On the reproducibility of numerical experiments with FEniCS

Paul Garlick

Tourbillion Technology Ltd

22nd March 2021

https://www.tourbillion-technology.com

Reproducibility

- Significant factor when assessing the trustworthiness of results

- Problems can be encountered when:

    – moving a simulation environment from one system to another

    – restoring the simulation environment from a previous project

- Solutions can be categorized as:

    – approximate: *docker*, *singularity*, *spack* . . .

    – exact (bit-identical): *nix*, **guix**
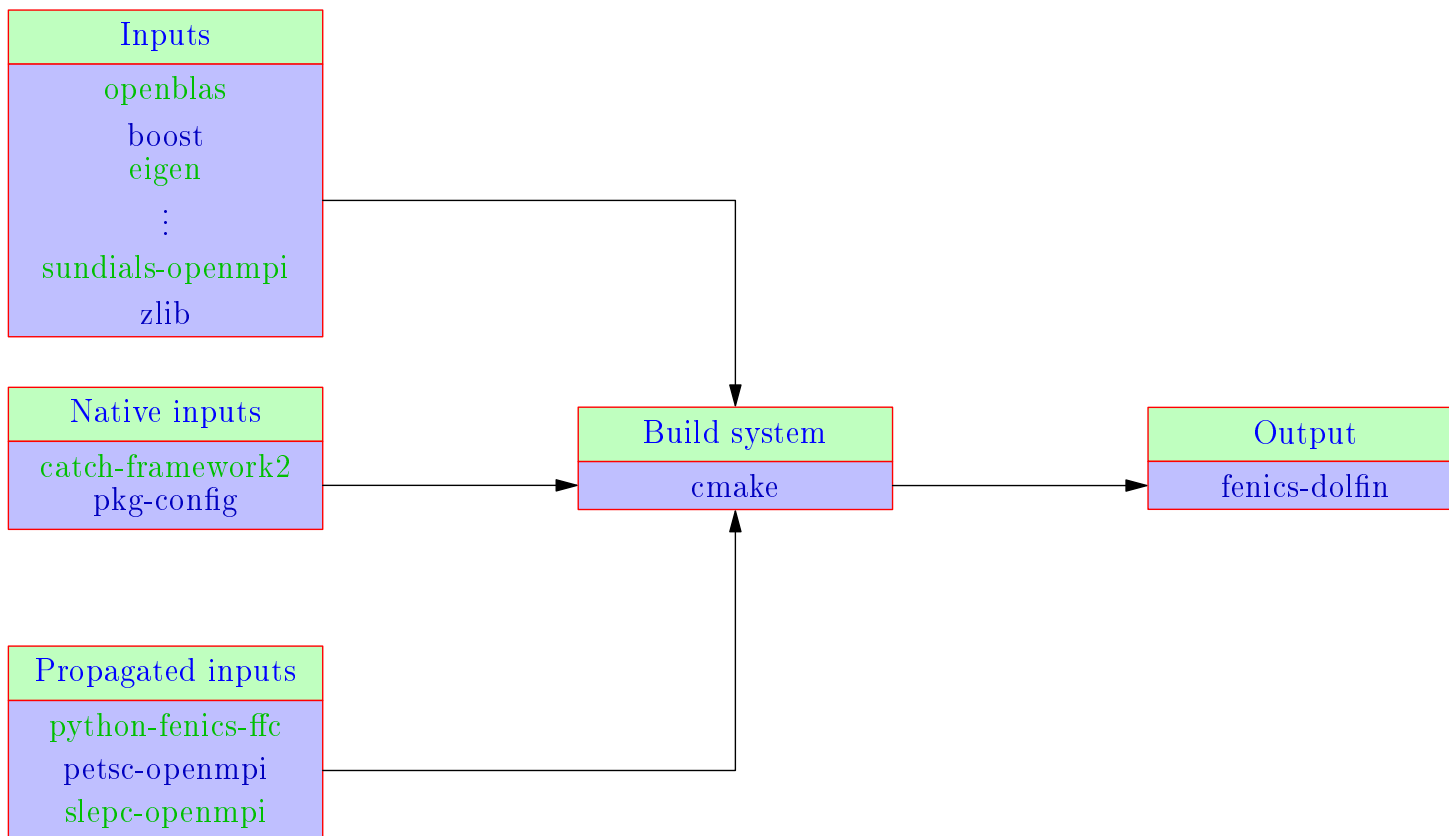
Guix
GNU Guix

Guix introduction

- GNU/Linux distribution and package manager

  – x86_64, aarch64, powerpc64le architectures

- 16,000+ packages

  – FEniCS is part of the *simulation* module

- 60+ contributors/month

- Guile Scheme API; embedded DSL for defining packages

- File structure allows a separation of concerns. The operating system and *user profiles* can be managed independently:

```
/ (root)
├── OS

└── /gnu
     └── /store
```

How it works

- Each package defines a list of inputs and a build function

- The build function returns the installed package with no side effects

Inputs
openblas
boost
eigen
⋮
sundials-openmpi
zlib

Native inputs
catch-framework2
pkg-config

Propagated inputs
python-fenics-ffc
petsc-openmpi
slepc-openmpi

Build system
cmake

Output
fenics-dolfin

Example build process.

- Build process can be expressed in mathematical form, relating inputs to output objects via the build function:

$$f_{b,h} : \{i_{0,h}, i_{1,h}, \ldots, i_{m,h}\} \mapsto \{o_0, o_1, \ldots, o_n\}$$

- Directory name for a package contains a hash of all of the inputs plus the package name and version:

```
/gnu/store/ywcdaz69y36...jbvw1igj-fenics-2019.1.0.post0
```

- The standard tools provide an automated process for managing complexity in package hierarchies

    – a *rolling release* update process

- To guarantee the reproducibility of a complete software environment two advanced features are needed:

    – a means to specify the packages to include

    – the ability to *pin* the distribution

Small list of lists

- User profile is defined by a *manifest* file:

```
(specifications->manifest
 '("emacs"
   "fenics"
   "gmsh"
   "python"
   ))
```

- Source code location is defined by a *channel specification* file:

```
(list (channel
        (name 'guix)
        (url
          "https://git.savannah.gnu.org/git/guix.git")
        (commit
          "a002e8a4f58a45034075cad27bf8eb65679bcc14")))
```

- Use version control to record changes to the lists:

  – move forward and back through history as needed

Status

- Package definitions completed:

  – FEniCS 2018.1.0.post1

  – FEniCS 2019.1.0.post0

- In progress:

  – pyadjoint

  – pygmsh, meshio

- Todo:

  – FEniCSX

  – Firedrake

Links

- Getting started:

  – `https://guix.gnu.org`

- Moving on:

  – `https://guix.gnu.org/manual`

  – `https://guix.gnu.org/cookbook`

- Joining in:

  – Guix simulation channel (site under construction)

- User support:

  – email: `pgarlick@tourbillion-technology.com`