Package Versions Matter

The switchr framework

Gabriel Becker, Michael Lawrence

June 30, 2015

Portable scripts

```
gisturi <- "https://gist.github.com/gmbecker/...")
switchTo("project", seed = gisturi)
## Analysis code here
```

- Script will run identically[1] everywhere
  - Use same versions of packages

[1] Assuming same version of R and data availability

Four pillars of Data Analysis

- Data
- Code
- Statistical Methods
- Software Used

Our Focus

- Data
- Code
- Statistical Methods
- Software Used
  - *including specific versions*

Definitions

- Package cohort - *A set of packages which are to be operated on as a single unit*
  - E.g., for testing, installation, loading, or publication.
- Versioned package cohort - *A package cohort in which some or all packages are associated with an exact release version*

Package Cohorts are crucial

- Reproducibility
  - Restore an environment in order to reproduce a result
- Collaborations
  - Working with the same versioned package cohort helps ensure comparability of results
- Package development
  - Differentiating and switching between development and production cohorts
- Large organizations/depts
  - Specify/provide canonical, versioned package cohorts for use by all members

Users need tools

To allow effective management of pkgs at the cohort level

- Package libraries
  - Create, populate, and switch between
- Generalized installation
  - Version specific
    - Past releases and devel versions
  - CRAN-style repositories and other sources (version control)
- Describing cohorts
  - Define versioned or non-versioned cohorts
  - Publish cohorts as manifests or repositories

Formal representation of a package cohort

- Package manifests define package cohort and contain info about each package
  - Name of the package
  - Location of the source code
  - Type of location
    - git, svn, CRAN, bioc, etc
- Seeding manifests define a versioned cohort on top of a package manifest
  - Specific versions for a subset of the packages
- Manifests act as a de-centralized, virtual CRAN-style repository
  - Can install packages "directly" using manifests

A package manifest

```
library(switchr)
ghman <- GithubManifest("gmbecker/fastdigest",
    "duncantl/CodeDepends")
ghman
```

```
A package manifest (PkgManifest object)

Contains 2 packages and 5 dependency repositories

Packages:
          name type
1  fastdigest  git
2 CodeDepends  git
```

A seeding manifest

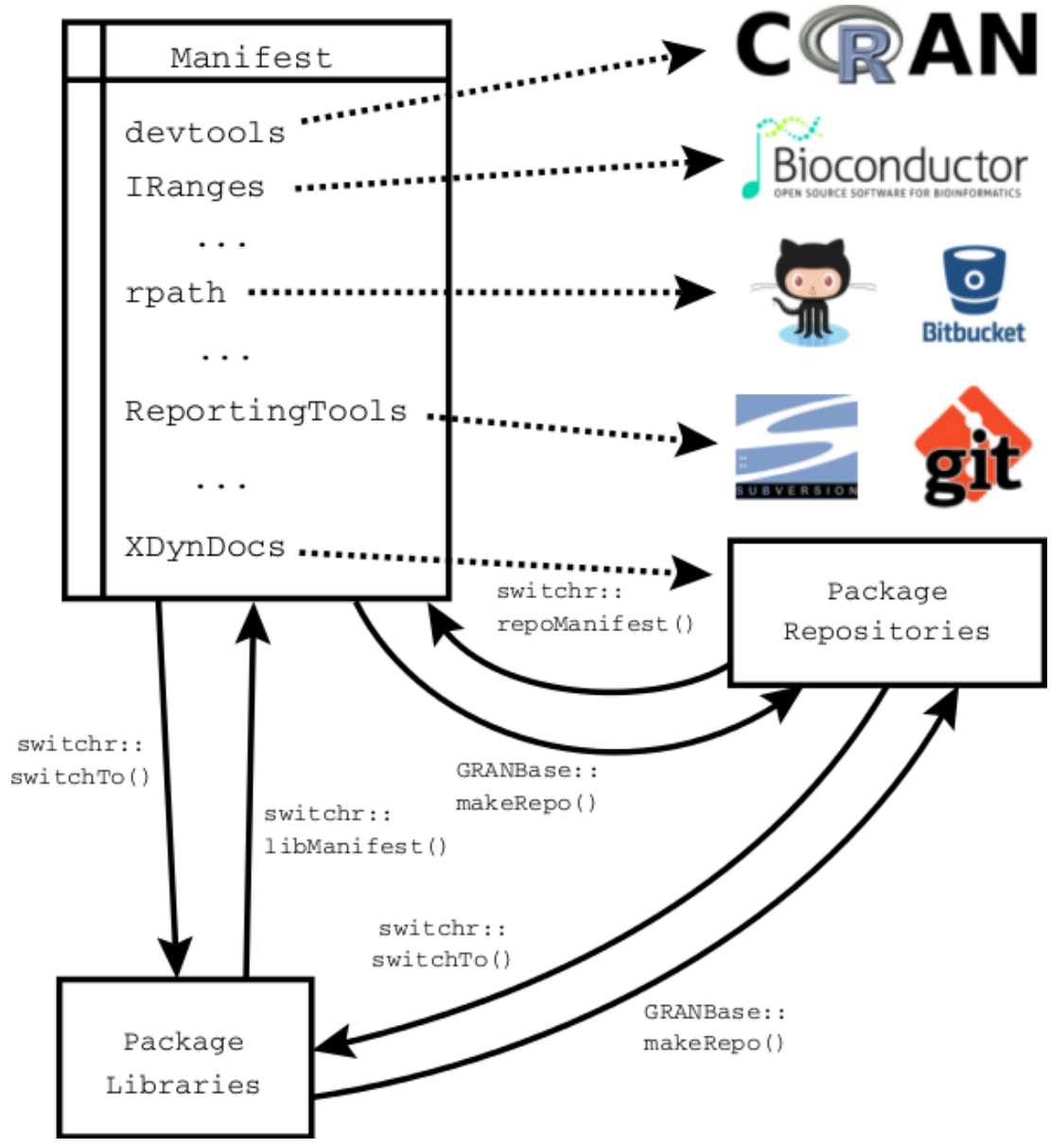
```
libman <- libManifest()
head(libman)
```

```
A seeding manifest (SessionManifest object)

Describes a cohort of 5 package versions.
145 packages are listed in the underlying package manifest

Package versions:
            name version
1 AnnotationDbi  1.30.1
2    assertthat     0.1
3        base64     1.1
4     base64enc   0.1-2
5     BatchJobs     1.7
```

A unified framework



Switching package libraries

```
switchTo("example")
```

```
Switched to the 'example' computing environment.
29 packages are currently available.
Packages installed in your site library ARE suppressed.
To switch back to your previous environment
type switchBack()
```

```
switchBack()
```

```
Reverted to the 'original' computing environment.
159 packages are currently available.
Packages installed in your site library ARE NOT suppressed.
To switch back to your previous environment
type switchBack()
```

Seeding libraries with manifests

```
switchTo("example2", seed = ghman)
```

- New library
  - Packages listed in seed are installed automatically
    - Exact versions if specified
- Existing library
  - Library is loaded without modification

You can safely have a `switchTo` call with a seed in your script

As gists

- switchrGist publishes manifests as Gists

```
library(switchrGist)
publishManifest(ghman, Gist())
```

As package repositories

- GRANBase creates CRAN/Bioc-like repositories from manifests
  - Permanent
  - Formally tested (as a cohort)

```
library(GRANBase)
makeRepo(ghman)
```

Previous CRAN state via metacran(db)

- Packages on CRAN for a particular R release

```
man <- rVersionManifest("2.14.1")
head(man)
```

```
A seeding manifest (SessionManifest object)

Describes a cohort of 5 package versions.
3410 packages are listed in the underlying package manifest

Package versions:
   name version
1  aaMI   1.0-1
2   abc     1.4
3   abd  0.1-22
4 abind   1.4-0
5   abn   0.5-1
```

Historically appropriate dependencies

- Manifest of dependencies given single package version

```
dtman <- cranPkgVersManifest("devtools", "1.4.1",
                             suggests="none")
head(dtman)
```

```
A seeding manifest (SessionManifest object)

Describes a cohort of 5 package versions.
9 packages are listed in the underlying package manifest

Package versions:
      name  version
1 devtools    1.4.1
2     httr      0.2
3    RCurl 1.95-4.1
4  memoise      0.1
5  whisker    0.3-2
```

Frozen repositories from previous CRAN states

- We can convert, e.g., the devtools manifest into a repository

```
repo <- makeRepo(dtman, basedir="~/devtools1.4.1repo")
```

Installing from SVN checkouts of related Bioc pkgs

- Bioc packages are highly interdependent
  - Working off SVN for one means working off SVN for all
- switchr supports *lazy repositories*
  - Details are out of scope here
  - Will use existing checkouts or create new ones as necessary
  - Local changes will be reflected in repo

```
bman <- BiocSVNManifest("devel")
lrepo <- lazyRepo("rtracklayer",
                  pkg_manifest = bman,
                  dir = "~/mylocalcheckout")
install_packages("rtracklayer", lrepo)
```