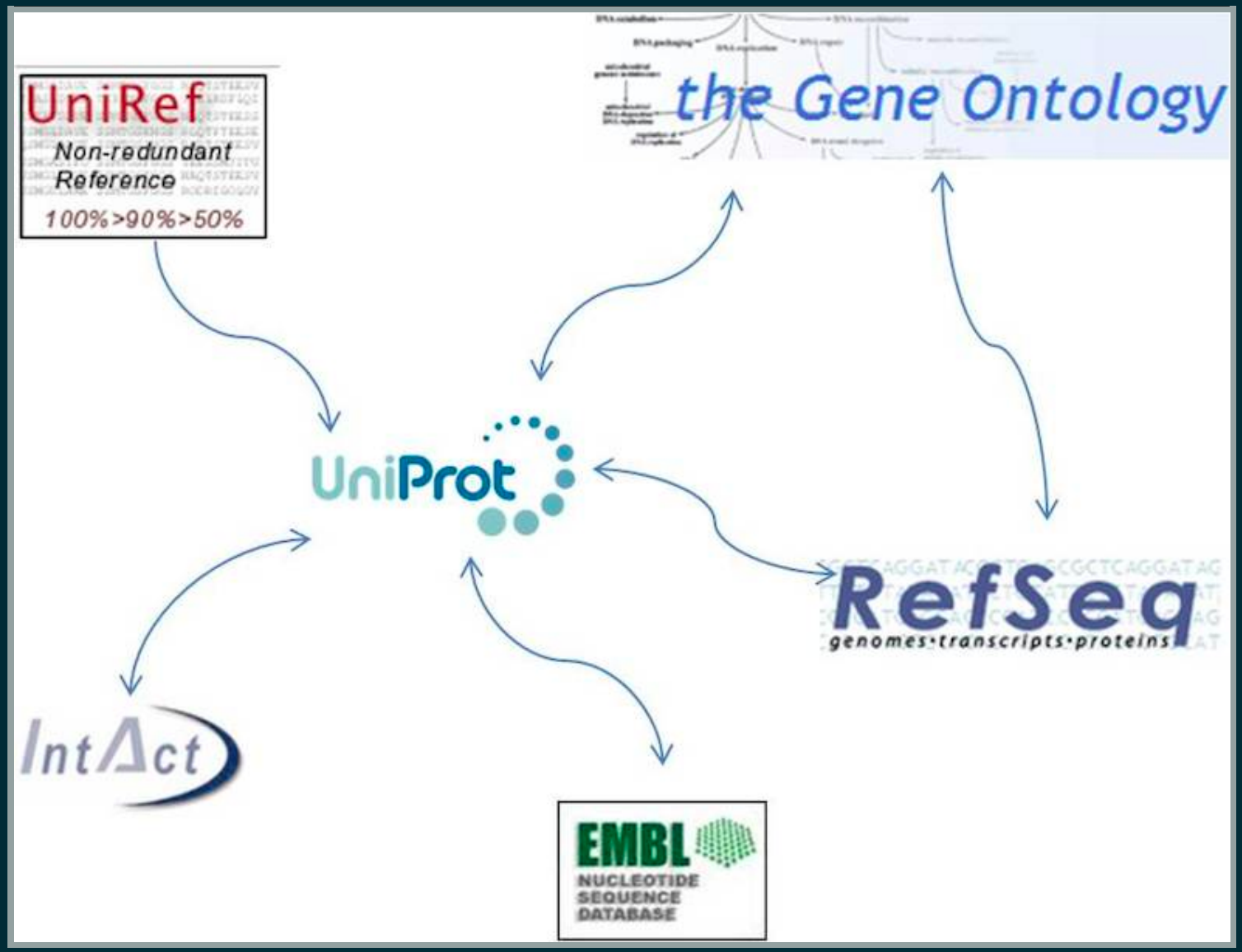
BIO4J + STATIKA

ALEXEY ALEKHIN

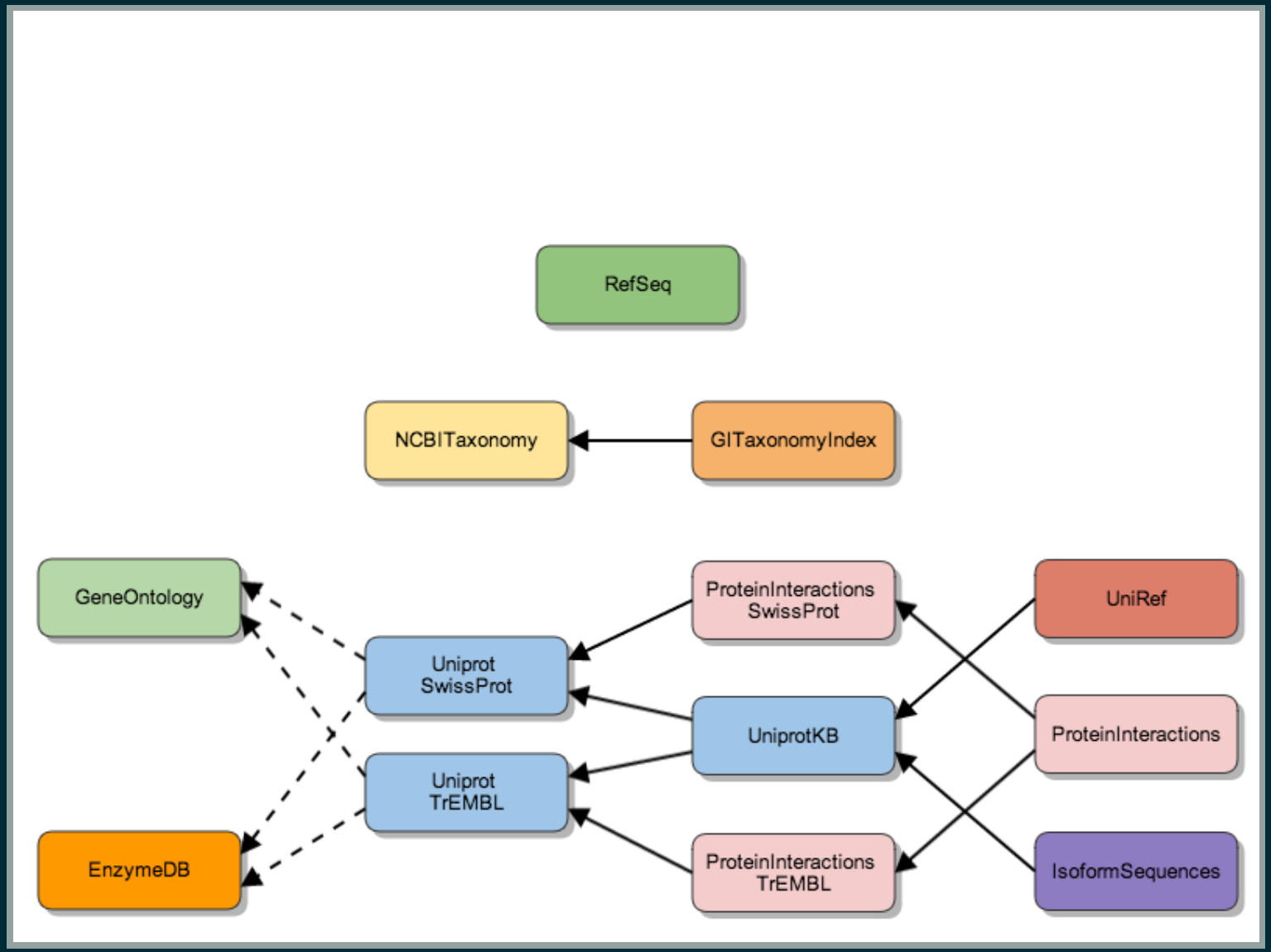
GRAPH DEVROOM @ FOSDEM 2014

BIO4J MODULES

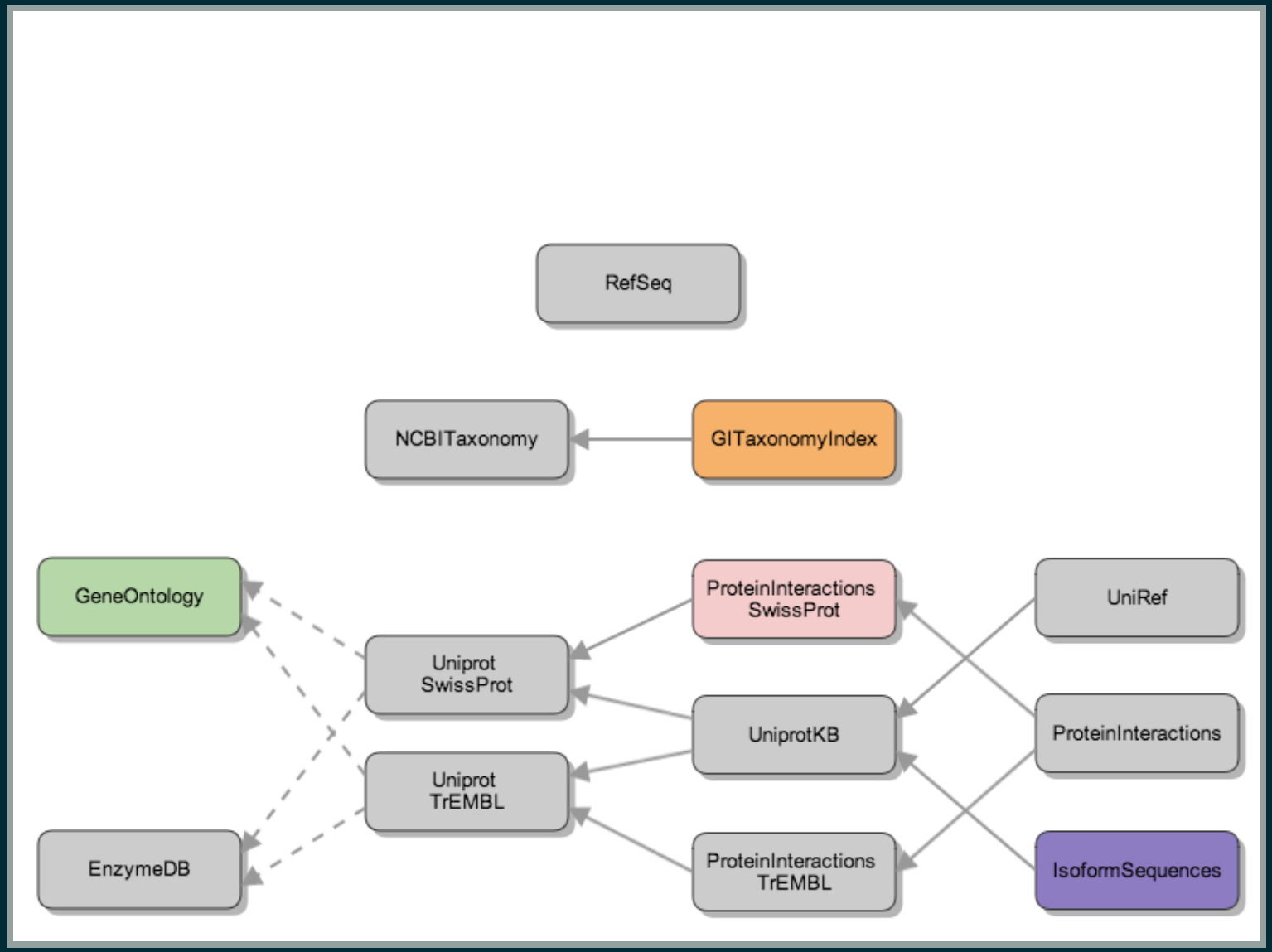
BIO4J DATA SOURCES



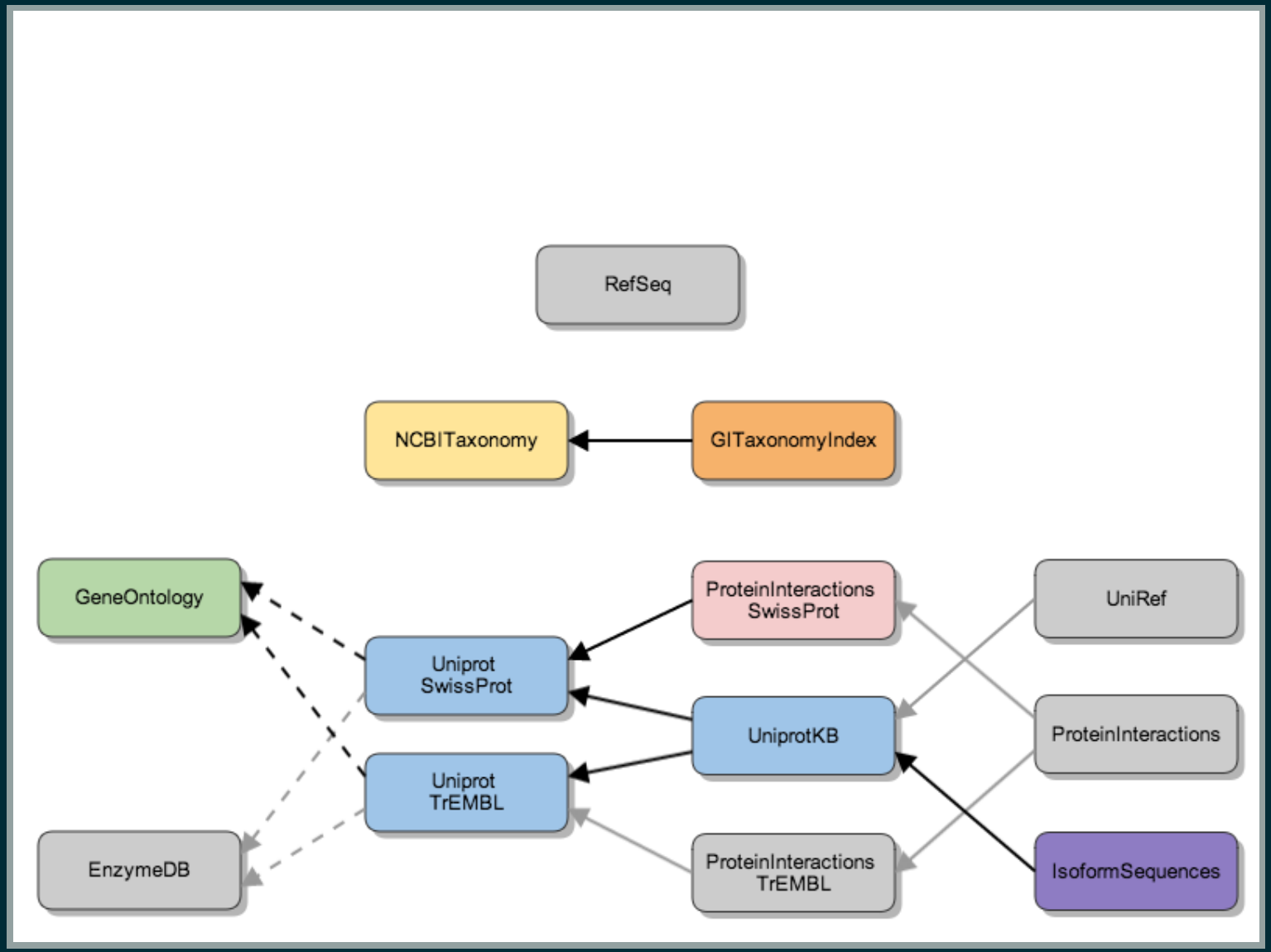
BIO4J MODULES HIERARCHY



BIO4J CUSTOM RELEASES



BIO4J CUSTOM RELEASES



GOALS

- Flexible module system
- Simple import process
- Dependencies management
- Easy and robust deployment

STATIKA

WHAT IS STATIKA

ABSTRACT MODULE SYSTEM

- Modules as Scala types — *bundles*
- They can *depend* on each other!
- It's validated by compiler —
  i.e. *statically*
- Linearizing types graph to get them in the right order

MANAGING ARTIFACTS

- Packing bundles into versioned artifacts (jars)
- Reusing SBT (Simple Build Tool) infrastructure
- Standardizing settings and release process with the sbt-statika plugin

DEPLOYMENT

Amazon Web Services + aws-statika lib

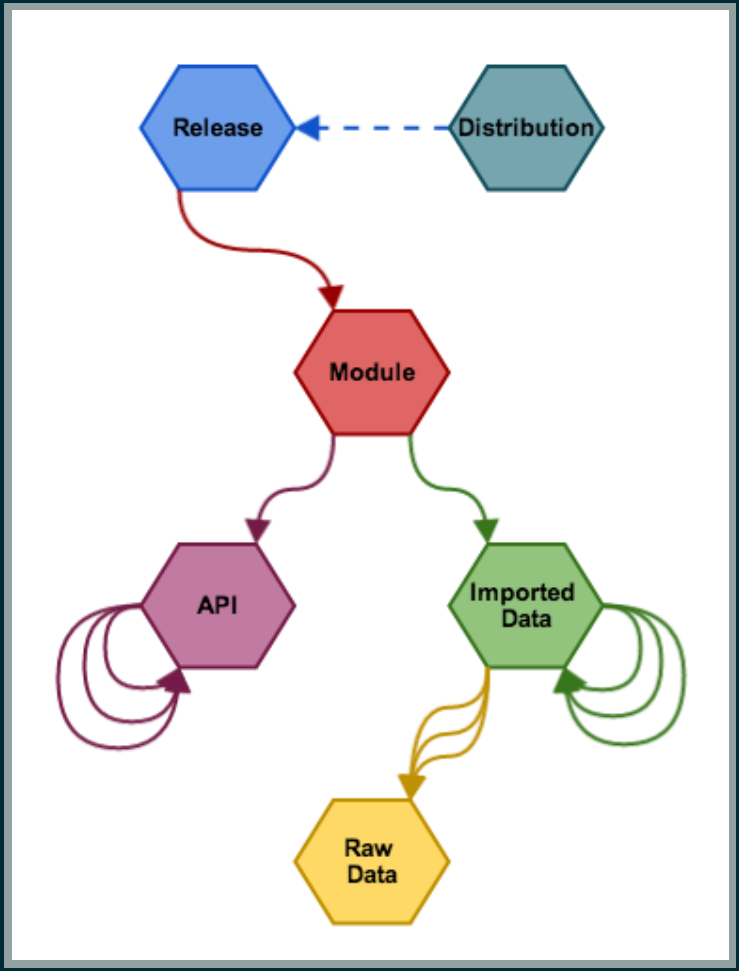
- Bundles can be *applied*, i.e. deployed it to an EC2 instance
- Statika *distributions* — an abstraction for the cloud infrastructure specifics

BIO4J + STATIKA

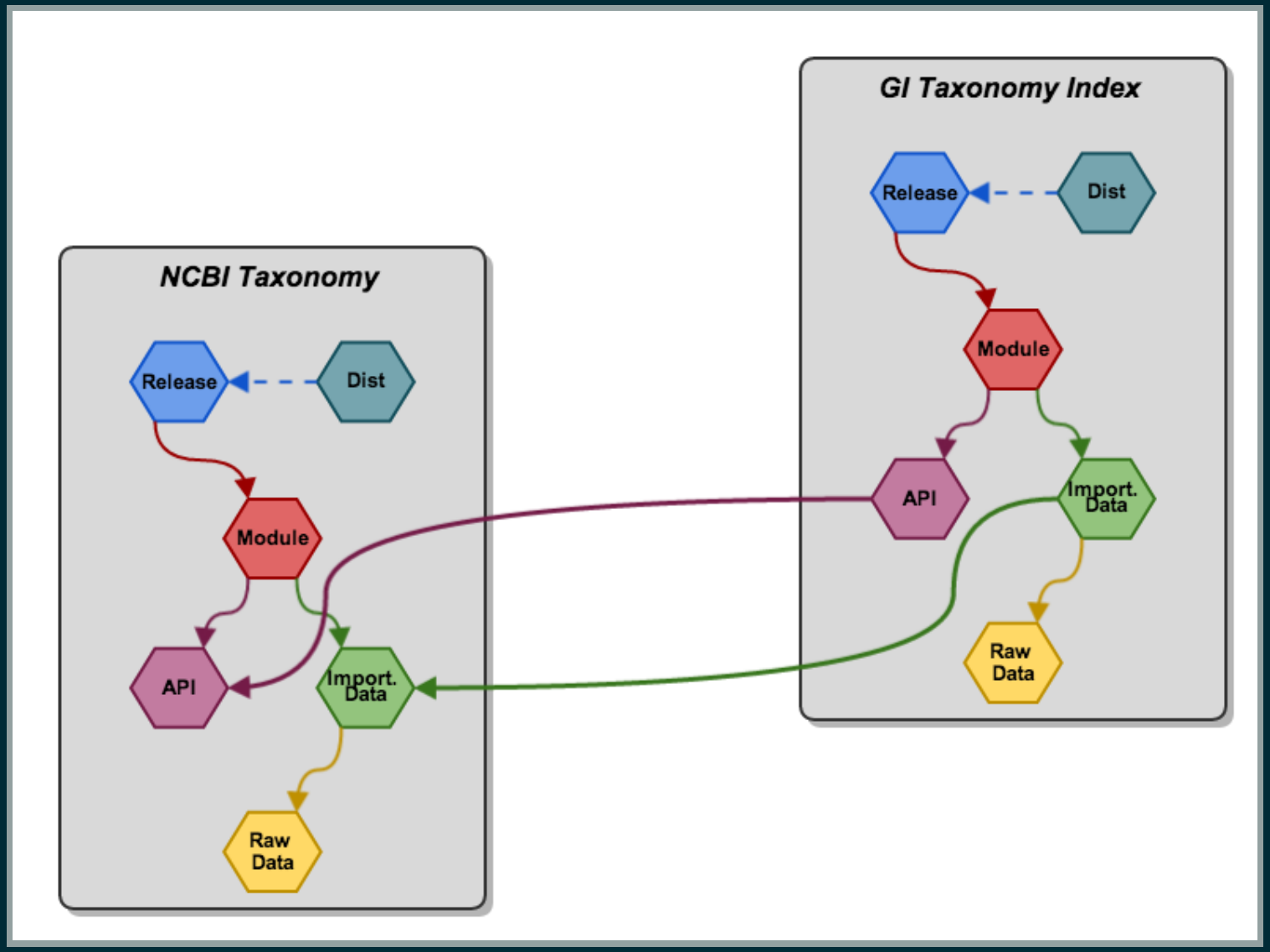
DEFINING BIO4J MODULES

- Raw data
- Node/relations type defs
- Importing process
- Exposing some API

INNER BUNDLES LAYOUT



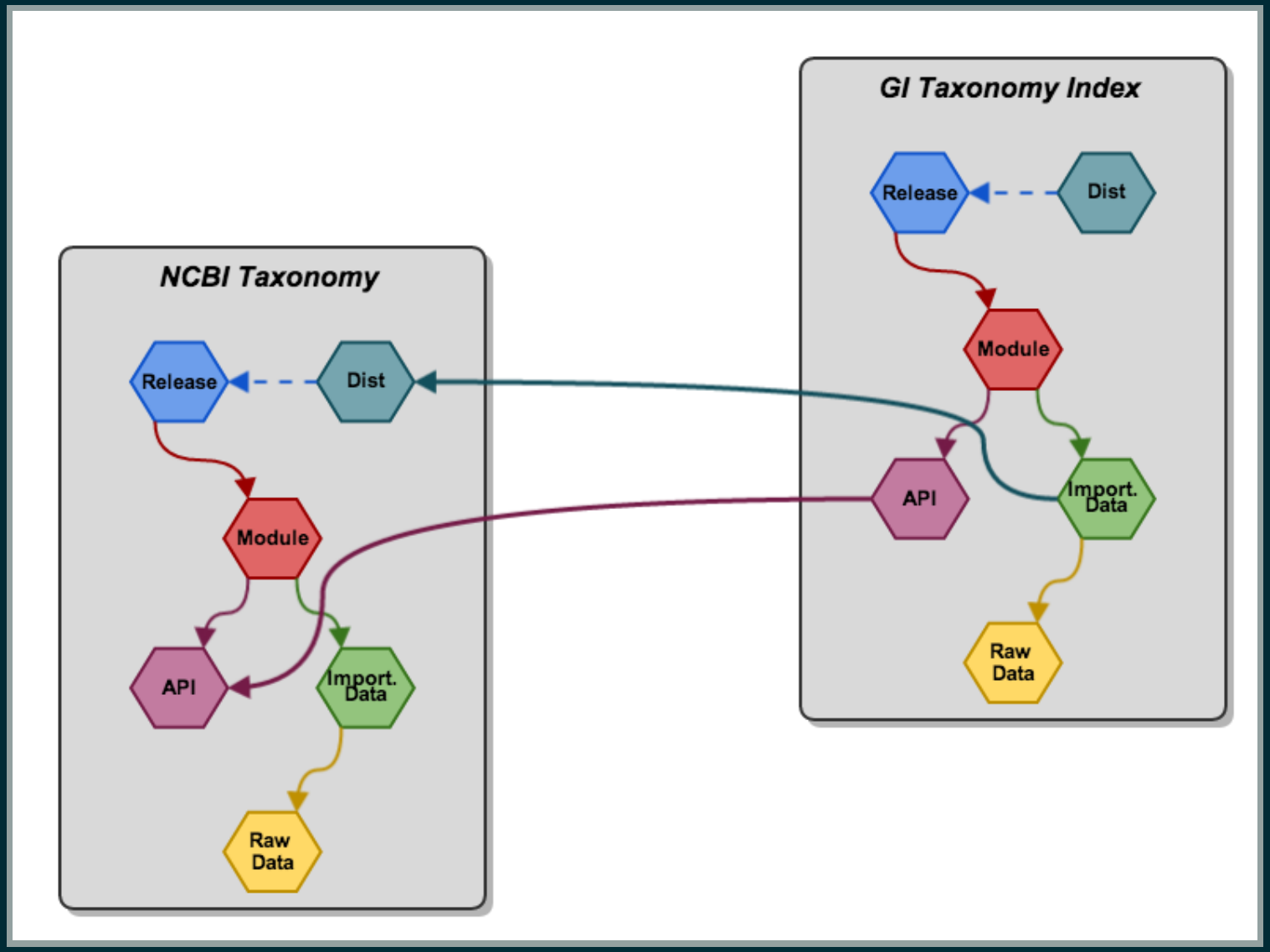
INNER BUNDLES LAYOUT EXAMPLE



INCREMENTAL IMPORT

- Incremental import of data
  to existing Bio4j distributions
- Not repeating already done work
- Easy to describe abstractly

INCREMENTAL IMPORT EXAMPLE



CUSTOM RELEASE OF BIO4J

- create a release-bundle with needed modules
- be sure not to spend resources on a wrong configuration — compile it!
- use tools for easy release and deployment: sbt-statika + statika-cli

SUMMARY

BIO4J + STATIKA = WIN!

- *Abstract* layout of bundles
- Hierarchy of *concrete* modules
- Tracking deps on *all levels*
- Doing it at *compile time*
- Using AWS *cloud* infrastructure for the actual work