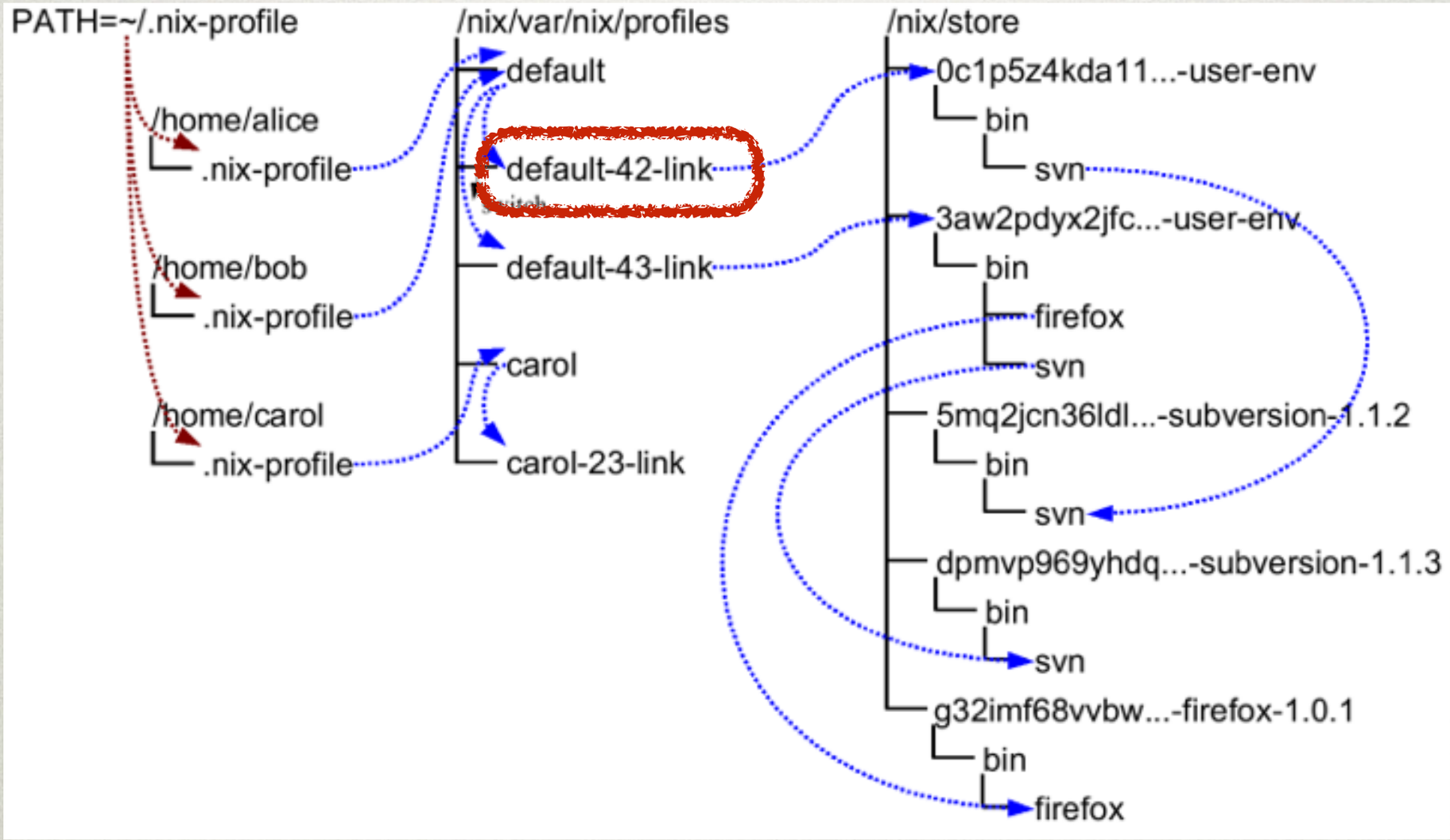
User Profile is symlink Farm



- Users / Profiles / Generations
- No `/usr`, `/usr/local`, `/opt/local`!

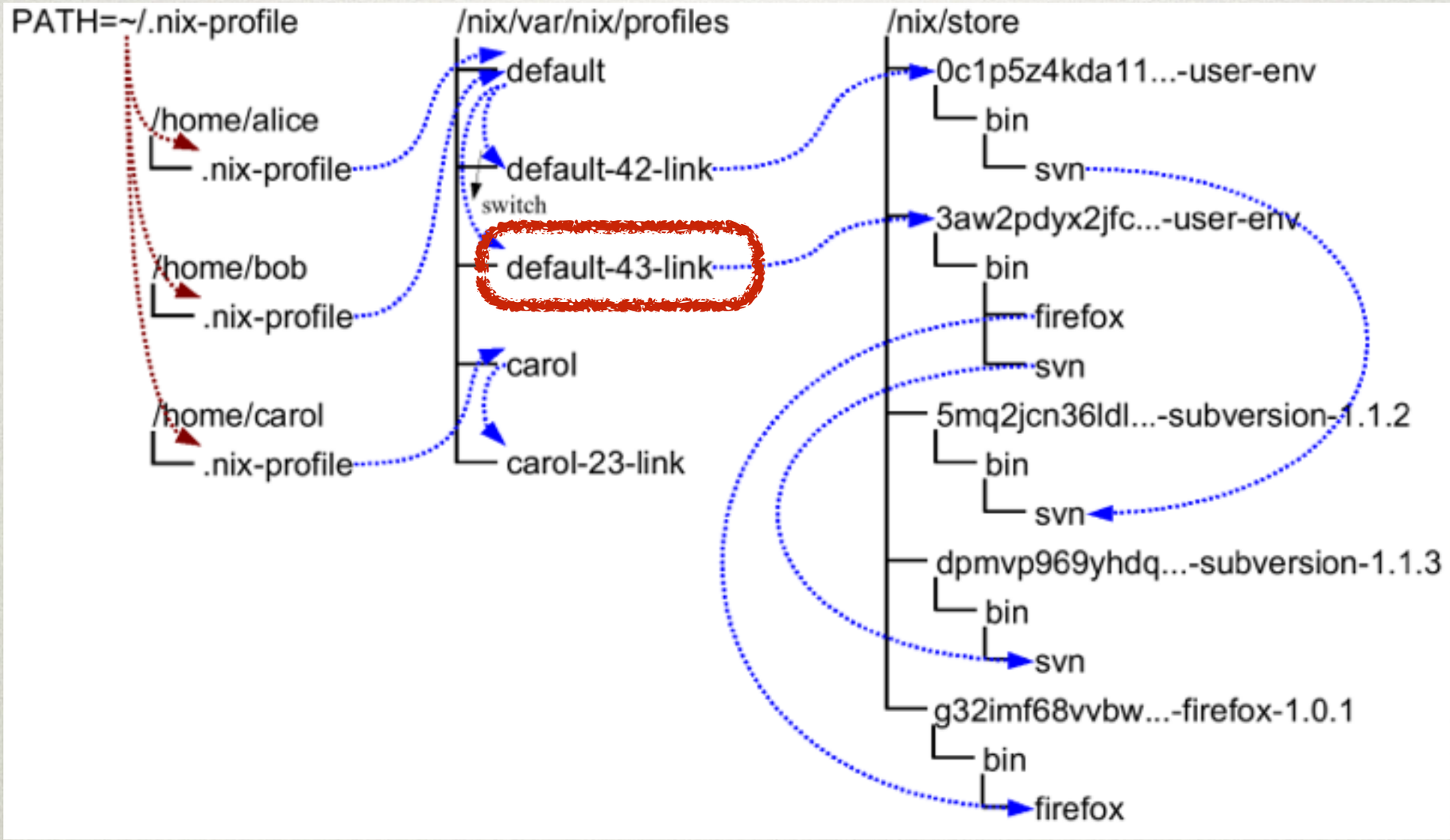
Escape from FHS for good!

User Profile is symlink Farm



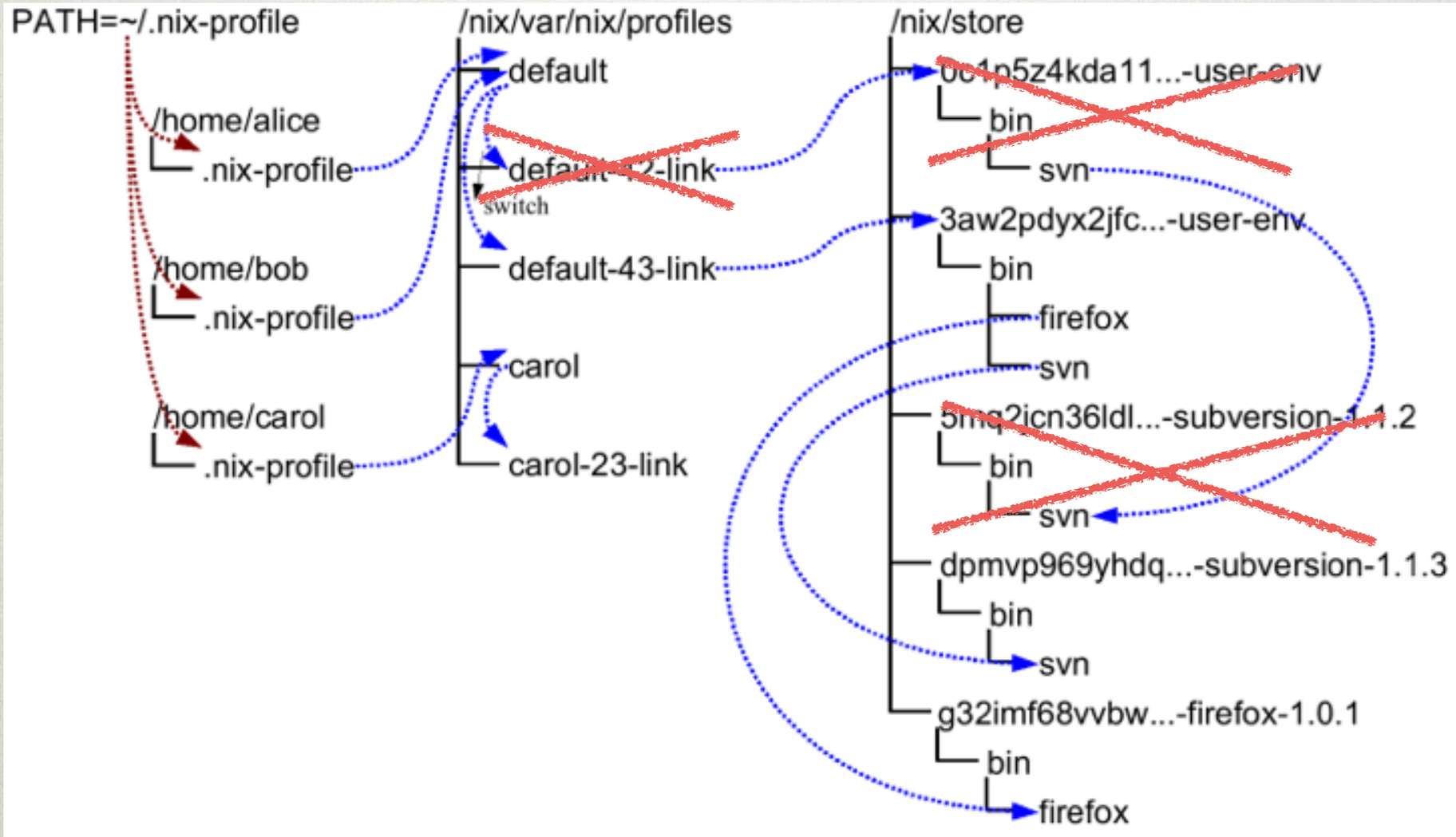
- There are no upgrade. Old versions coexist!
- Switch/rollback generation, profile

User Profile is symlink Farm



- There are no upgrade. Old versions coexist!
- Switch/rollback generation, profile

User Profile is symlink Farm



- Garbage collection!

Nix Expression Language

- Language for describing a package in purely functional package manager
- Describes package source, dependencies and build/install steps.

```
{stdenv, fetchurl, perl}: 2

stdenv.mkDerivation { 3
  name = "hello-2.1.1"; 4
  builder = ./builder.sh; 5
  src = fetchurl { 6
    url = http://ftp.gnu.org/pub/gnu/hello/hello-2.1.1.tar.gz;
    md5 = "70c9ccf9fac07f762c24f2df2290784d";
  };
  inherit perl; 7
}
```

Note that this is a function!

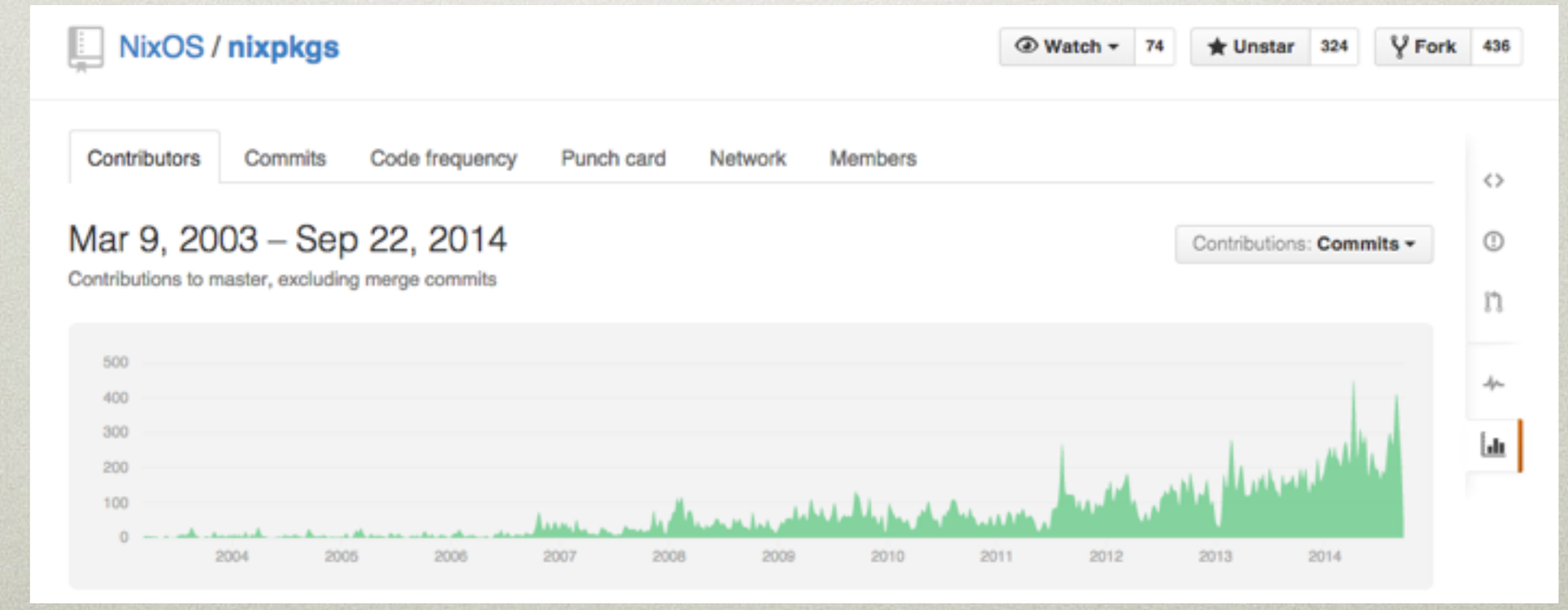
Derivation

- Intermediate rep to build a Nix package
  - Fixed schema in terms of Nix expression primitives
  - A Nix expression for a package generates a derivation
  - Every argument is applied and replaced by actual hash.
  - Hash for a package is calculated from dep derivations.
  - Packages in Nix store are finally derived from derivations

```
Derive([(“out","/nix/store/v5af71yn6jrzig3m53r4hqbbpmqjqqz7-hello-2.9","","")]
      ,[(“/nix/store/2k85dpw0b2pmyqbdrdlmb8f510l85b4b-bash-4.2-p47.drv",["out"])
       ,(“/nix/store/7h9jks2glccchjwcxv6hdx9lgvyyw486-stdenv.drv",["out"])
       ,(“/nix/store/daznf9d4hcb4ynz94xfd6r8gkigysrpy-hello-2.9.tar.gz.drv",["out"])]
      ,[“/nix/store/9krlzvny65gdc8s7kpb6lkx8cd02c25b-default-builder.sh"]
      ,”x86_64-linux","/nix/store/q5wfq0i6w4p6a7155p4hia6p3n8rk7aq-bash-4.2-p47/bin/bash"
      ,[“-e","/nix/store/9krlzvny65gdc8s7kpb6lkx8cd02c25b-default-builder.sh"]
      ,[(“buildInputs","")
      ,(“builder","/nix/store/q5wfq0i6w4p6a7155p4hia6p3n8rk7aq-bash-4.2-p47/bin/bash")
      ,(“doCheck","1"),("name","hello-2.9"),("nativeBuildInputs","")
      ,(“out","/nix/store/v5af71yn6jrzig3m53r4hqbbpmqjqqz7-hello-2.9")
      ,(“propagatedBuildInputs",""),("propagatedNativeBuildInputs","")
      ,(“src","/nix/store/xdilnlzvvsf7r33gs4vy9jq2bmazlc0j-hello-2.9.tar.gz")
      ,(“stdenv","/nix/store/v4ii73wpbg78b5p6fn0yq1cn76ya8fp4-stdenv")
```

Nixpkgs: Collection of nix expression/packages

- ~8000 packages
- Wide range of libraries, tools, and applications
- Basis for NixOS : Linux distribution on top of Nix and Nixpkgs

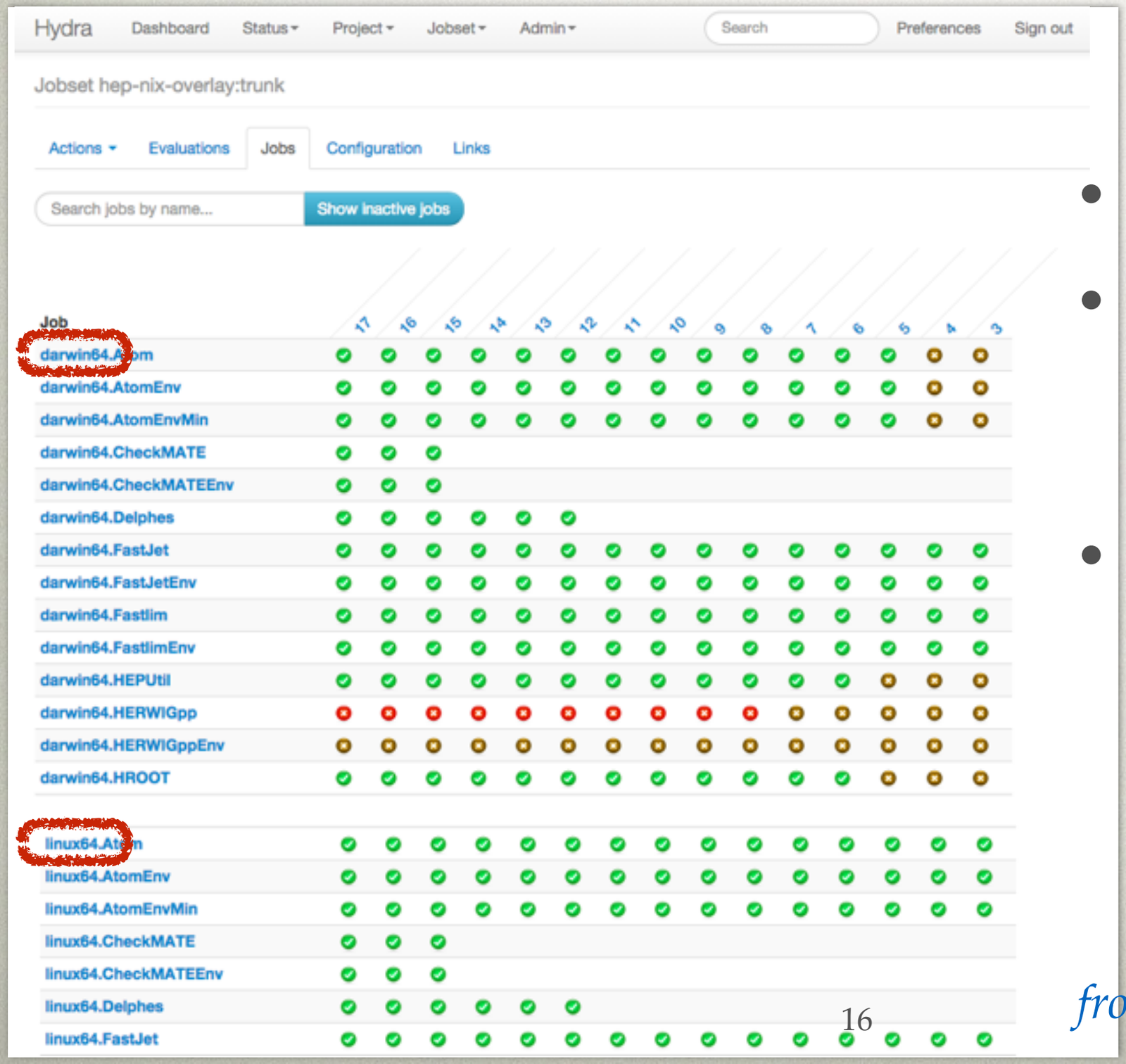


Nix Channel

- Nix allows for simultaneous source/binary deployment.
- Prebuilt nix packages with the same hash can be deployed to any systems with Nix.
- Due to the unique Nix system, completely independent Nix channels can be subscribed by users without clashes.

Very easy and safe to operate Nix channels!

Hydra: Continuous Build system on Nix

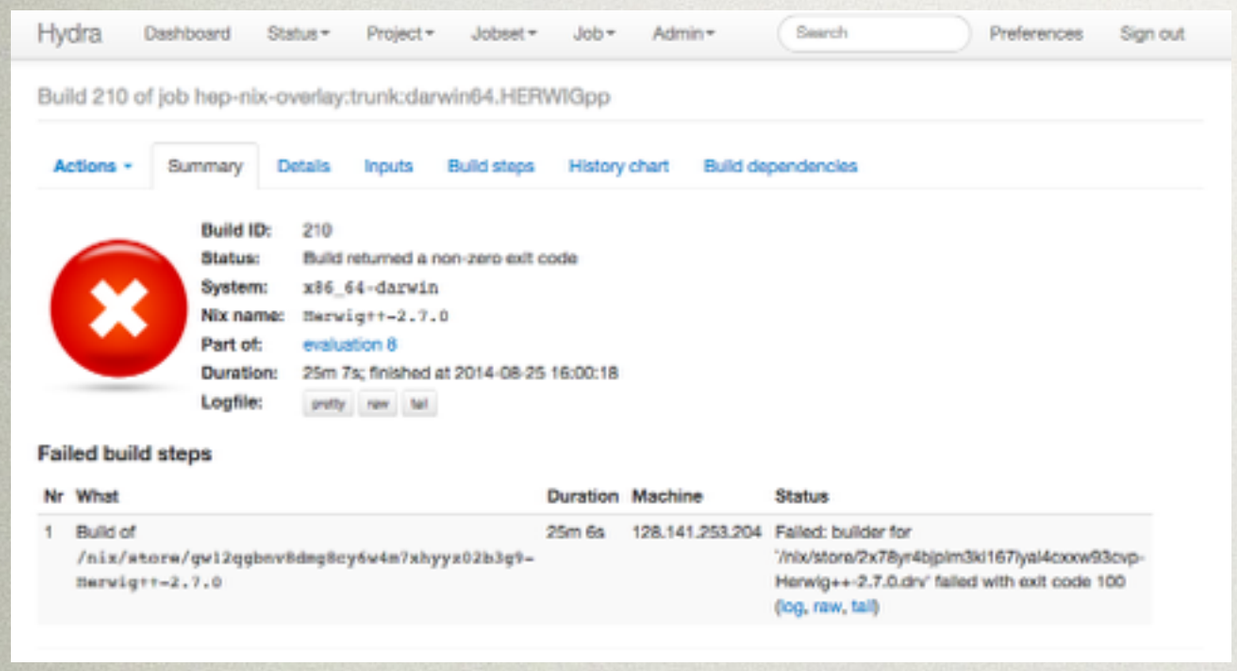


- Based on Nix
- Can employ multiple build workers with different platforms.
- Still early yet, but already has a nice web interface!

from hep-nix-overlay channel

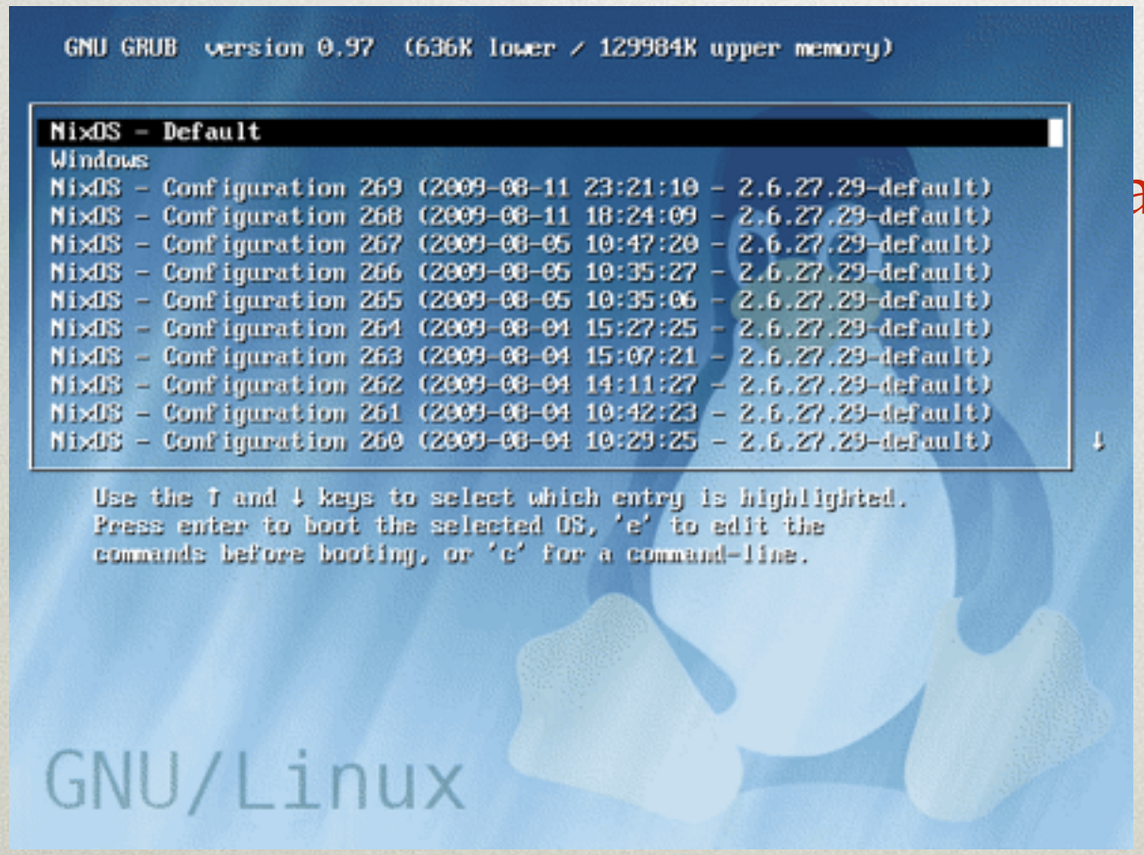
Hydra: Continuous Build system on Nix

- Success / Failure Log



Other information and beyond

- Nix/Nixpkgs/Hydra are open source software.
  - Nix: LGPL 2.1, Nixpkgs: BSD3, Hydra: GPL3
- NixOS: Linux distribution



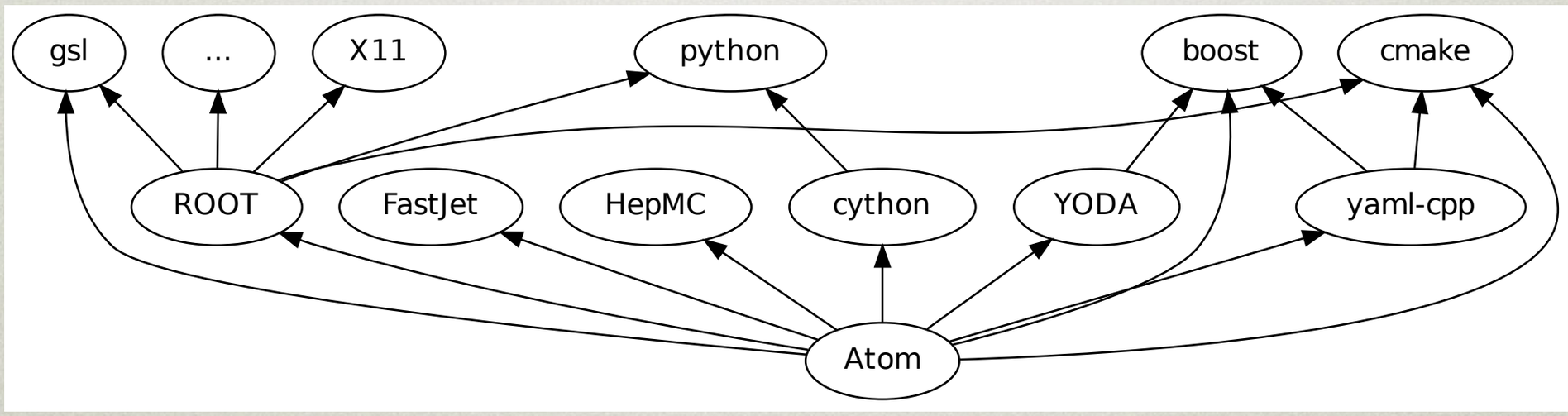
Even one can roll back an OS setup and reboot.

grub screenshot from NixOS

- NixOps: NixOS-based cloud deployment tool

My project: hep-nix-overlay

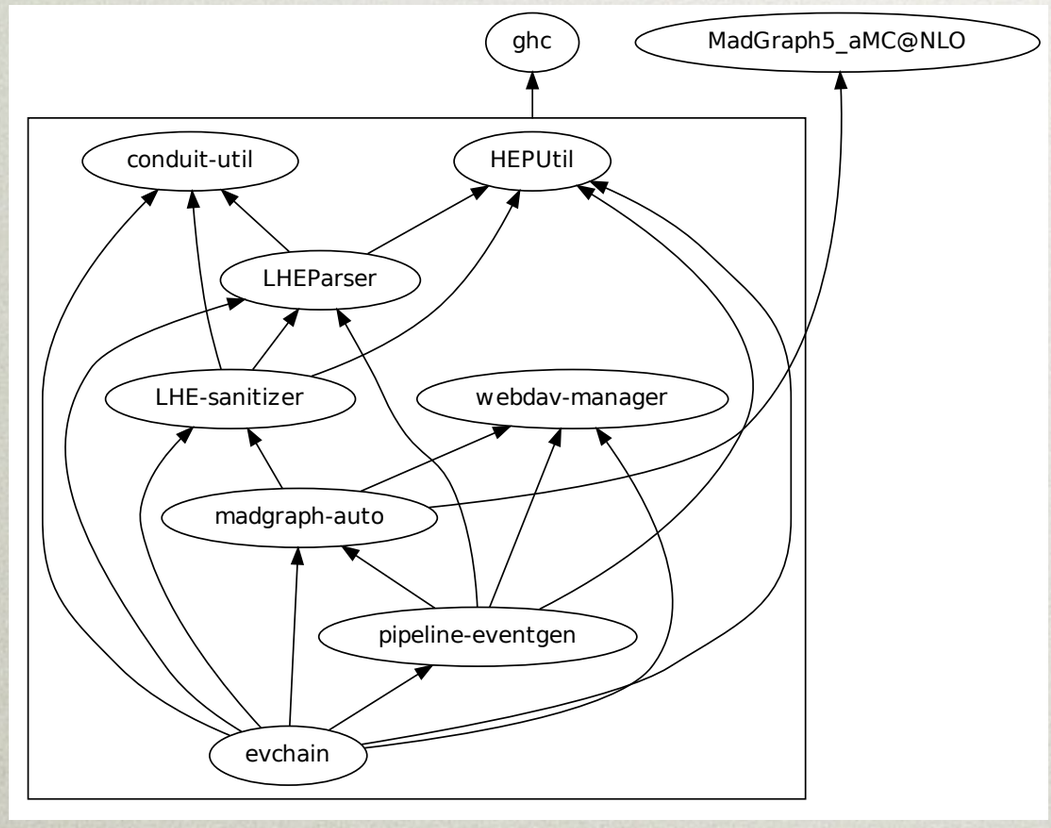
- Make a light-weight distribution channel for HEP software in TH community
- I want to deliver my software to newbies!



Dependency graphs of ATOM*: Automatic Tester Of Models*

My project: hep-nix-overlay

- Some software is even written in uncommon language, tools and build procedures.



Dependency graphs of `evchain`

ex) `evchain`: tool for chaining multi-part MC events

- written in Haskell
- Toolchain can be quite unfamiliar to average HEP people.
- Dependency is complicated. Although Haskell build tools (cabal) can deal with this well, build procedure may involve install/ reinstall dependency packages.

My project: hep-nix-overlay

- Requirements
  - Flexible and newbie friendly: source/binary installation
  - Based on a variety of development packages
  - User-space installation: considering cluster operation
  - Can allow for massive 3rd party contributions: stable with multiple version of installed packages
  - OS X support

My project: hep-nix-overlay

- hep-nix-overlay repo:

  http://github.com/wavewave/hep-nix-overlay

- hep-nix-overlay hydra status:

  http://hep-nix-overlay.cern.ch

- hep-nix-overlay channel:

  http://hep-nix-overlay.cern.ch/project/hep-nix-overlay/channel/latest

- Anyone can contribute to hep-nix-overlay repo and then the build will be tested with Hydra continuously.

hep-nix-overlay
ex) HROOT

- HROOT: Haskell binding to ROOT
- Nix installation

```
$ bash <(curl https://nixos.org/nix/install)
```

- Subscribing hep-nix-overlay

```
$ nix-channel --add http://hep-nix-overlay.cern.ch/
project/hep-nix-overlay/channel/latest hep-nix-overlay

$ nix-channel –update
```

- Installing HROOT

```
$ nix-env -i haskell-HROOT-ghc7.8.3
```

Will download and install ghc, ROOT, HROOT

Conclusion

- Nix is a purely functional package manager.
- Nix manages every package with hash in nix store to achieve purity.
- On Nix systems, multiple versions with different dependencies can coexist with each other.
- Hydra is a continuous build system based on Nix.
- hep-nix-overlay is a light-weight HEP software distribution channel.

References

- Nix/NixOS project homepage: http://nixos.org
- E. Dolstra's Ph.D. thesis:

  "The Purely Functional Software Deployment Model"

  http://nixos.org/~eelco/pubs/phd-thesis.pdf
- NixOS github organization: http://github.com/NixOS
- Nixpkgs monitor: http://monitor.nixos.org
- Nixpkgs hydra build system: http://hydra.nixos.org
- Blog aggregation: http://planet.nixos.org

Thank you!