Invenio Technology

Selected Practical Software Development Lessons From A Large Digital Library System

Tibor Šimko
<tibor.simko@cern.ch>

Department of Information Technology
CERN

August 2013 / openlab talk

Outline

1 Introduction
- Digital Library
- Invenio

2 Case Studies
- Episode 1: Python
- Episode 2: Git
- Episode 3: Testing
- Episode 4: Building Efficient Indexes
- Episode 5: NIH
- Episode 6: Scalability

3 Conclusions

Outline

1 Introduction
- Digital Library
- Invenio

2 Case Studies
- Episode 1: Python
- Episode 2: Git
- Episode 3: Testing
- Episode 4: Building Efficient Indexes
- Episode 5: NIH
- Episode 6: Scalability

3 Conclusions

Outline

1 Introduction
- Digital Library
- Invenio

2 Case Studies
- Episode 1: Python
- Episode 2: Git
- Episode 3: Testing
- Episode 4: Building Efficient Indexes
- Episode 5: NIH
- Episode 6: Scalability

3 Conclusions

What is Digital Library?

- *"library in which collections are stored in digital formats (as opposed to print, microform, or other media) and accessible by computers"*
- (1) institutional document repositories
- (2) world-wide subject-based information systems

Example #1: CERN Document Server

- managing CERN and selected non-CERN high-energy physics and related documents since ∼1993
- more than 1,000,000 records
- articles, books, theses, photos, videos, and more
- powered by Invenio since 2002
- `http://cdsweb.cern.ch/`

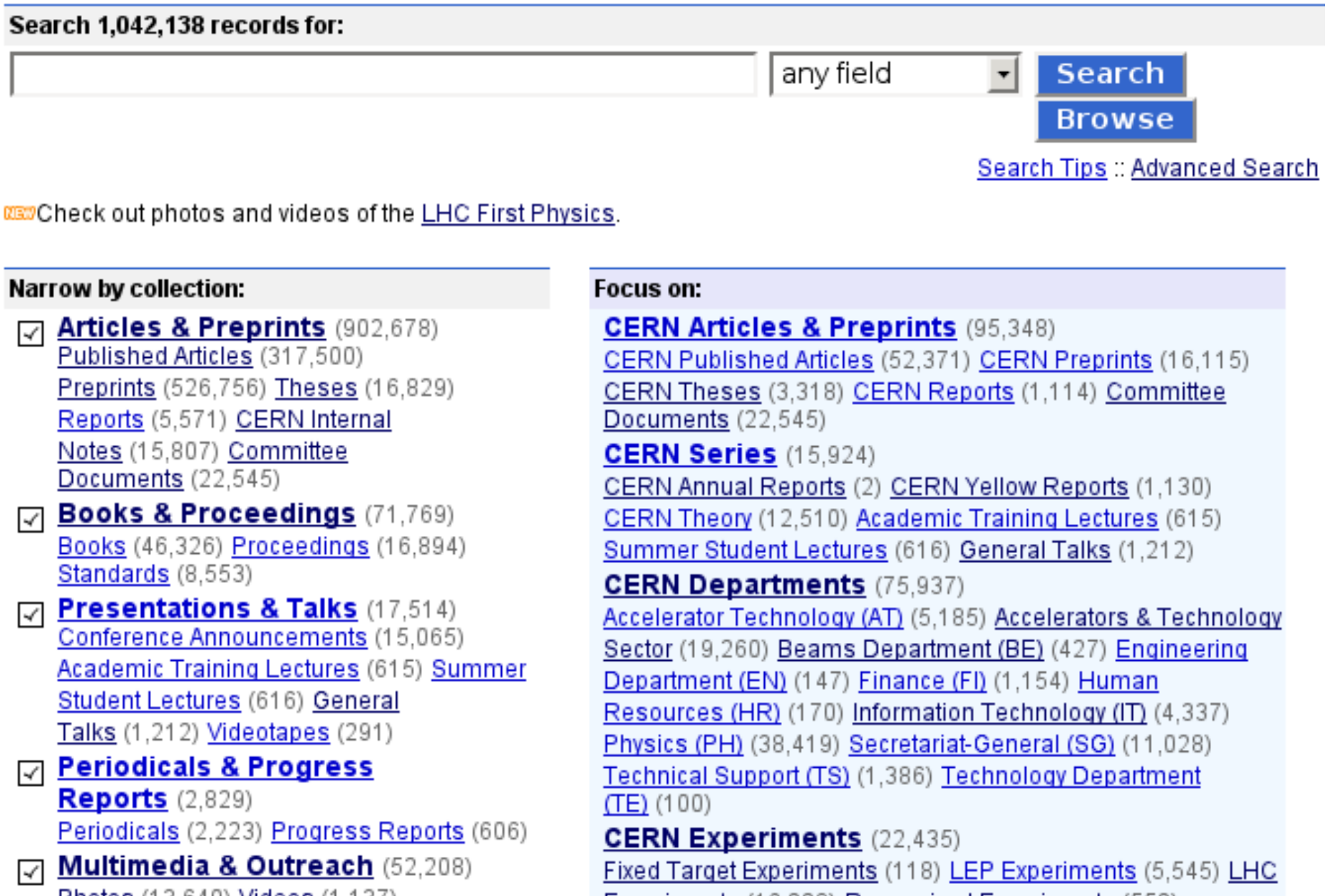
What is Digital Library?

- *"library in which collections are stored in digital formats (as opposed to print, microform, or other media) and accessible by computers"*
- (1) institutional document repositories
- (2) world-wide subject-based information systems

Example #1: CERN Document Server

- managing CERN and selected non-CERN high-energy physics and related documents since $\sim$1993
- more than 1,000,000 records
- articles, books, theses, photos, videos, and more
- powered by Invenio since 2002
- `http://cdsweb.cern.ch/`

CDS: Collection Tree



CDS: Search for Books

Search:

python cookbook | any field | Search | Browse

Search Tips :: Advanced Search

Search collections:

Books | *** add another collection ***

Sort: latest first | desc. | - or rank by -

Display results: 10 results | split by collection

Output format: HTML brief

Books **2** records found Search took 0.10 seconds.

☐ 1. 

Python Cookbook . - 2nd ed. / Martelli, Alex
Beijing : O'Reilly, 2005. - 807 p.
Purchase from CERN Bookshop - CERN library copies

This book at Amazon

Detailed record - Similar records

☐ 2. 

Python Cookbook / Martelli, Alex (ed.) ; Ascher, David (ed.)
Beijing : O'Reilly, 2002. - 574 p.
CERN library copies

CDS: Search for Photos

Photos

Search:

hc tunnel | any field | Search | Browse

Search Tips :: Advanced Search

Search collections:

Photos | *** add another collection ***

Sort by: latest first | desc. | - or rank by -

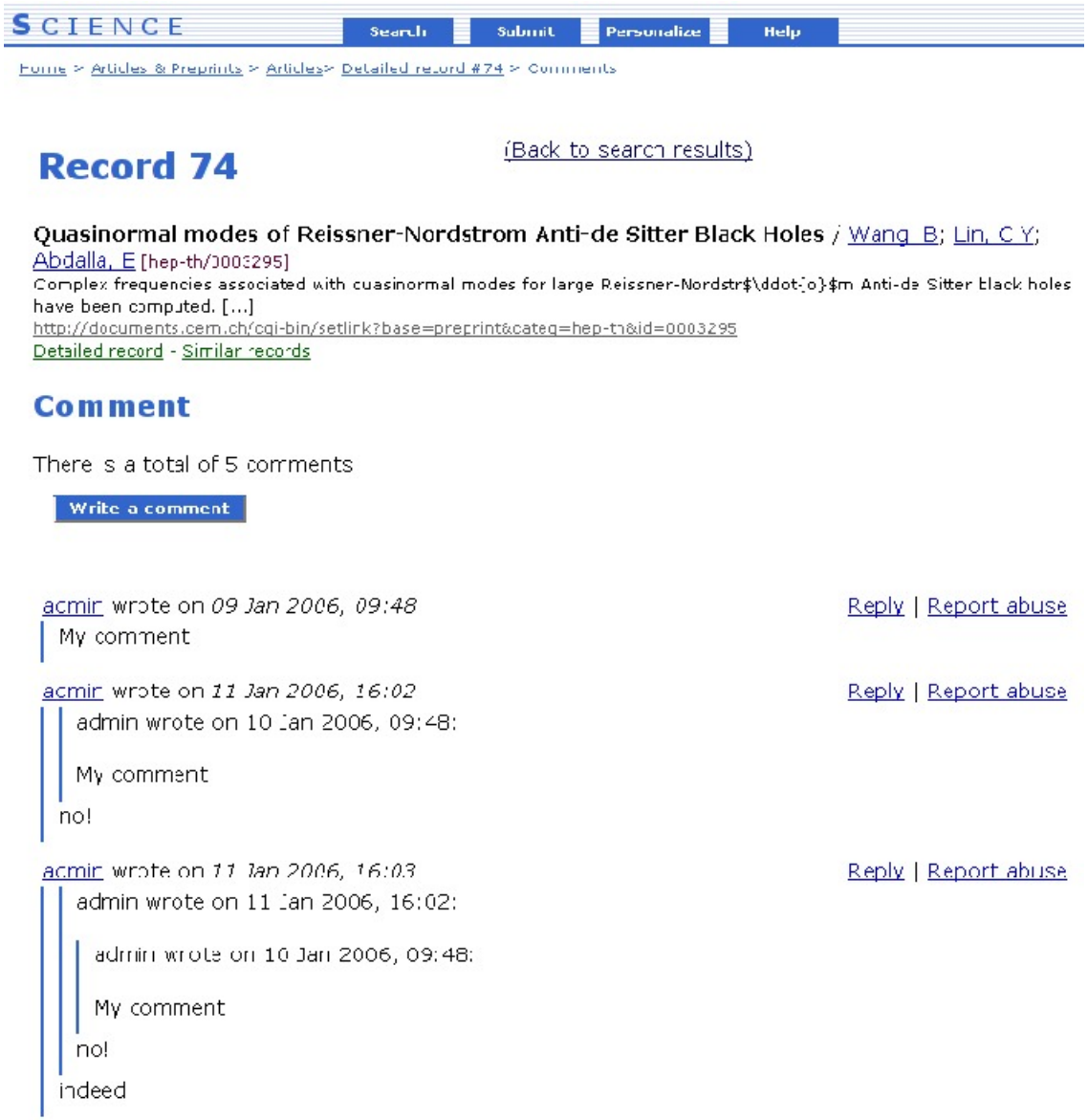
Display results: 10 results | single list

Output format: HTML portfolo

Photos 178 records found 1 - 12 jump to record: 1 Search took 0.23 seconds.

Photos : **178** records found 1 - 12 jump to record: 1

CDS Features: Commenting



Invenio Features: Reviewing

People who viewed this page also viewed:

(3) **The Feyman lectures on physics** - Feynman, Richard Phillips *et al*

(3) **Learning Windows server 20032nd ed. ;** - Hassell, Jonathan

(2) **With the unveiling of its new sign, the CERN Control Centre was officially inaugurated on Thursday 16 March.** - IT-UDS-AVC Team - CERN-VIDEOCLIP-2006-08

(2) **Liability hedging and portofolio choice** - Scherer, Bernd

(2) **Conduite de projet Web2e éd. ;** - Bordage, Stephane

Rate this document:

Average review score: ★★★★★ based on 1 reviews

Readers found the following reviews to be most helpful.

★★★★★ **A wonderful (and fun) guide to Common Lisp**

Reviewed by tsi on 14 Nov 2006, 17:48

0 out of 0 people found this review useful

(Test.) I've been recommending this text to people who want to start learning Common Lisp since it was first available in draft form on the author's web site. Now that it's out in print I can enthusiastically recommend that anybody who is interested in learning Common Lisp - or even curious about how the language can improve your productivity - purchase it.

Peter has a very enjoyable and easy-to-understand writing style, and he starts early with practical examples that show how Common Lisp can be used to solved problems. Chapter 3, 'A Simple Database", is a great explanation of how programs are grown from pieces in Common Lisp to solve large problems. It's presented early and draws people in to the problem solving techniques used when programming in Lisp.

CDS: Create Personal Alert

Search:

neutrino mixing | any field | Search | Browse

Search Tips :: Advanced Search

Results overview: Found **4,236** records in 0.07 seconds.

Articles & Preprints, **4,193** records found
Books & Proceedings, **19** records found
Presentations & Talks, **11** records found
Multimedia & Outreach, **13** records found

1. **Constraining sterile neutrinos with a low energy beta-beam** / Agarwalla, Sanjib Kumar
Task hep-ex
We study the possibility to use a low energy beta-beam facility to search for sterile neutrinos by measuring the disappearance of electron anti-neutrinos. This channel is particularly sensitive since it allows to use inverse beta decay as detection reaction; thus it is free from hadronic uncertainties, provided the neutrino energy is below the pion production threshold. [...]
arXiv:1006.1640; VPI-IPNAS-10-10.- 2010 - Published in : Published in AIP Conf.Proc.: 1222 (2010) , pp. 169-173 Preprint
Detailed record - Similar records

Interested in being notified about new results for this query?
Set up a personal email alert or subscribe to the RSS feed.

CDS: Add to Personal Basket

☐ 8. **Bridging flavour violation and leptogenesis in SU(3) family models** / Calibbi, Lorenzo (Max-Planck-Institut fuer Physik) ; Chun, Eung Jin (Korea Institute for Advanced Study)
We reconsider basic, in the sense of minimal field content, Pati-Salam x SU(3) family models which make use of the Type I see-saw mechanism to reproduce the observed mixing and mass spectrum in the neutrino sector. [...]
arXiv:1005.5563 ; KIAS-P10014 ; IC-2010-021 ; MPP-2010-58. - 2010.
Preprint
Detailed record - Similar records

☐ 9. **Rare muon and tau decays in A4 Models** / Feruglio, Ferruccio ; Paris, Alessio
We analyze the most general dimension six effective Lagrangian, invariant under the flavour symmetry A4 x Z3 x U(1) proposed to reproduce the near tri-bimaximal lepton mixing observed in neutrino oscillations. [...]
arXiv:1005.5526 ; DFPD-10-TH-9. - 2010.
Preprint
Detailed record - Similar records

☐ 10. **Quark and lepton mixing angles with a dodeca-symmetry** / Kim, Jihn E ; Seo, Min-Seok
The discrete symmetry D_12 at the electroweak scale is used to fix the quark and lepton mixing angles. [...]
arXiv:1005.4684. - 2010.
Preprint
Detailed record - Similar records

ADD TO BASKET

CDS: Display Personal Basket

Home > Your Account > Your Baskets > Personal baskets > Physics > Standard Model

Display baskets

Personal baskets > Physics Back to Your Baskets | Create basket | Edit topic

Standard Model (3)

Standard Model Edit basket | Delete basket
3 items, 2 notes
last update: 11 Jun 2010, 14:21

1. **Non-Abelian Flat Directions in a Minimal Superstring Standard Model** / Cleaver, G B ; Faraggi, A E ; Nanopoulos, Dimitri V ; Walker, J W [ACT-2000-1] [CTP-TAMU-2000-2] [OUTP-2000-03-P] [TPI-MINN-2000-6] [hep-ph/0002060]
   Recently, by studying exact flat directions of non-Abelian singlet fields, wedemonstrated the existence of free fermionic heterotic-string models in whichthe SU(3)_C x SU(2)_L x U(1)_Y-charged matter spectrum, just below the stringscale, consists solely of the MSSM spectrum. [...]
   Published in **Mod. Phys. Lett. A: 15 (2000) pp. 1191-1202**
   Fulltext: PDF; PS.GZ
   Detailed record - Notes (2) Copy item | Remove item

2. **Precise calculation of parity nonconservation in cesium and test of the standard model** / Dzuba, V A ; Flambaum, V V ; Ginges, J S M [hep-ph/0204134]
   We have calculated the 6s-7s parity nonconserving (PNC) E1 transition amplitude, E_{PNC}, in cesium. [...]
   Fulltext: PDF; PS.GZ

CDS: Organize and Share Your Baskets

Home > Your Account > Your Baskets > Personal baskets

Display baskets

Personal baskets | Group baskets | Public baskets

Physics (1)
Standard Model

Programming (3)
Linux, Python, SQL

Search baskets for:

in Your personal baskets Search

☐ Search also in notes (where allowed)

CDS: Journals and Bulletins

CERN **The Bulletin**

Archives | Contact us | Sign Up! | Staff Association | CERN Home

search english | français

Issue No. 23-24/2010 - Monday 7 June 2010

News Articles | Official News | Training and Development | General Information | Staff Association

Lyn Evans decelerates!

After more than 40 years at CERN, 15 of which were dedicated to ensuring that the LHC comes to completion, Lyn Evans is retiring. The Imperial College Professor and recently-elected Fellow of the British Royal Society has set himself new challenges, but plans to keep strong links with CERN. His big thank you goes to the many hundreds of people who built one of the most complex scientific instruments ever conceived by mankind. >>

What's New

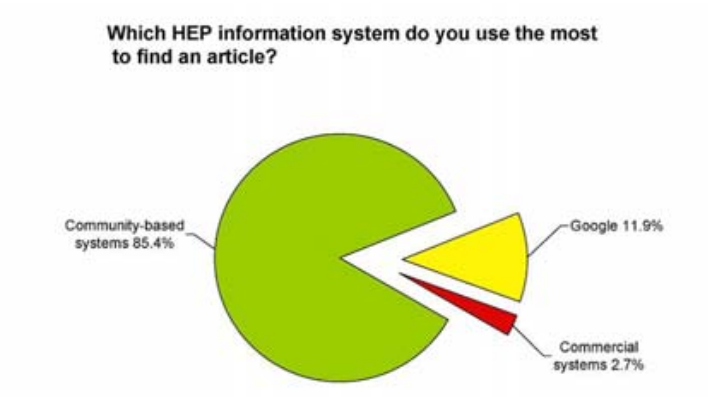
News Articles

- Lyn Evans decelerates!
- Security needs you
- New computer security campaign
- A better beam quality
- Uniting forces in physics and medicine
- Neutrino oscillations make their first appearance in OPERA
- It sounds good!
- "Draw me a physicist" exhibition opens
- Council Chamber exhibition
- Irène Jacob visits CERN
- News from the Library
- Back to the 80s

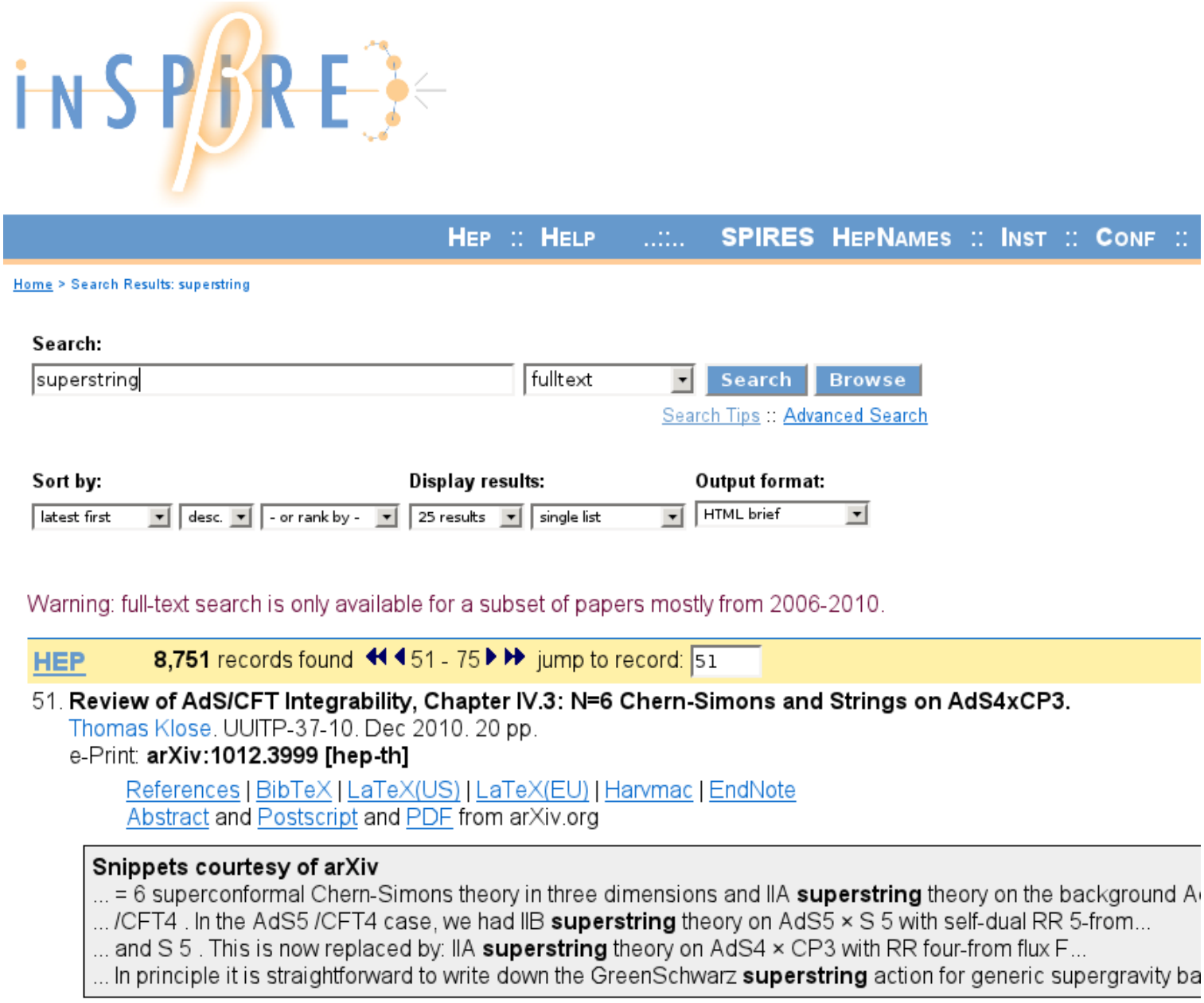
What is digital library?

Example #2: INSPIRE

- world-wide high-energy physics information system
- run by CERN, DESY, FNAL, SLAC
- metadata curation since 1960s, Invenio technology since 2007
- citation analysis, author/affiliation analysis
- close partnership with arXiv and ADS
- `http://inspirehep.net/`



INSPIRE: full-text search



INSPIRE: cite summary

HEP :: HELP ..::.. SPIRES HEPNAM

Home > Search Results: standard model

Search:

standard model | any field | Search | Browse

Search Tips :: Advanced Search

Sort by: latest first | desc. | - or rank by -

Display results: 25 results | single list

Output format: HTML citesummary

Citation summary results	All papers	Published only
Total number of citable papers analyzed:	26,314	16,241
Total number of citations:	671,190	613,629
Average citations per paper:	25.5	37.8
Breakdown of papers by citations:		
Renowned papers (500+)	78	76
Famous papers (250-499)	230	223
Very well-known papers (100-249)	1,050	1,002
Well-known papers (50-99)	1,857	1,748
Known papers (10-49)	7,978	6,787
Less known papers (1-9)	10,261	5,202
Unknown papers (0)	4,860	1,203
Additional Citation Metrics		
h-index	268	265

INSPIRE: citation history

Information | References (43) | Citations (1199) | Files | Plots

Unified Interactions of Leptons and Hadrons - Fritzsch, Harald *et al.* Annals Phys. 93 (1975) 193-266 . CALT-68-467

Cited by: 1199 records

(3876) **Review of Particle Physics** - Particle Data Group Collaboration (Amsler, Claude *et al.*) Phys.Lett. B667 (2008) 1-1340
(3744) **The Inflationary Universe: A Possible Solution to the Horizon and Flatness Problems** - Guth, Alan H. Phys.Rev. D23 (1981) 347-356 . SLA
(1112) **Aspects of the Grand Unification of Strong, Weak and Electromagnetic Interactions** - Buras, A.J. *et al.* Nucl.Phys. B135 (1978) 66-92 . CE
(1095) **mu --> e gamma at a Rate of One Out of 1-Billion Muon Decays?** - Minkowski, Peter Phys.Lett. B67 (1977) 421 . Print-77-0182 (BERN)
(984) **Hierarchy of Quark Masses, Cabibbo Angles and CP Violation** - Froggatt, C.D. *et al.* Nucl.Phys. B147 (1979) 277 . CERN-TH-2519

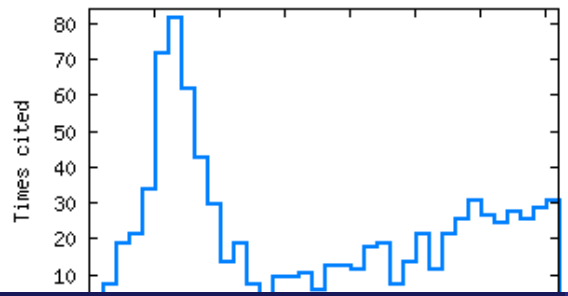
more

Co-cited with: 18929 records

(810) **Unity of All Elementary Particle Forces** - Georgi, H. *et al.* Phys.Rev.Lett. 32 (1974) 438-441
(511) **Lepton Number as the Fourth Color** - Pati, Jogesh C. *et al.* Phys.Rev. D10 (1974) 275-289 . IC/74/7
(367) **Hierarchy of Interactions in Unified Gauge Theories** - Georgi, H. *et al.* Phys.Rev.Lett. 33 (1974) 451-454 . Print-74-1122 Rev. (HARVARD), PRI
(320) **A Model of Leptons** - Weinberg, Steven Phys.Rev.Lett. 19 (1967) 1264-1266
(253) **Unified Lepton-Hadron Symmetry and a Gauge Theory of the Basic Interactions** - Pati, Jogesh C. *et al.* Phys.Rev. D8 (1973) 1240-1251 . IC

more

Citation history:



INSPIRE: author pages

Welcome to INSPIRE β: the up
We now recommend that you
Please send feedback on INSP

HEP :: HELP ..::.. SPIRES HEPNAMES :: INST :: CONF :: EXP :: JOBS

Home > > Search Results

Dixon, Lance J

Name variants

Dixon, Lance J (140)

Papers

All papers (140)
Published (89)
Conference (31)
Review (11)
Lectures (4)
Introductory (2)

Frequent keywords

supersymmetry (46)
quantum chromodynamics (45)
perturbation theory: higher-order (35)
Feynman graph: higher-order (32)
unitarity (24)
string model (18)
electron positron: annihilation (16)

Affiliations

SLAC (89)
unknown affiliation (50)
Saclay (4)
UCLA (4)
CERN (2)
MIT, LNS (2)
Princeton U. (1)
Penn State U. (1)
Durham U., IPPP (1)
Santa Barbara, KITP (1)
Cambridge U., DAMTP (1)
Brown U. (1)
Durham U. (1)

Frequent co-authors

Dixon, Lance J. (140)
Bern, Zvi (38)
Bern, Z. (35)
Kosower, David A. (29)

Outline

1 Introduction
- Digital Library
- Invenio

2 Case Studies
- Episode 1: Python
- Episode 2: Git
- Episode 3: Testing
- Episode 4: Building Efficient Indexes
- Episode 5: NIH
- Episode 6: Scalability

3 Conclusions

Invenio Key Features

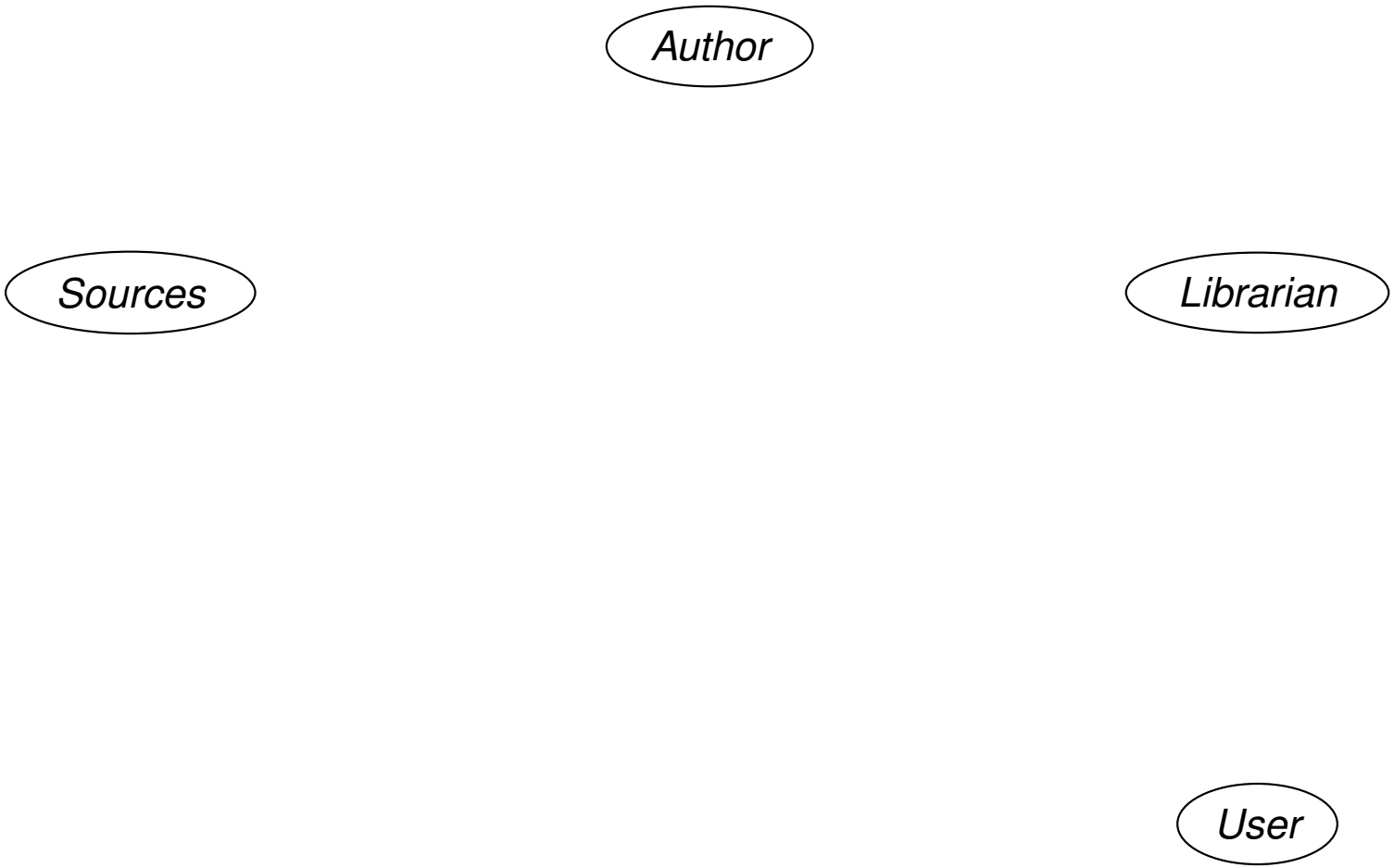
- **navigable collection tree** (regular, virtual)
- **powerful search engine**
  - Google-like speed for up to 5M records
  - combined metadata, reference and fulltext search
- **flexible metadata** (MARC, OA)
  - handling any kind of document (multimedia)
  - customizable input, formatting and linking
- **personalization** and **collaborative** features:
  - alerts, baskets, groups, reviews, comments
  - internationalisation (28 languages)
- **open source**, GNU General Public License
  - co-developed by CERN (2002–), EPFL (2004–), DESY/FNAL/SLAC (2008–), CfA (2009–), Cornell (2011–)
  - installed at 30+ institutions world-wide

Invenio Modules: Overview

Invenio Modules: Overview

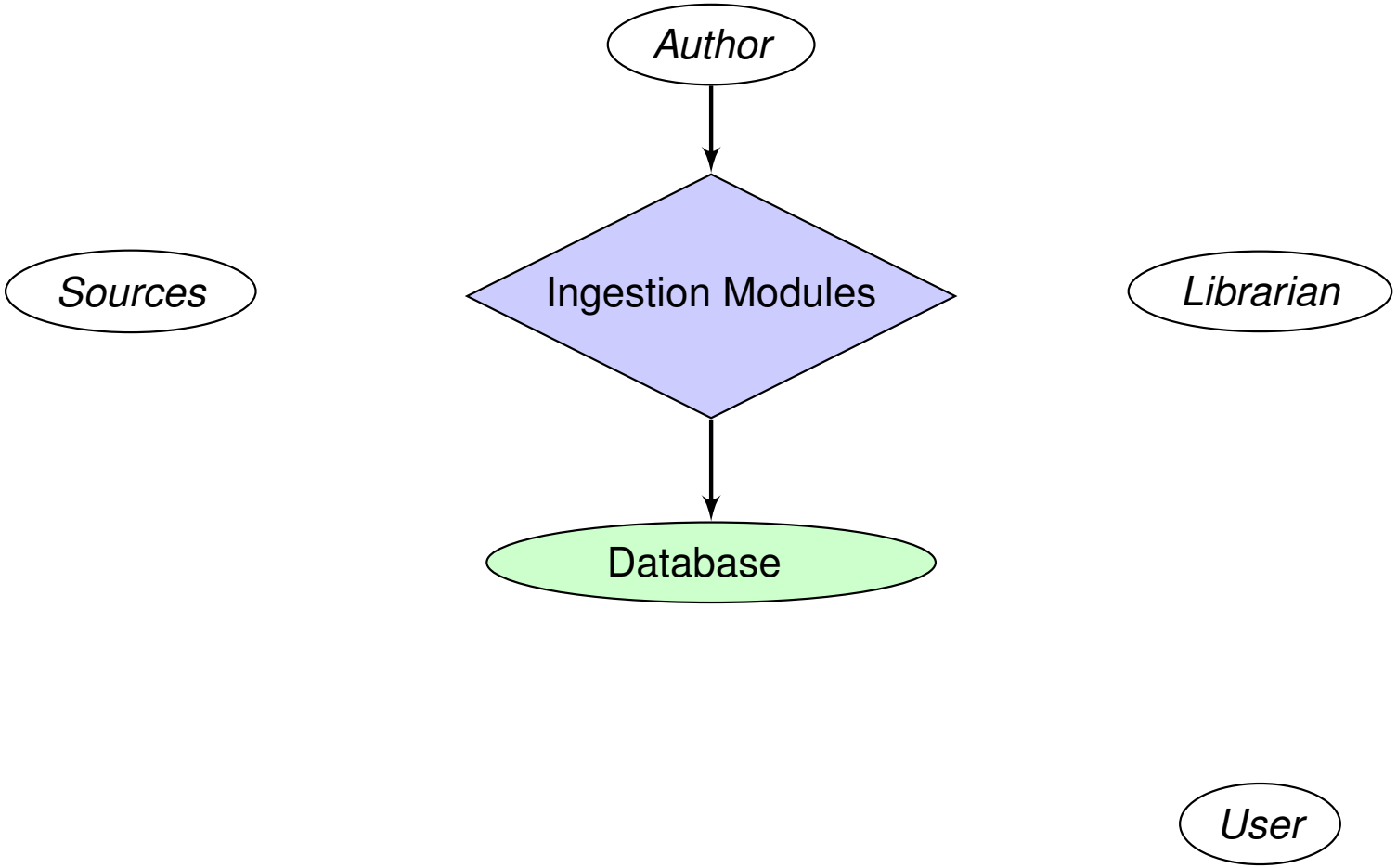
Invenio Modules: Overview

Invenio Modules: Overview

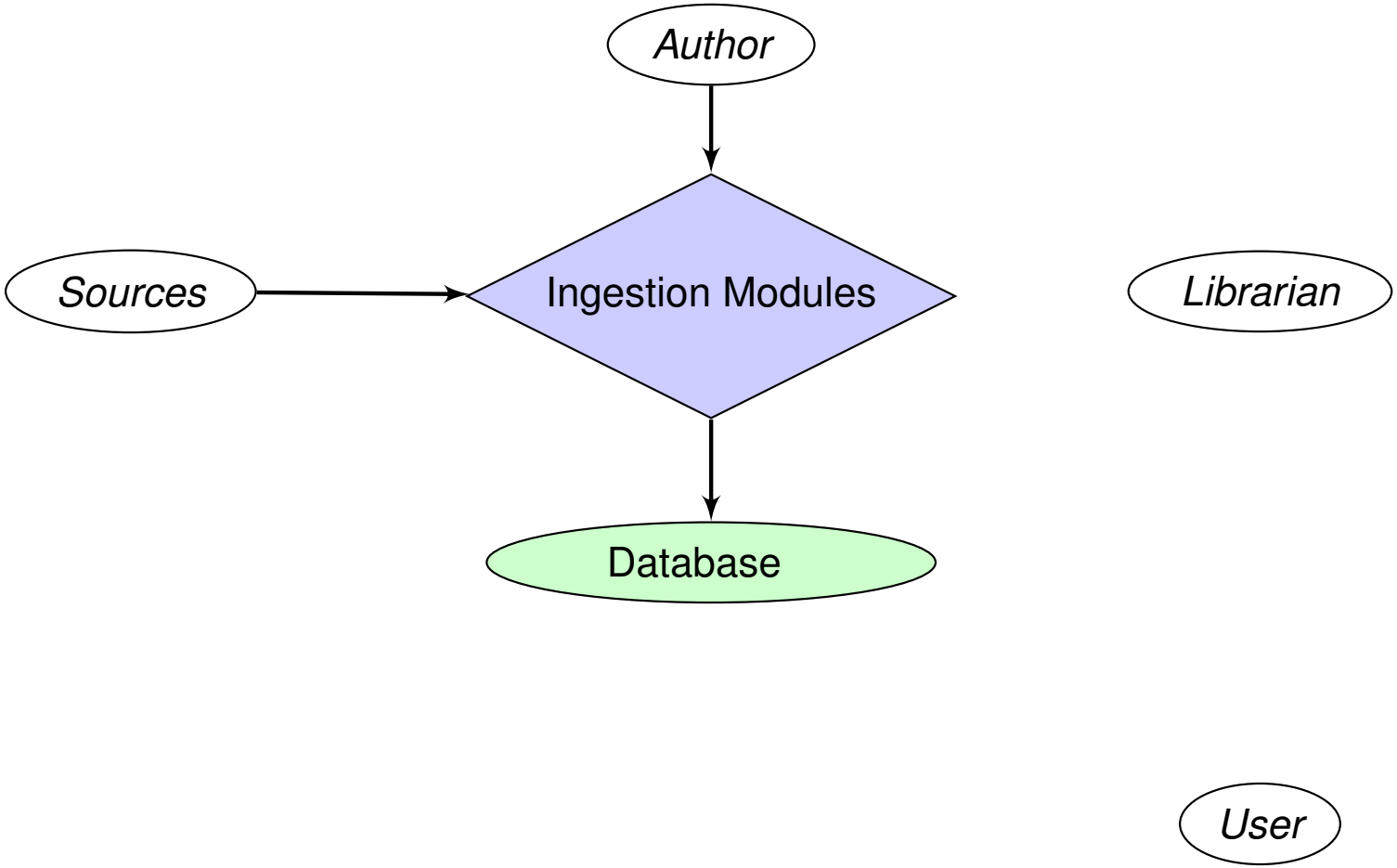


Invenio Modules: Overview

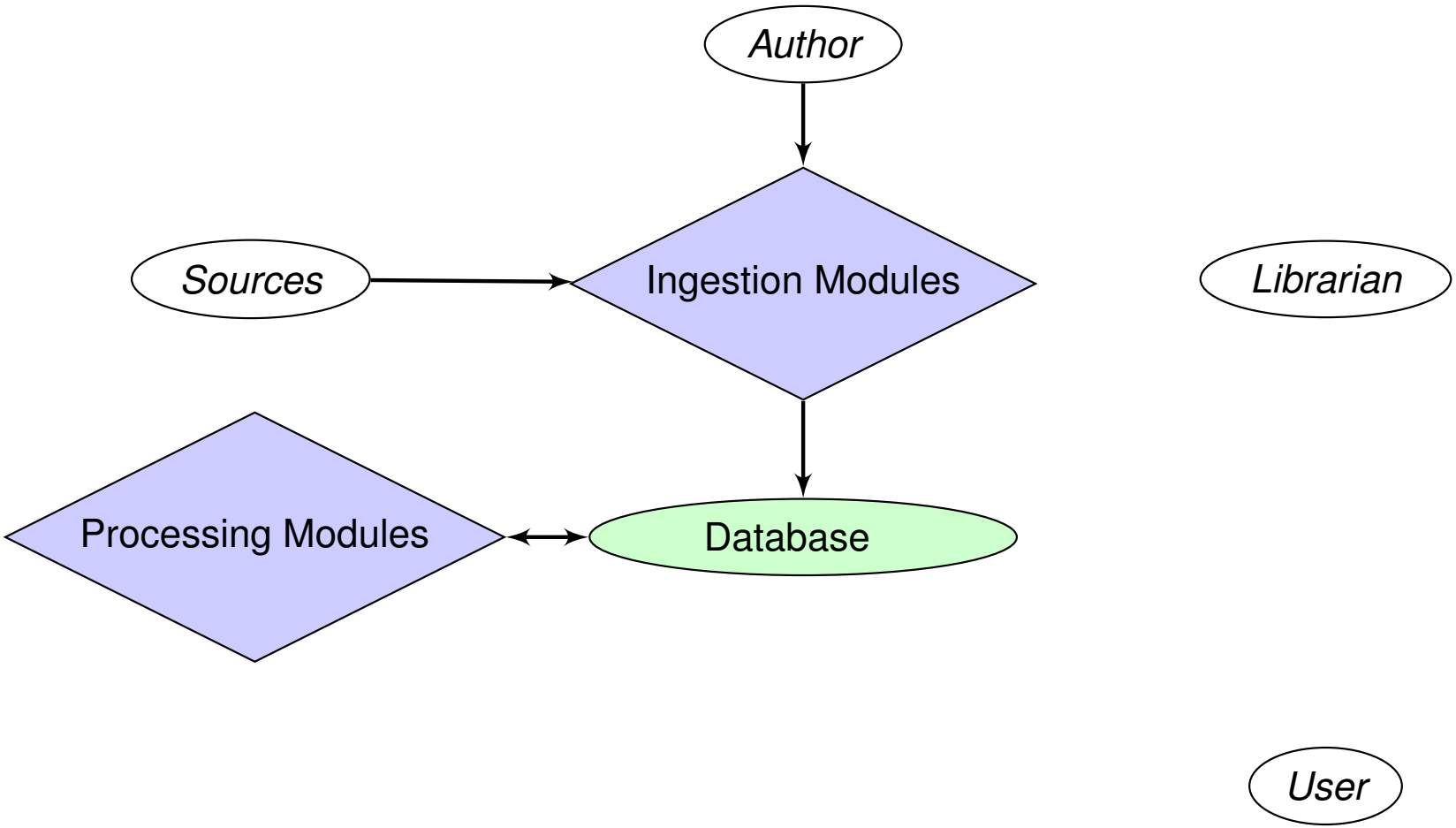
Invenio Modules: Overview



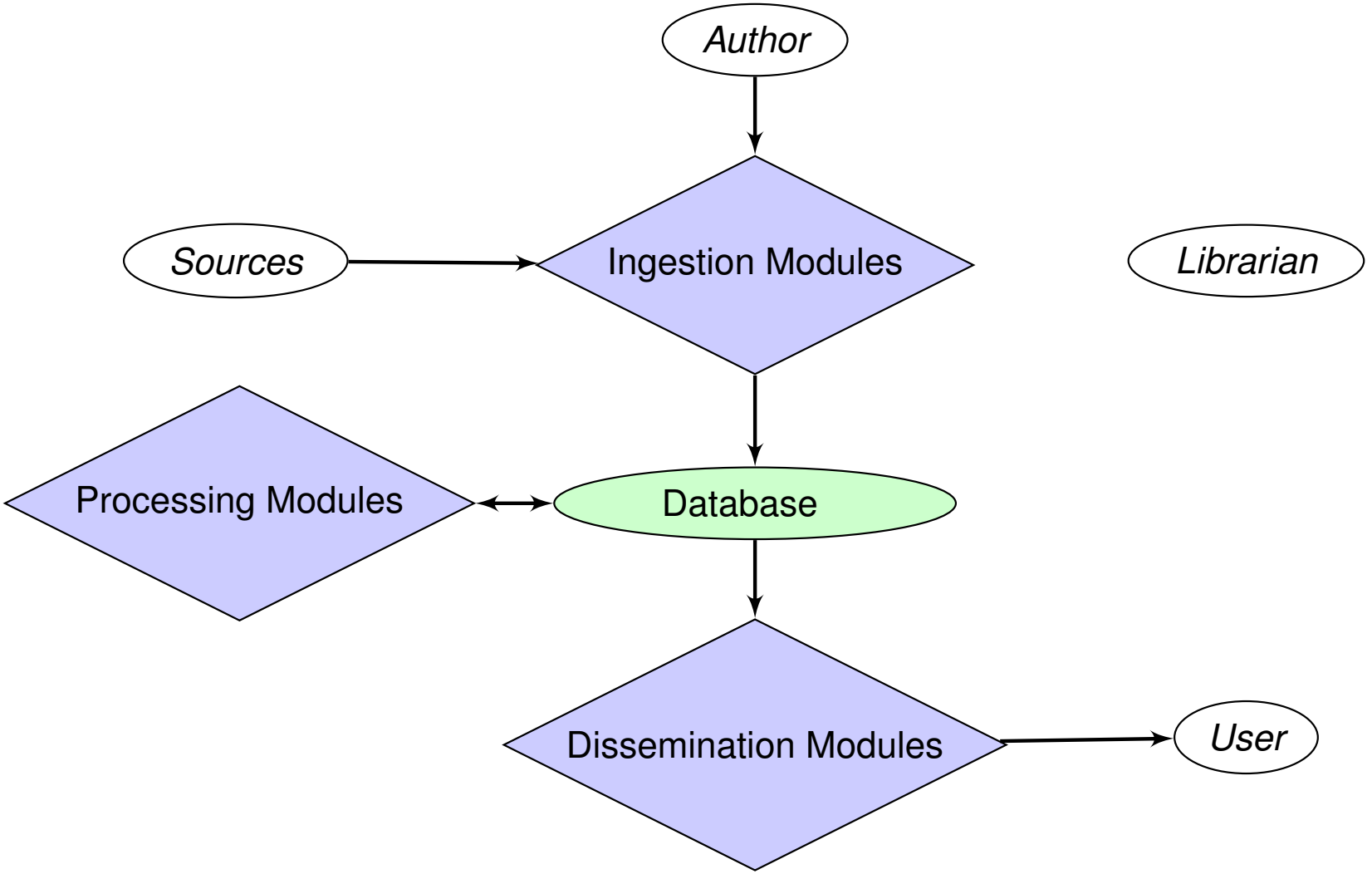
Invenio Modules: Overview



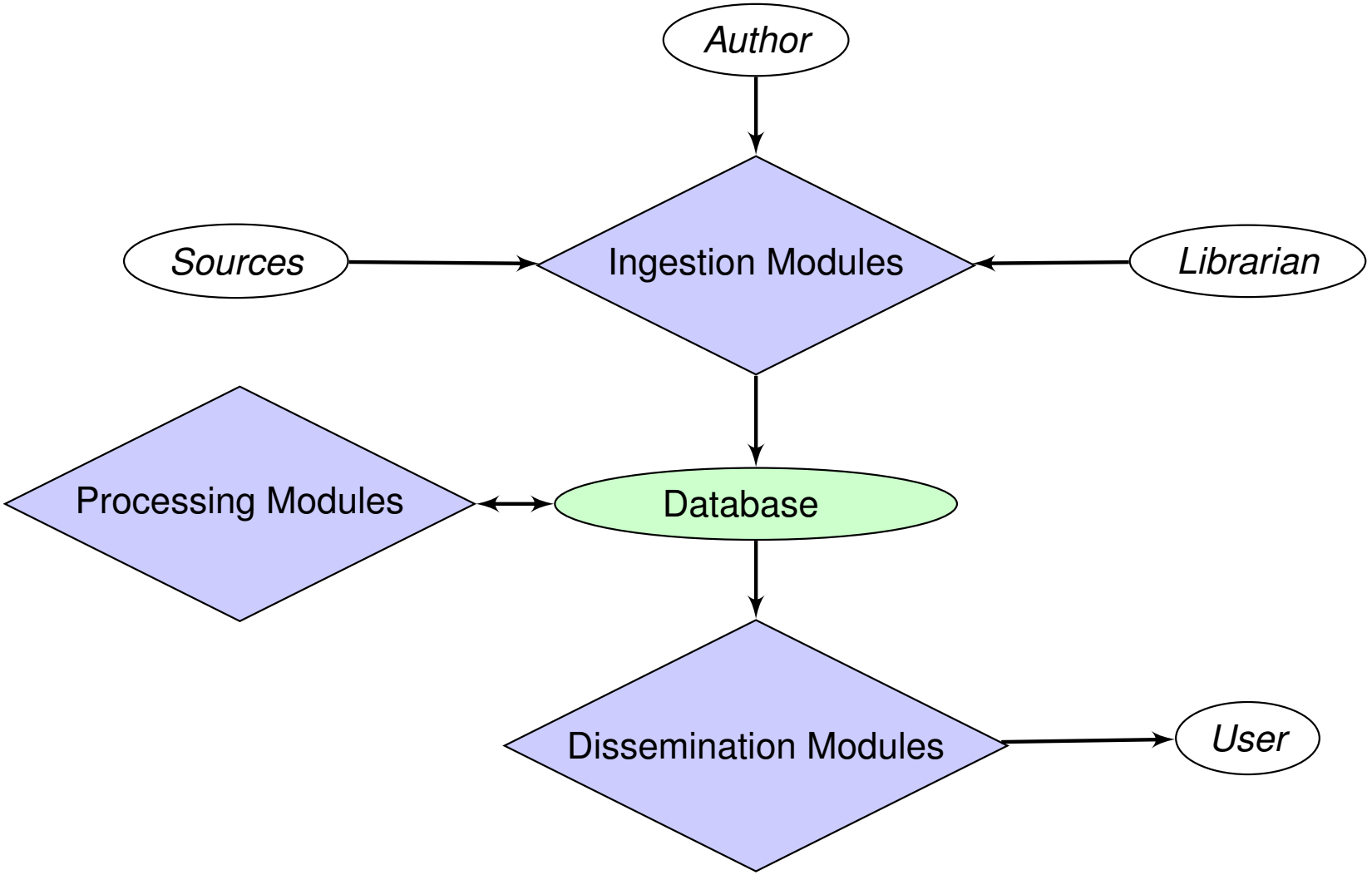
Invenio Modules: Overview



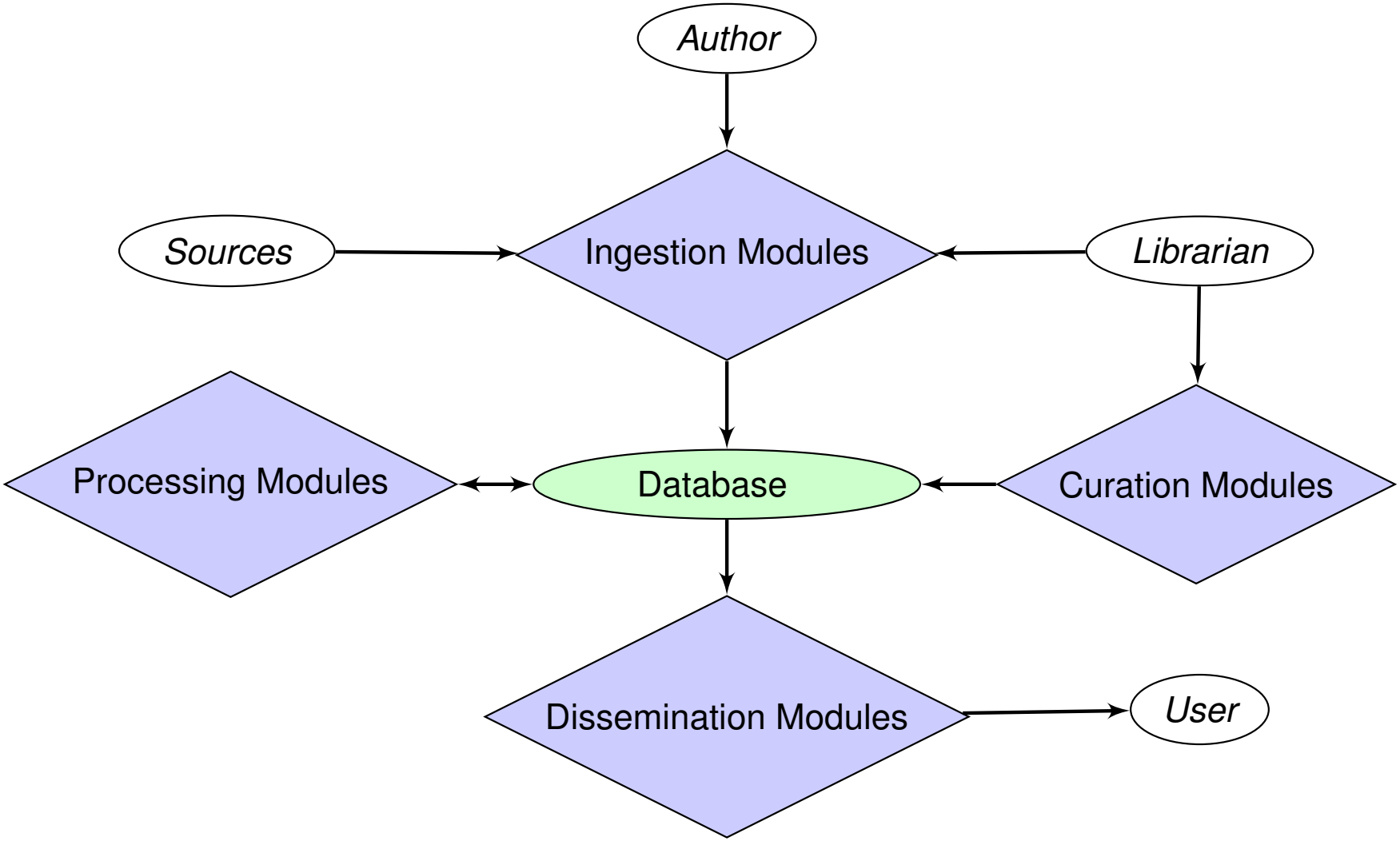
Invenio Modules: Overview



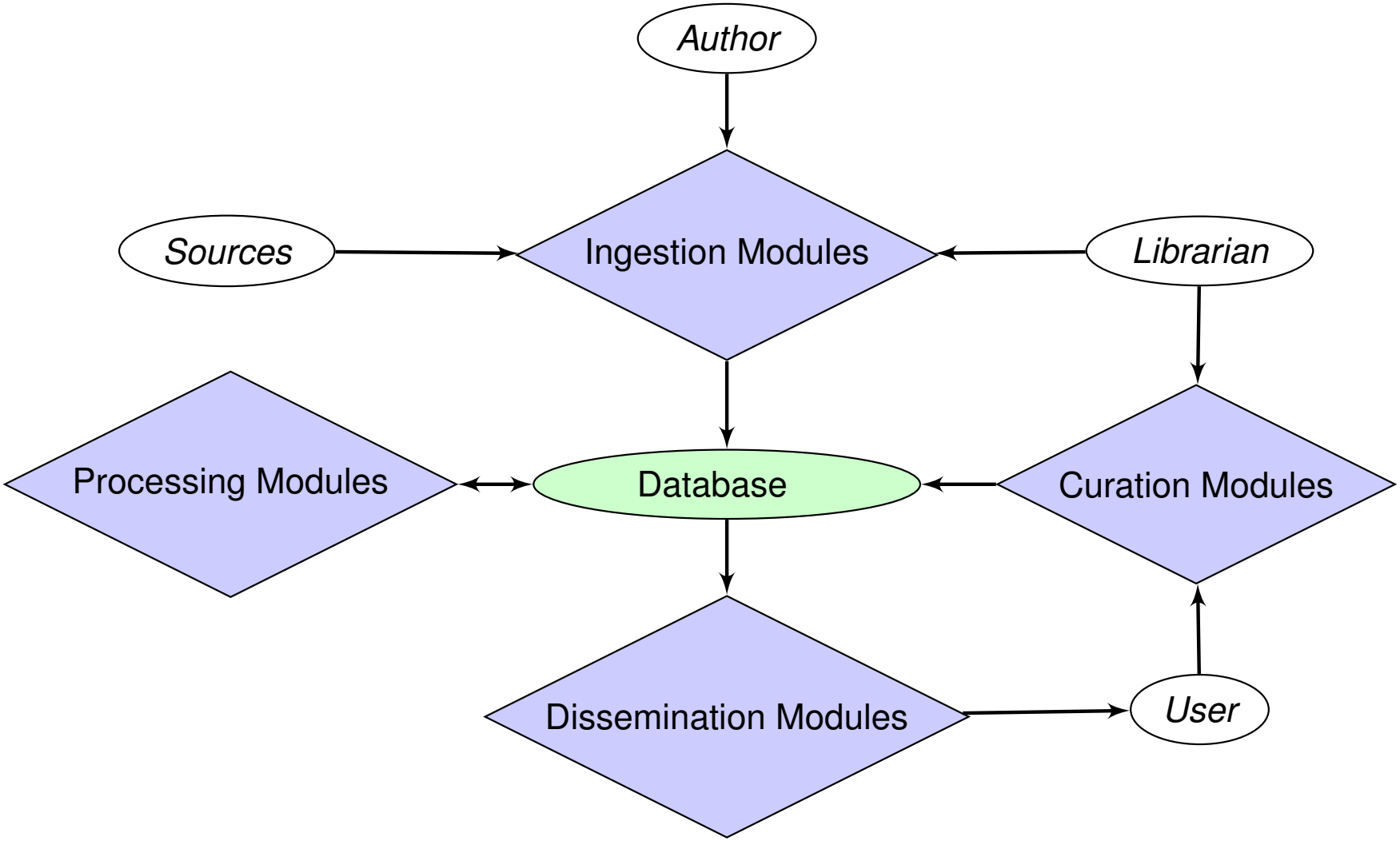
Invenio Modules: Overview



Invenio Modules: Overview



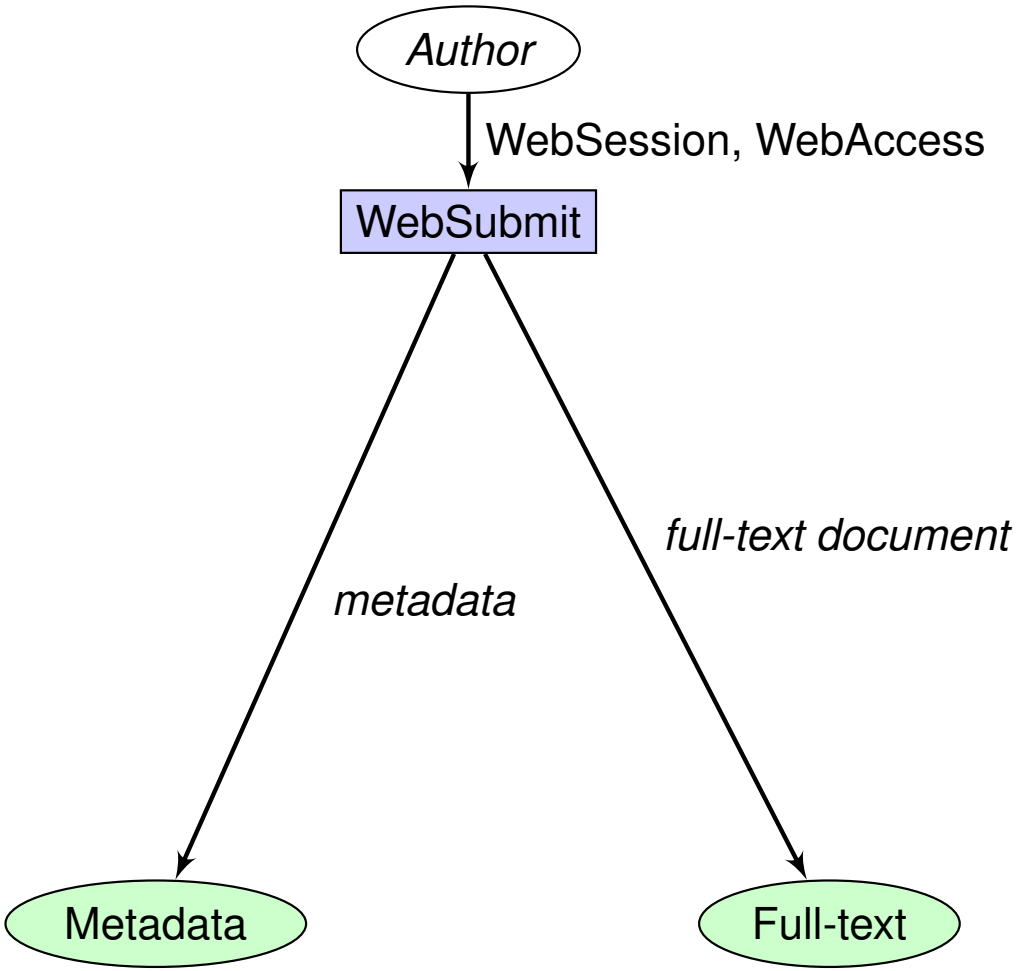
Invenio Modules: Overview



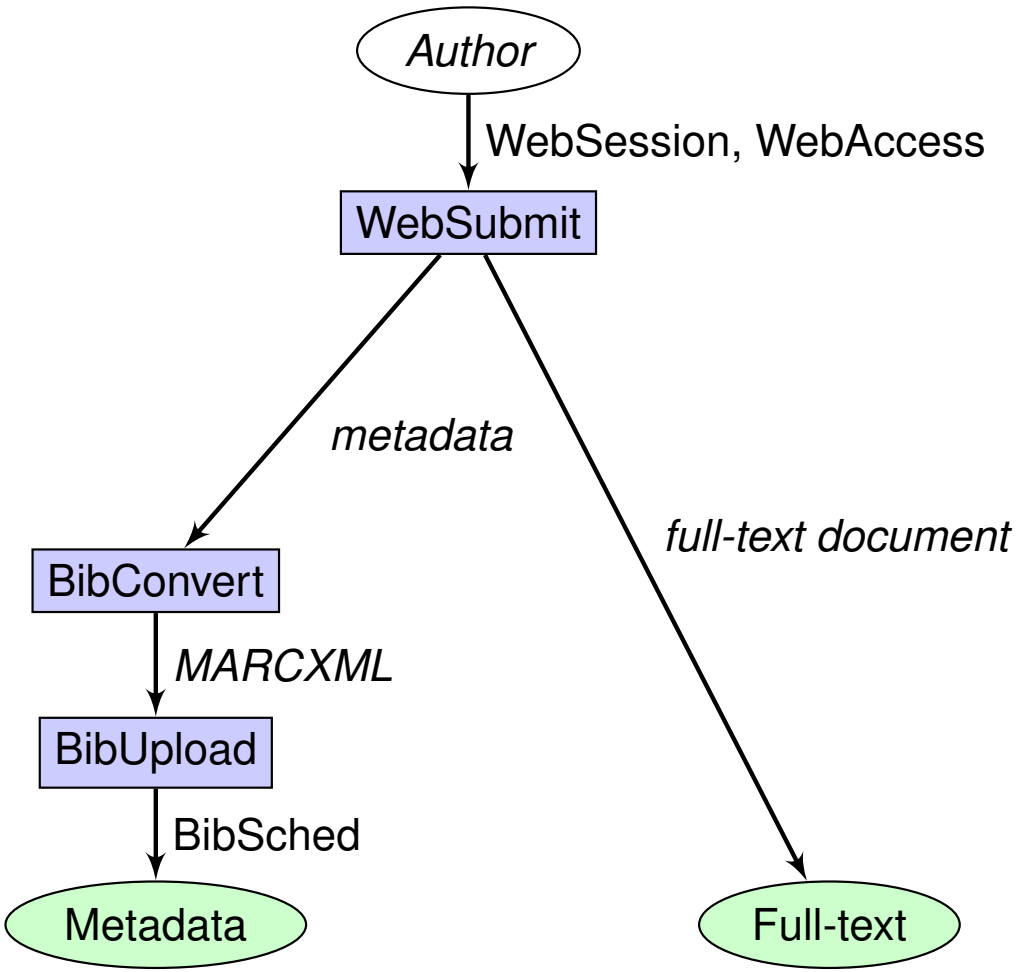
Invenio Modules: Ingestion

Invenio Modules: Ingestion

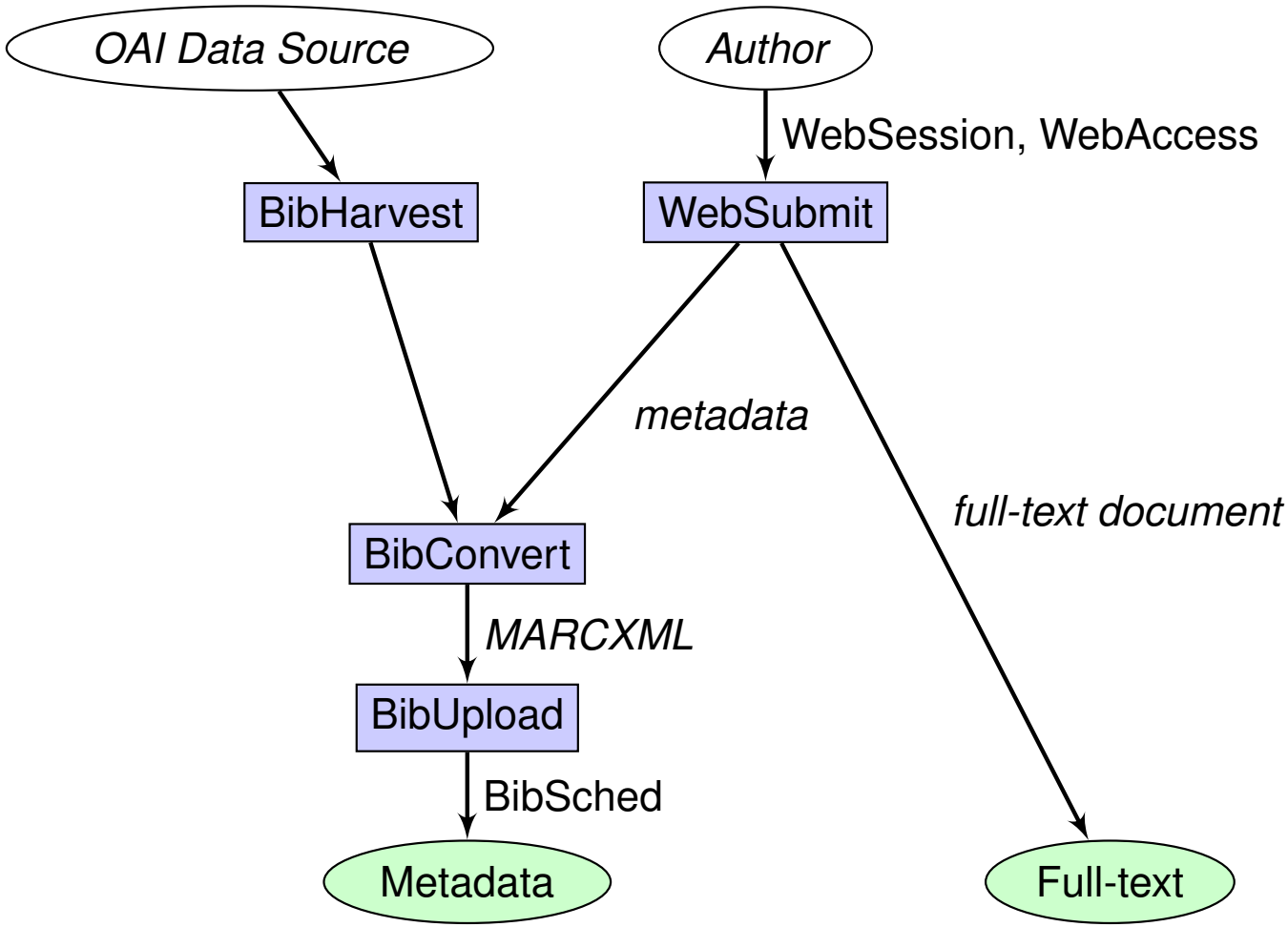
Invenio Modules: Ingestion



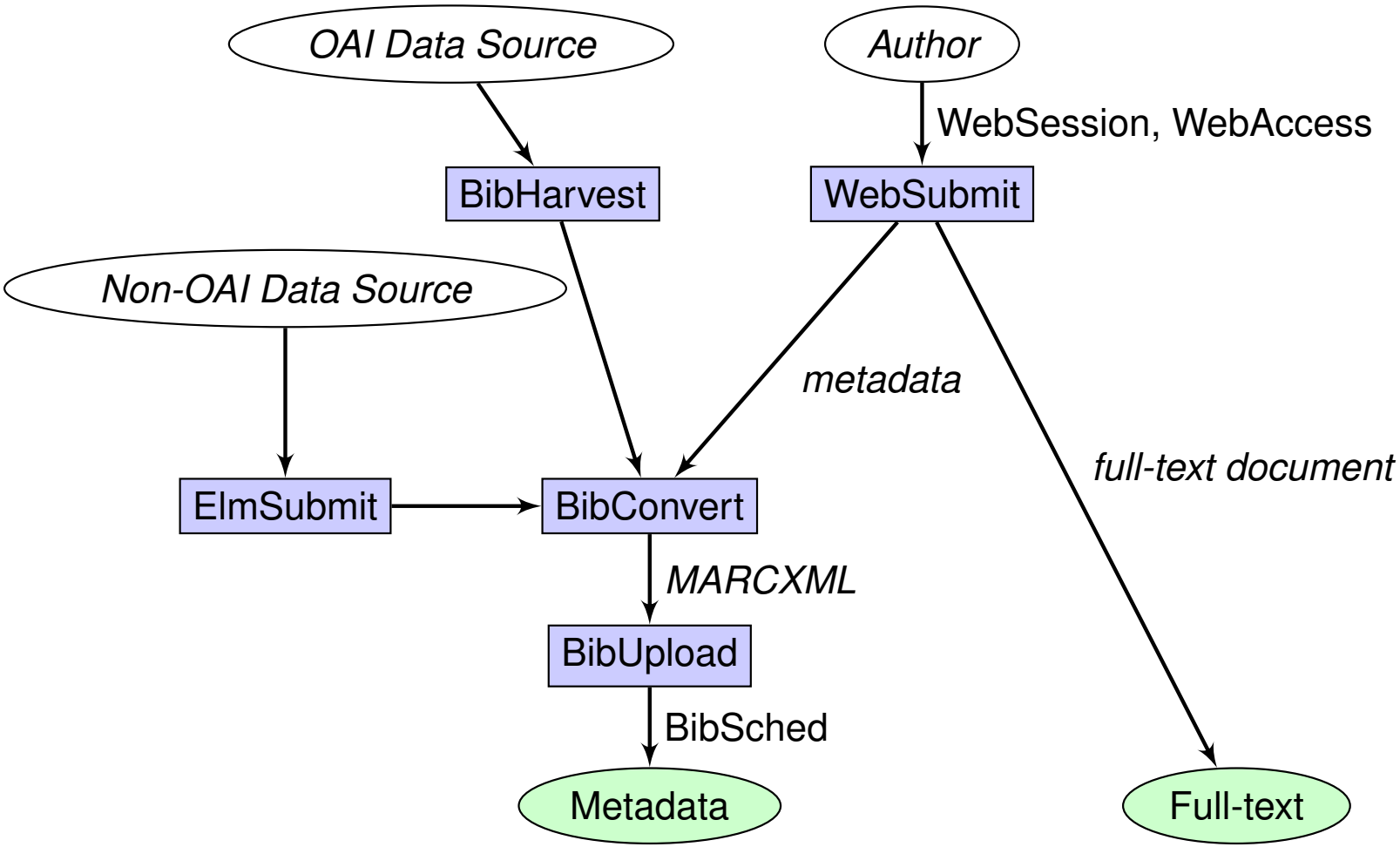
Invenio Modules: Ingestion



Invenio Modules: Ingestion



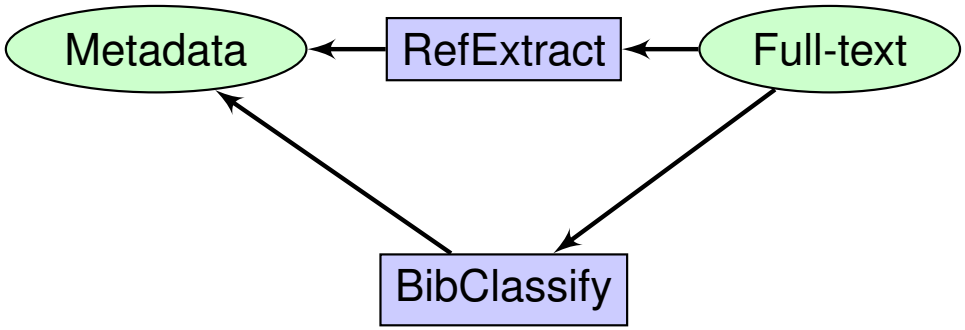
Invenio Modules: Ingestion



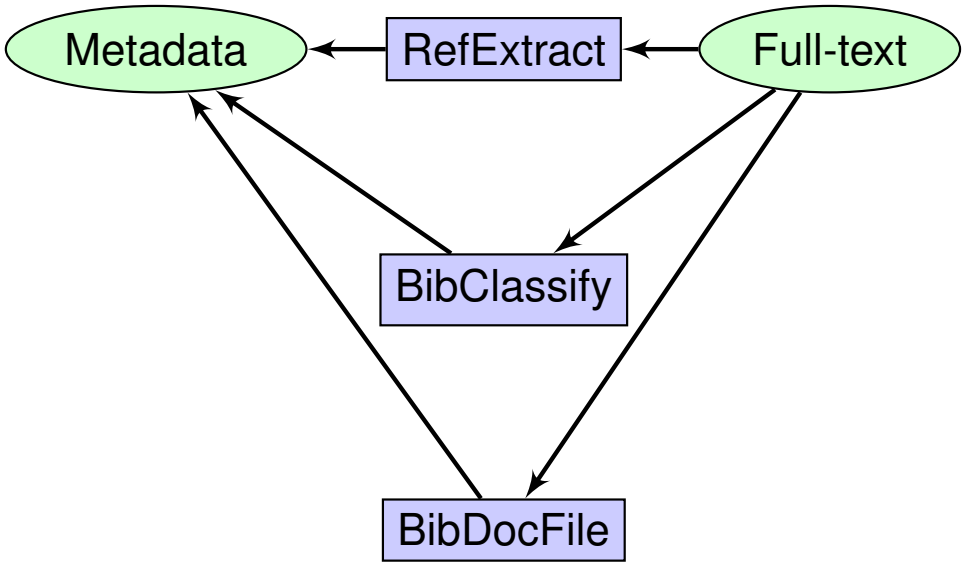
Invenio Modules: Processing

Invenio Modules: Processing

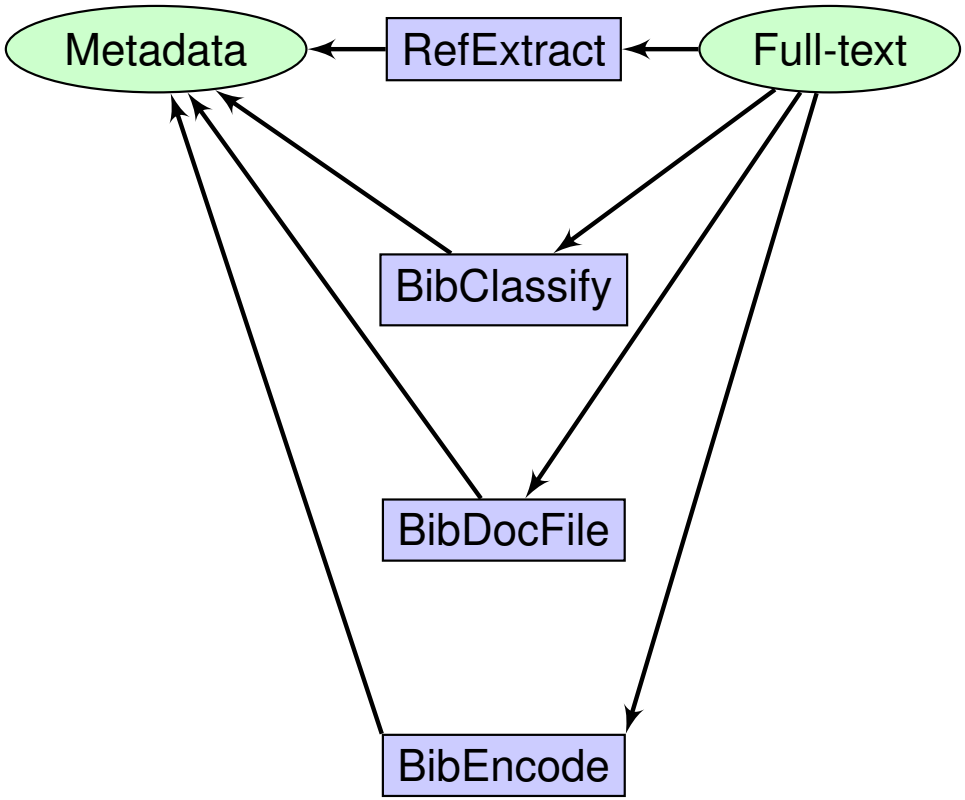
Invenio Modules: Processing



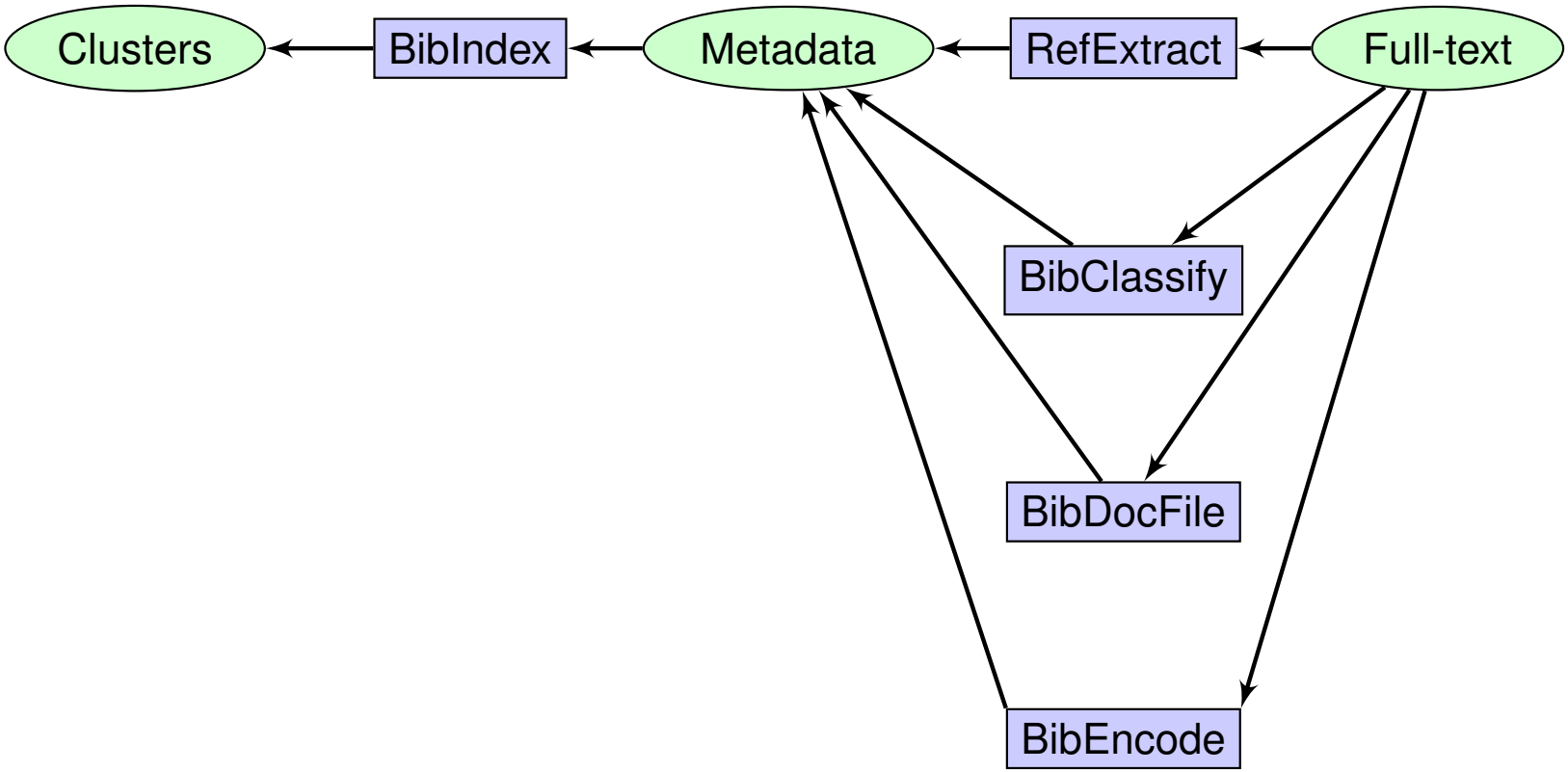
Invenio Modules: Processing



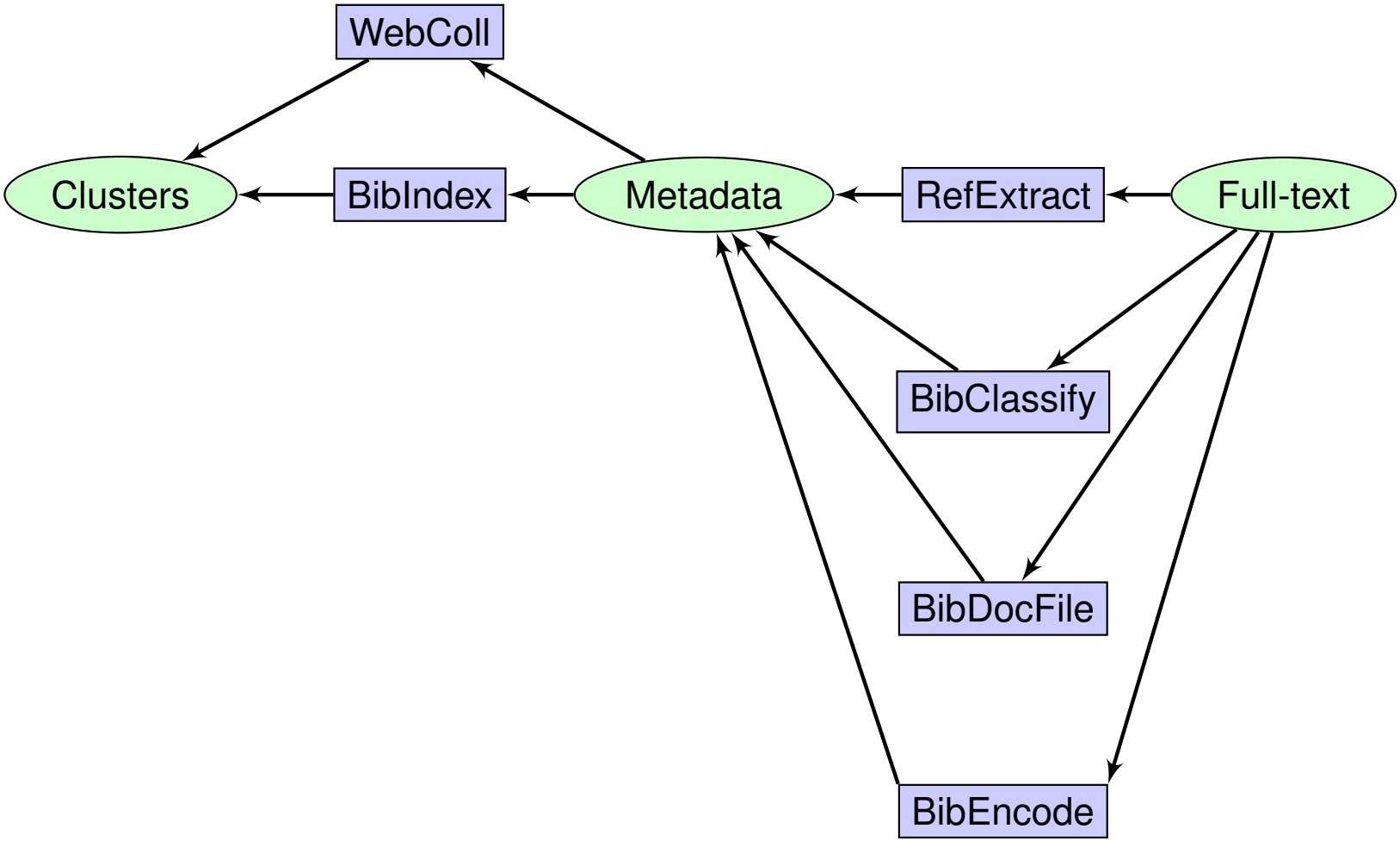
Invenio Modules: Processing



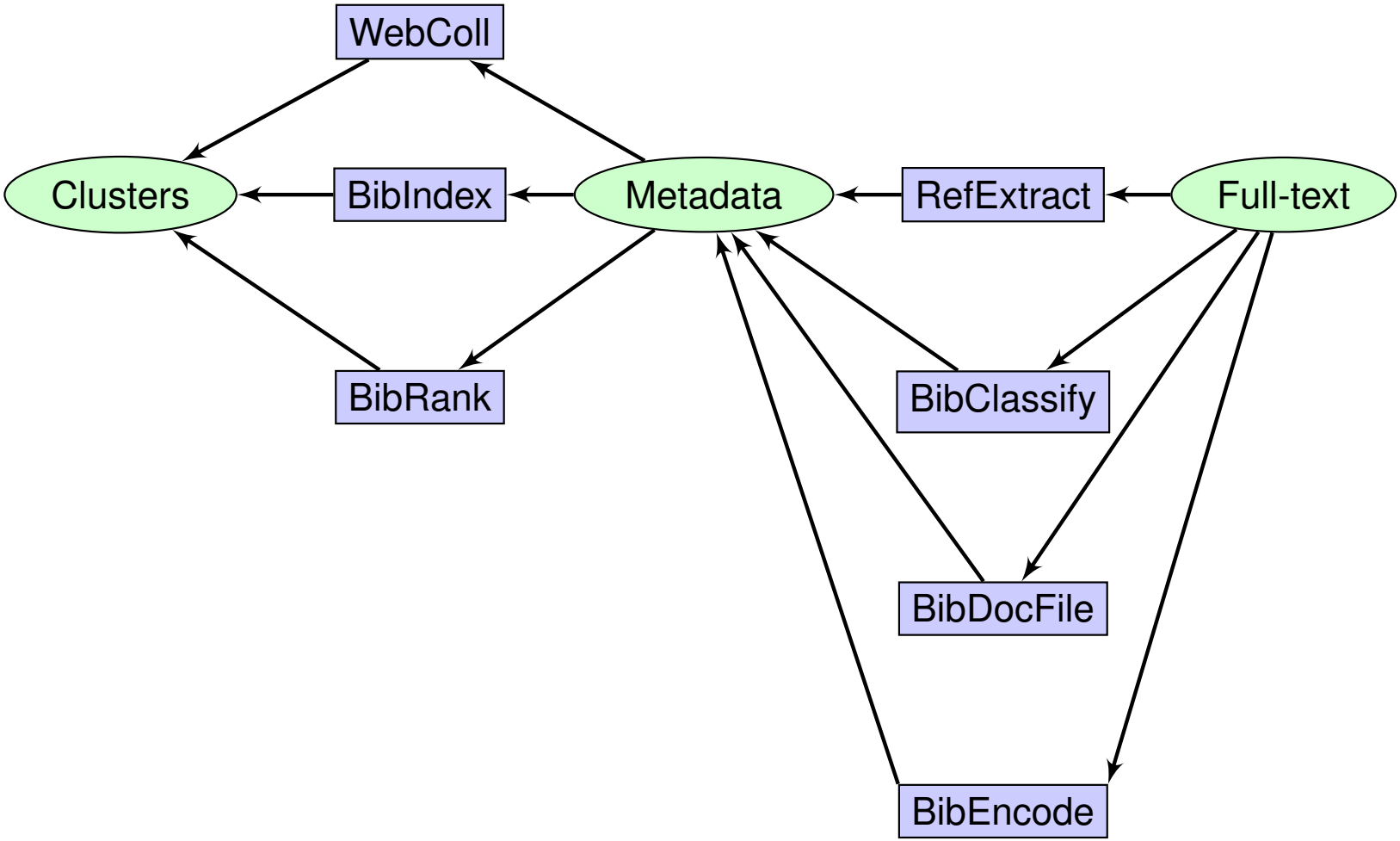
Invenio Modules: Processing



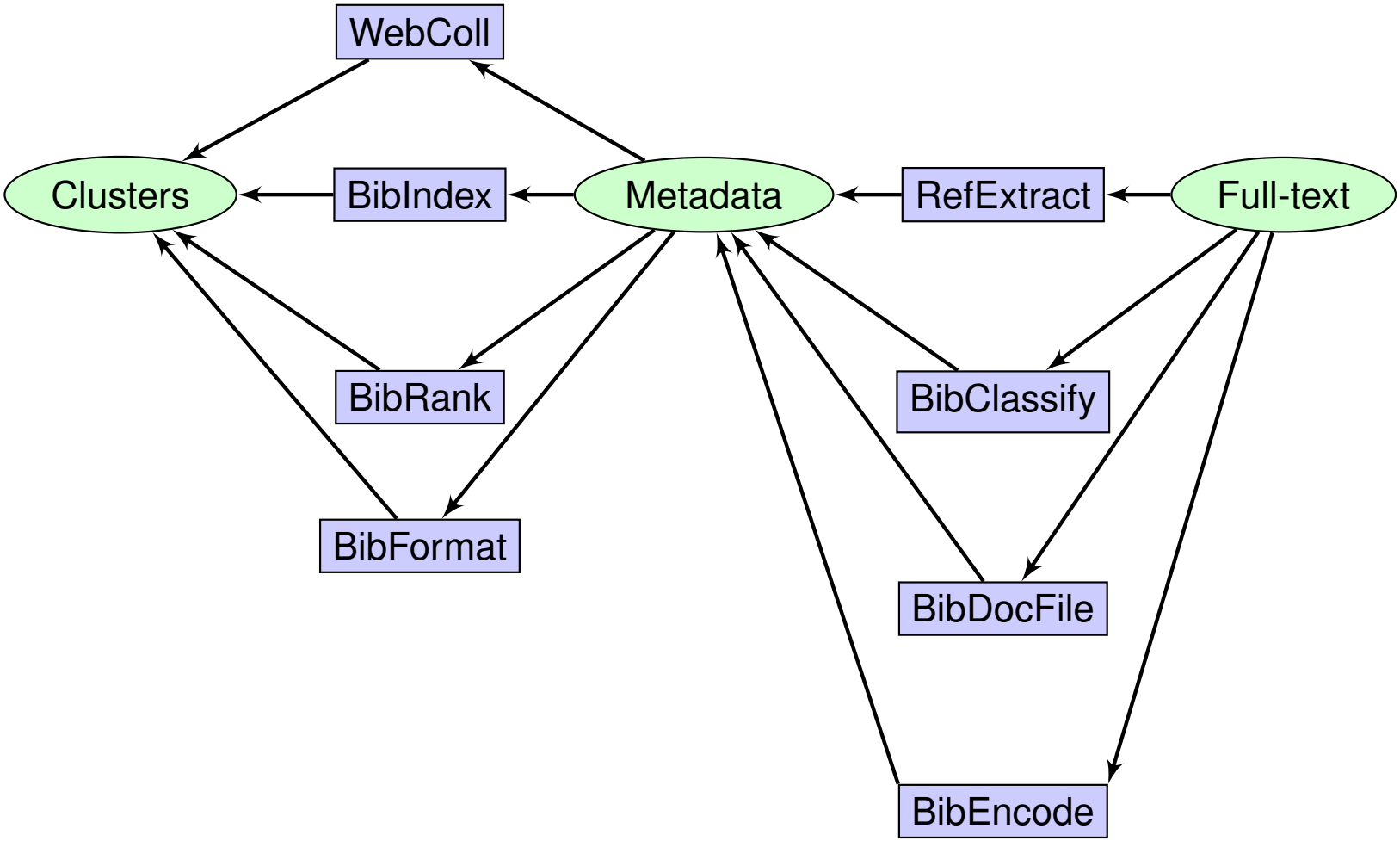
Invenio Modules: Processing



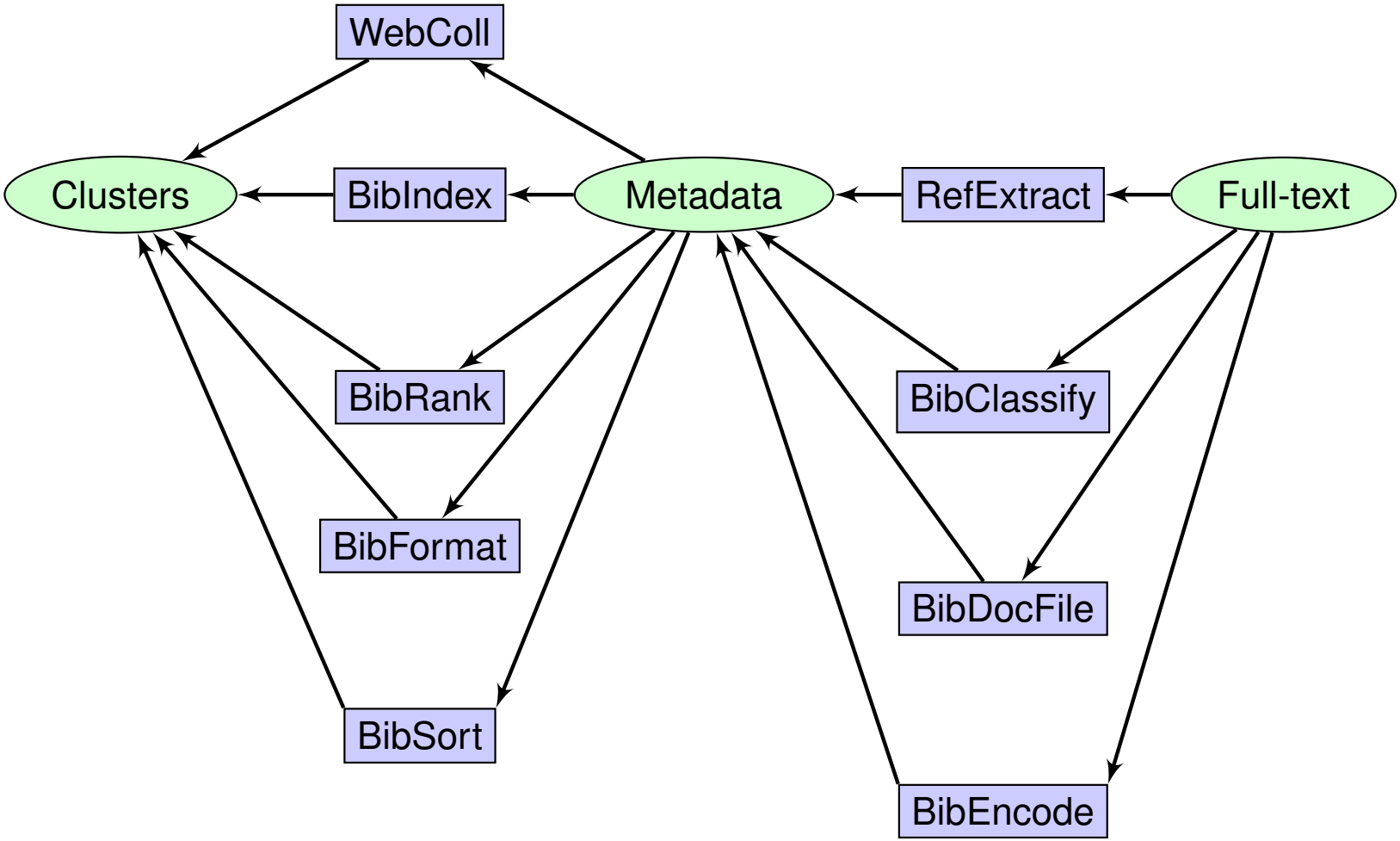
Invenio Modules: Processing



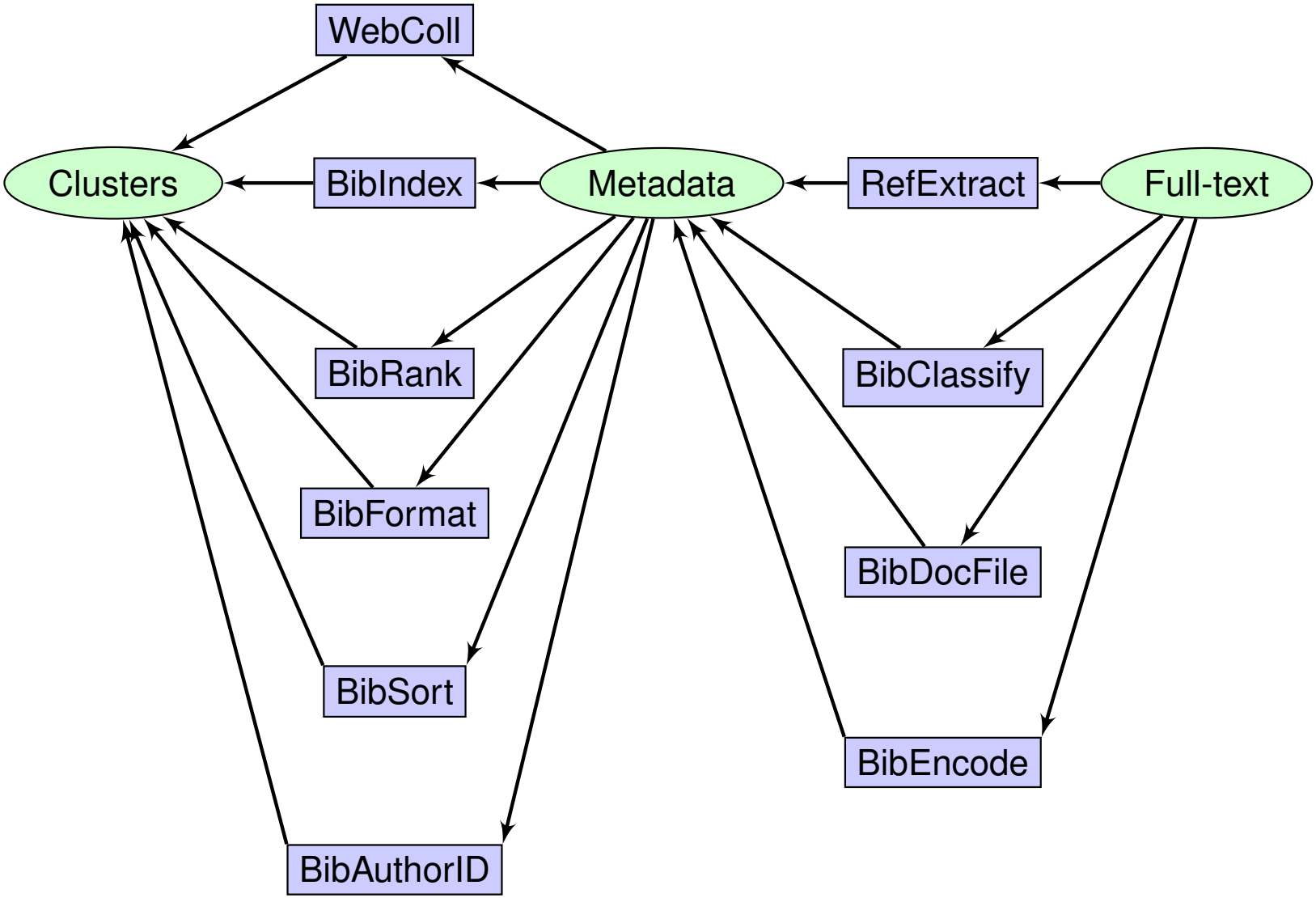
Invenio Modules: Processing



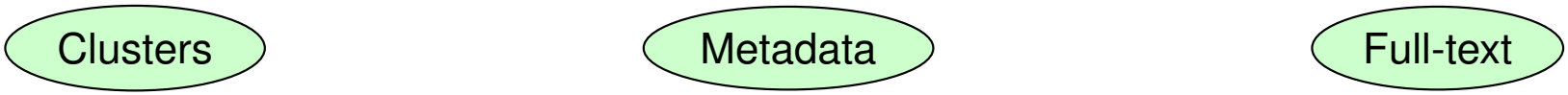
Invenio Modules: Processing



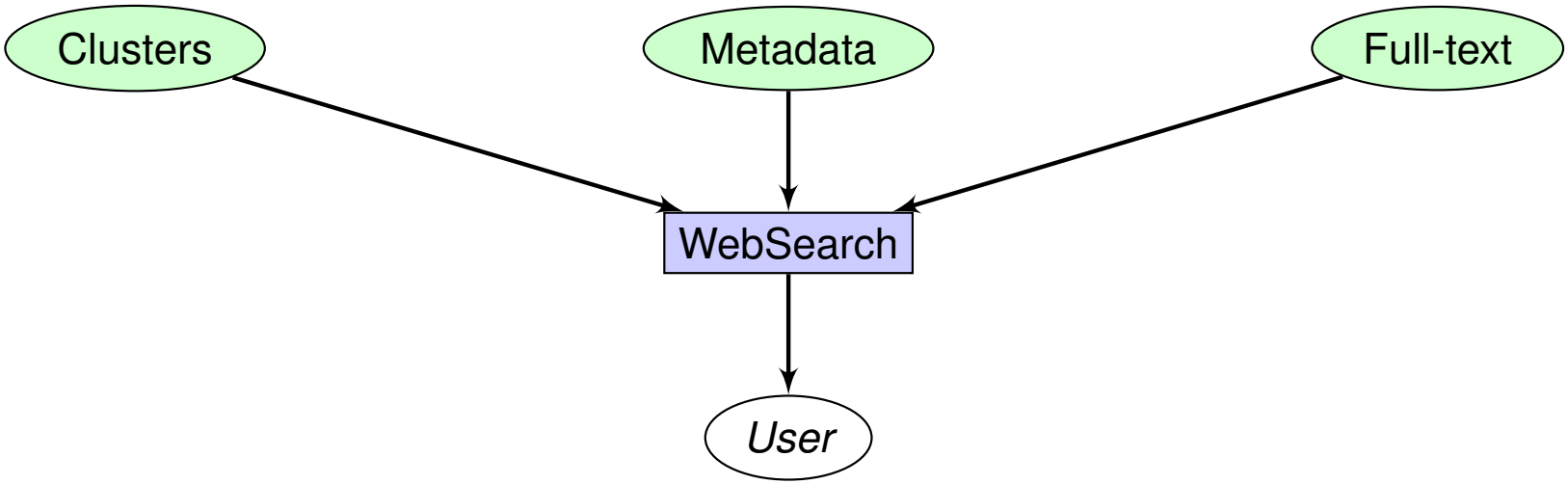
Invenio Modules: Processing



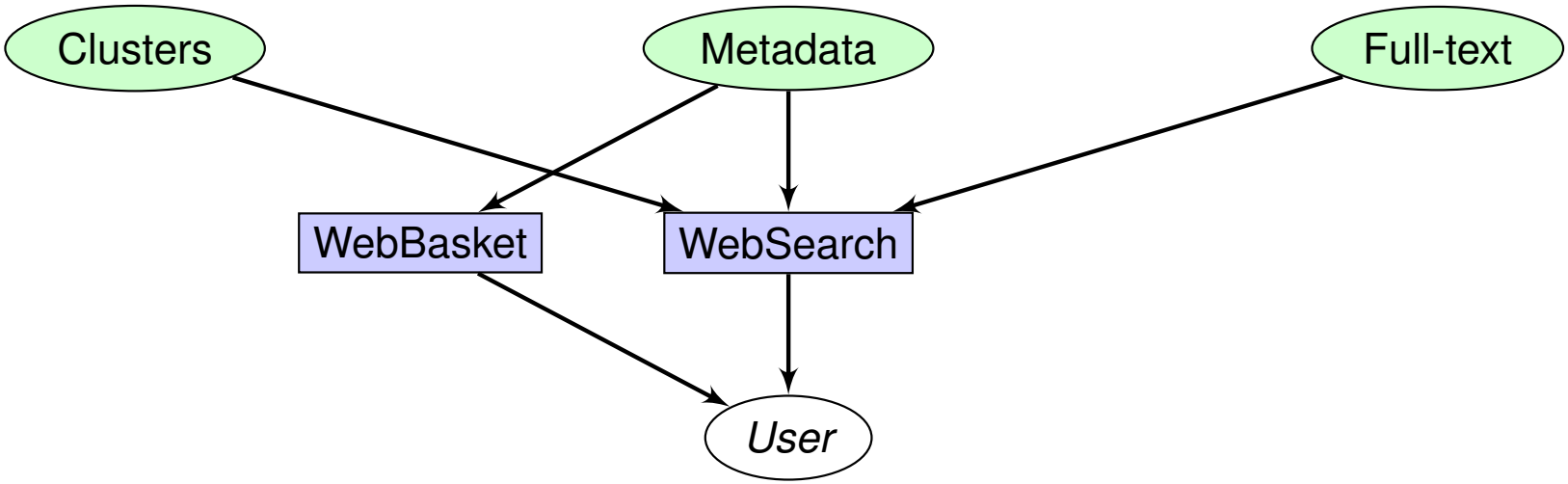
Invenio Modules: Dissemination



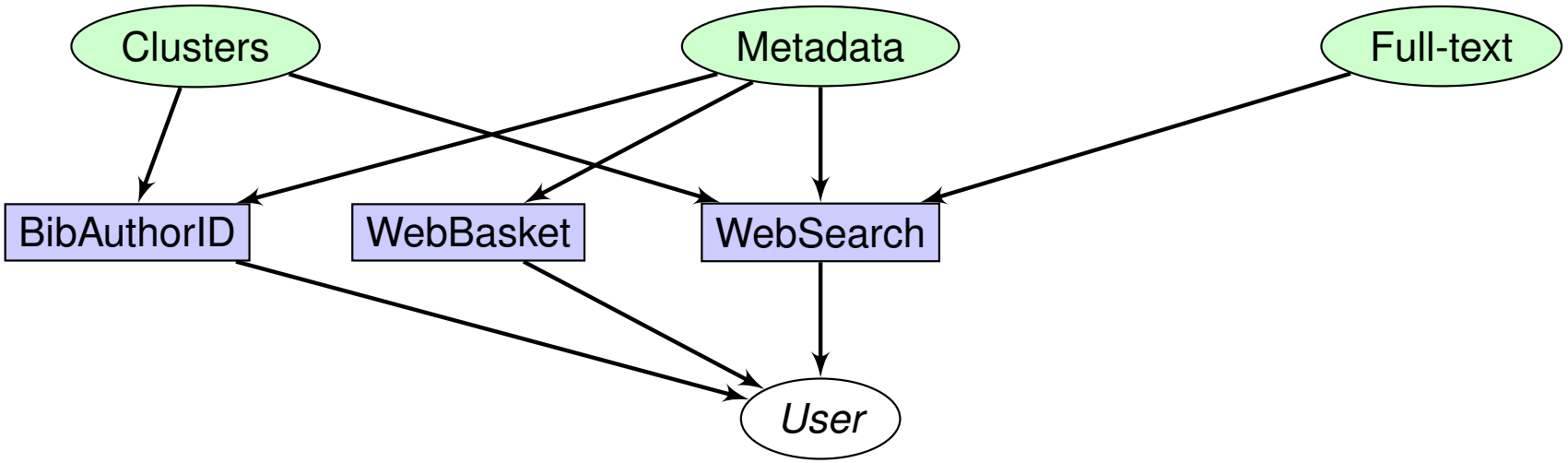
Invenio Modules: Dissemination



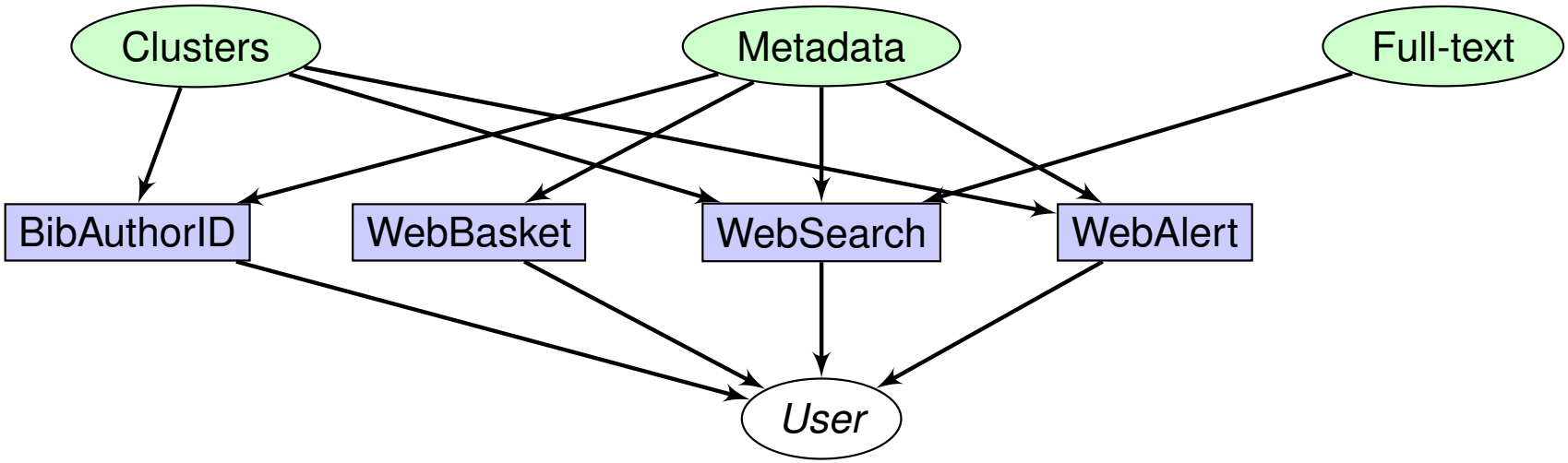
Invenio Modules: Dissemination



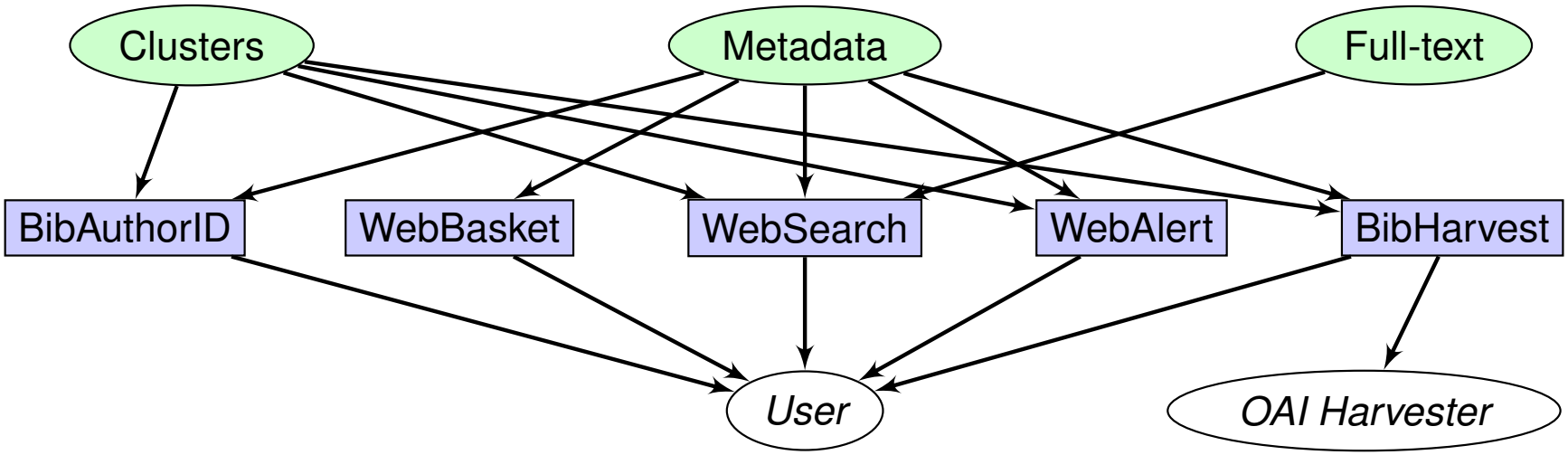
Invenio Modules: Dissemination



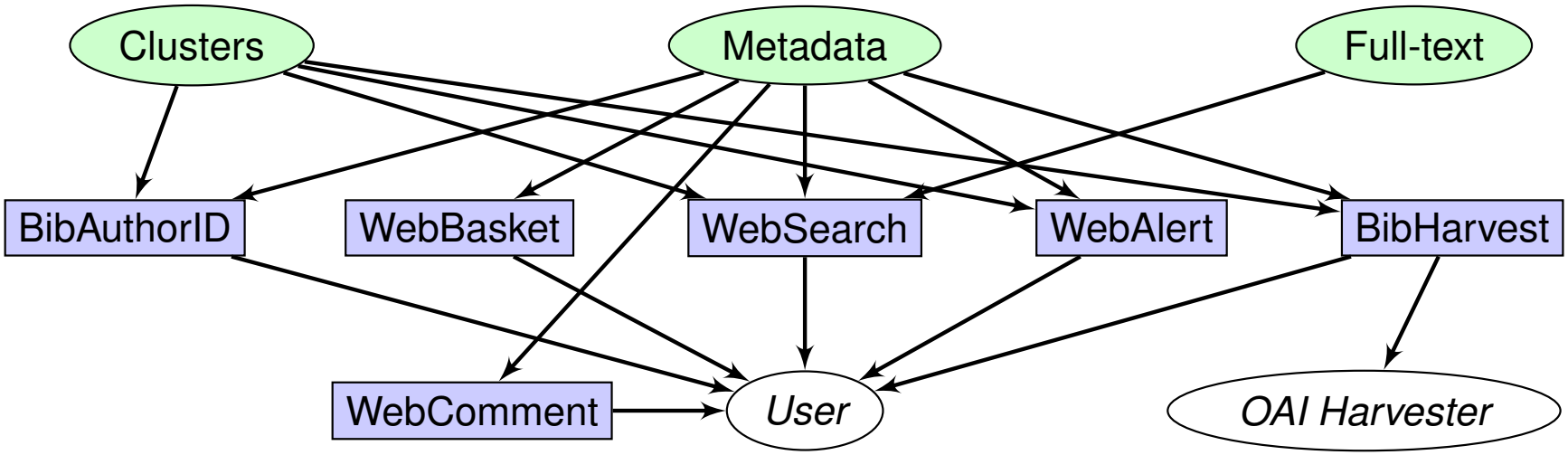
Invenio Modules: Dissemination



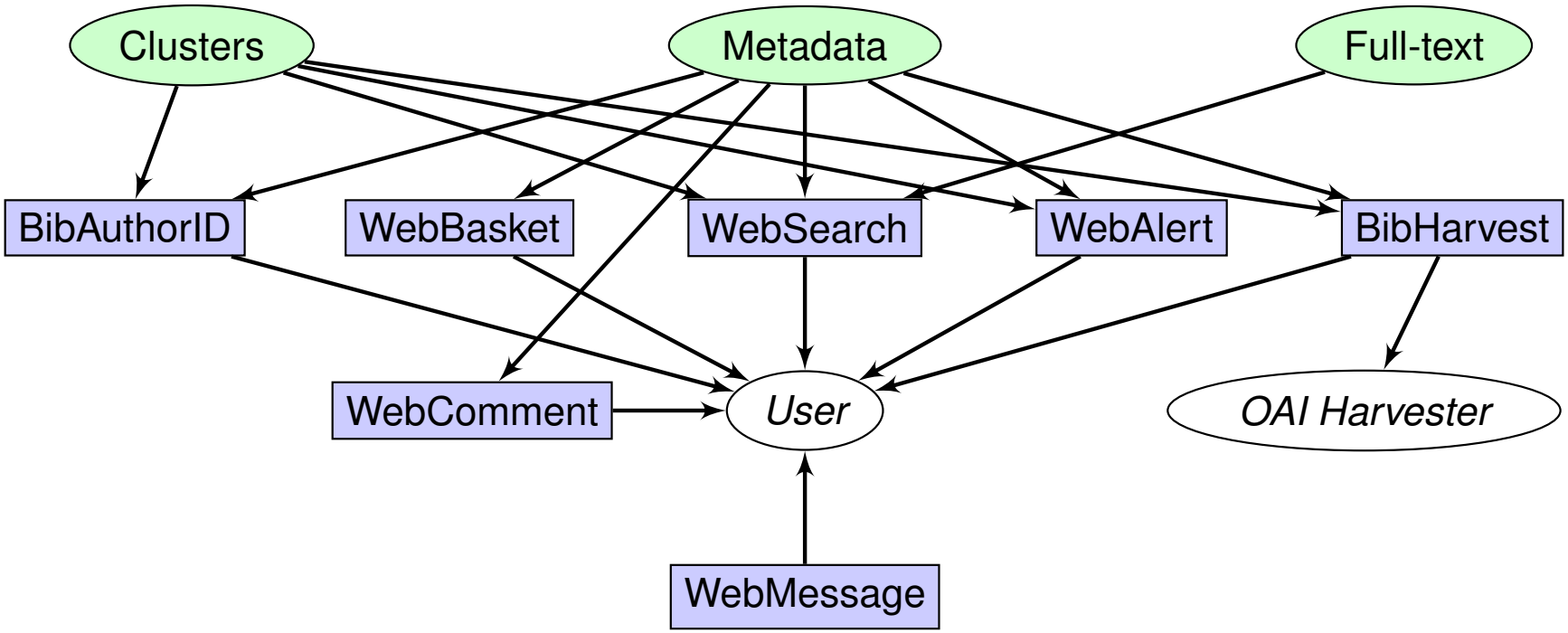
Invenio Modules: Dissemination



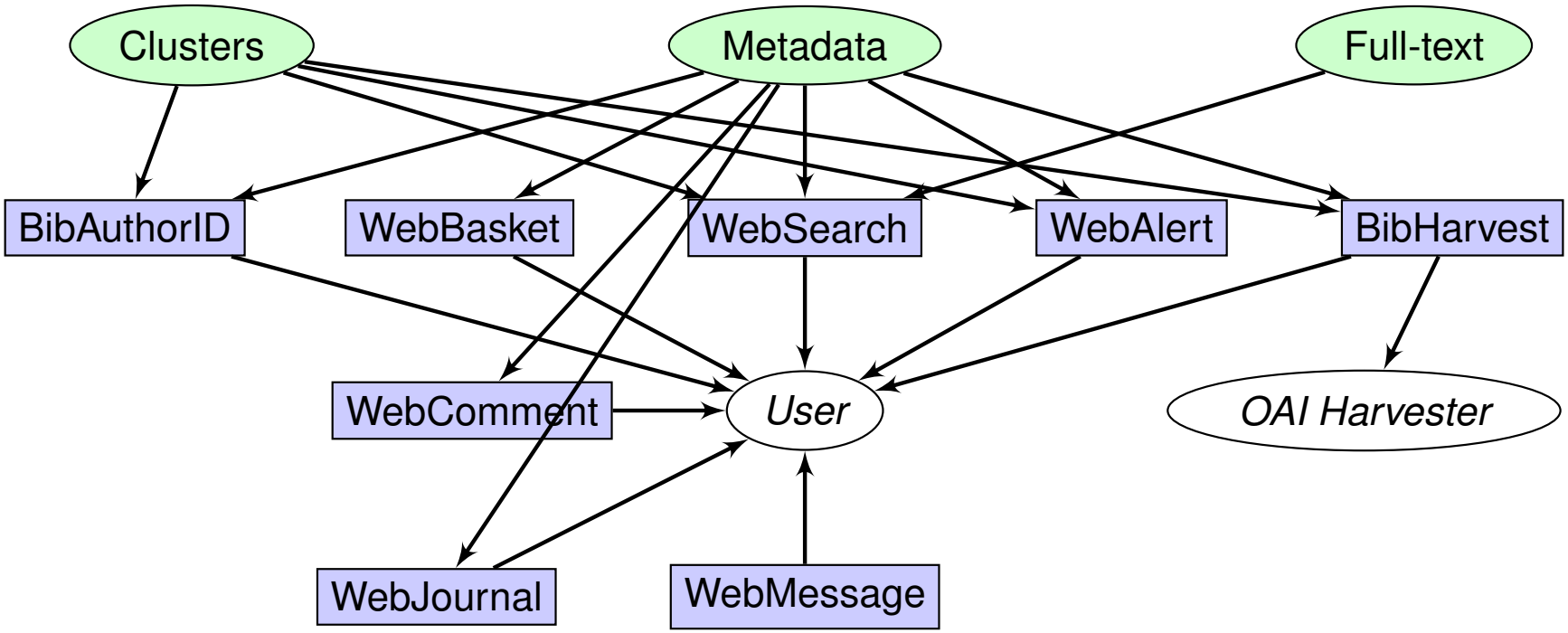
Invenio Modules: Dissemination



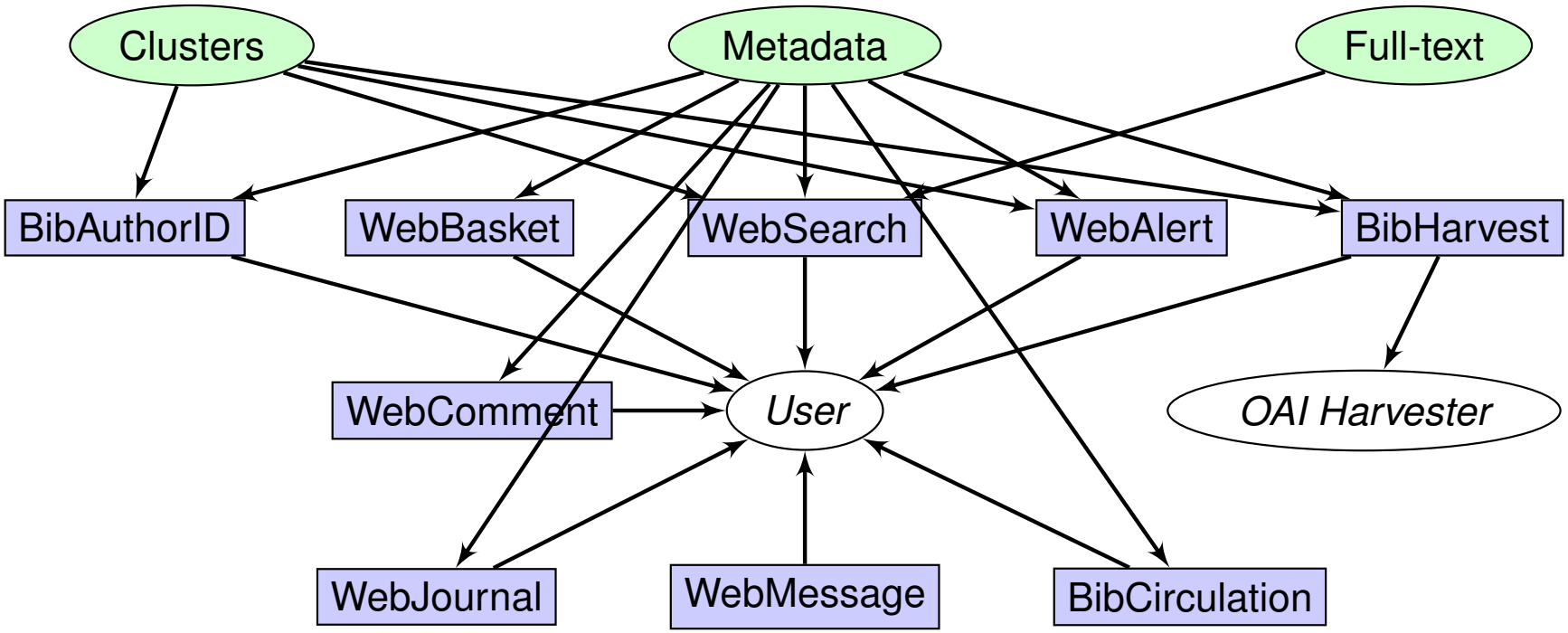
Invenio Modules: Dissemination



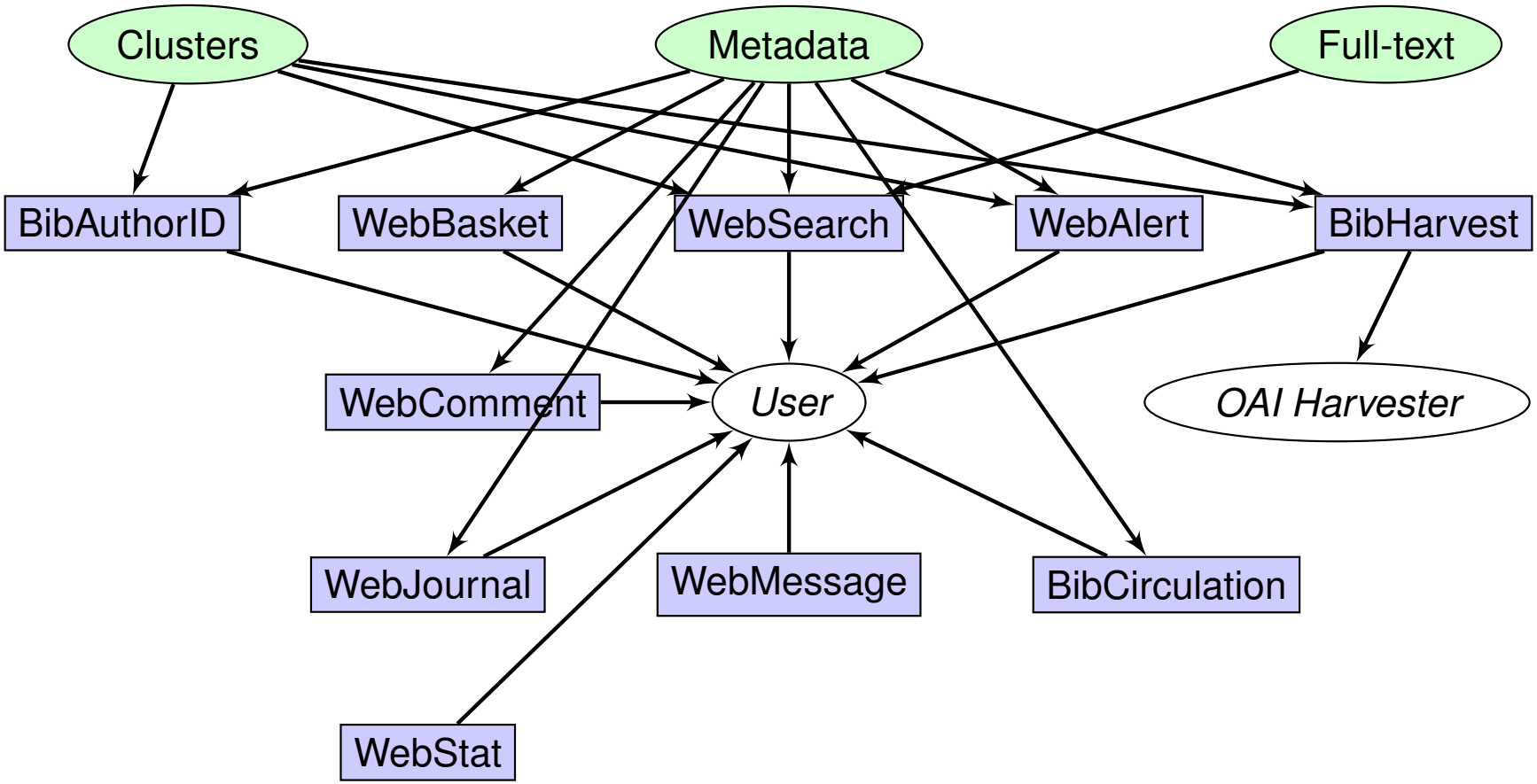
Invenio Modules: Dissemination



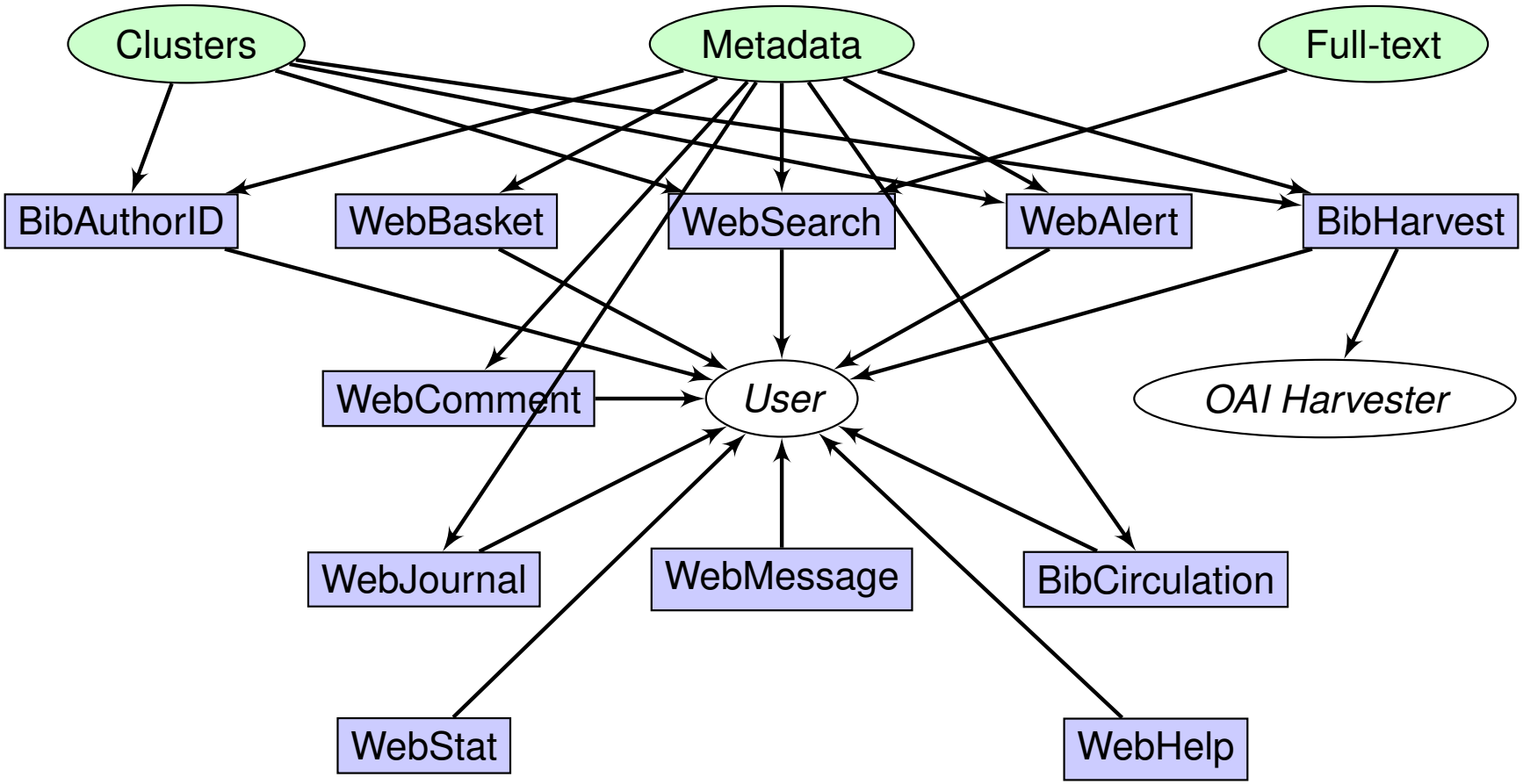
Invenio Modules: Dissemination



Invenio Modules: Dissemination



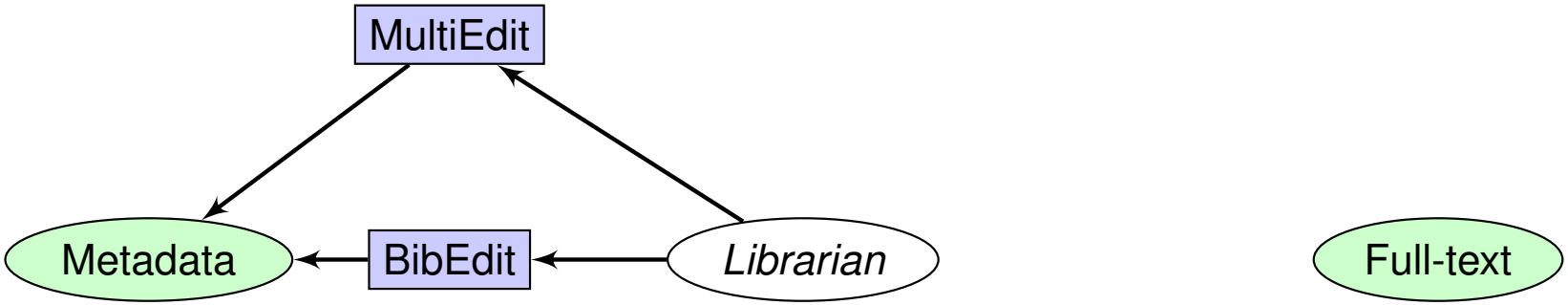
Invenio Modules: Dissemination



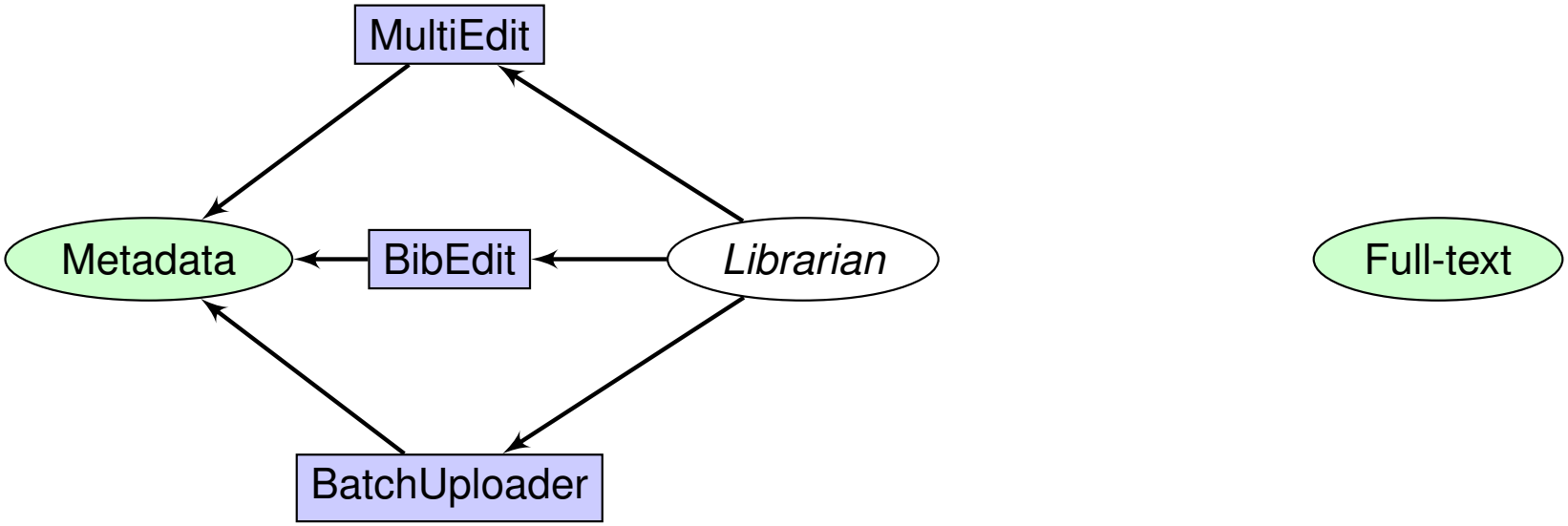
Invenio Modules: Curation

Invenio Modules: Curation

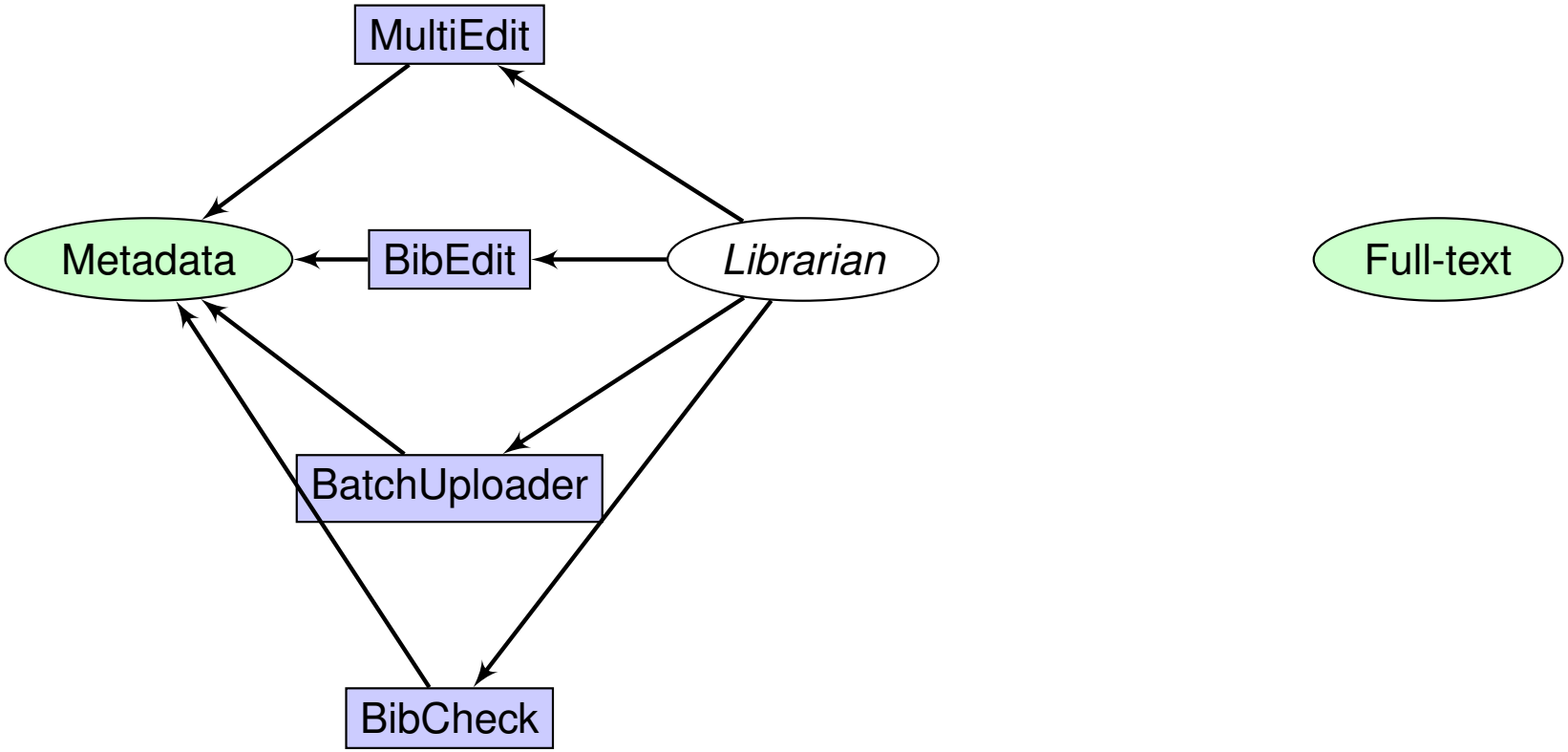
Invenio Modules: Curation



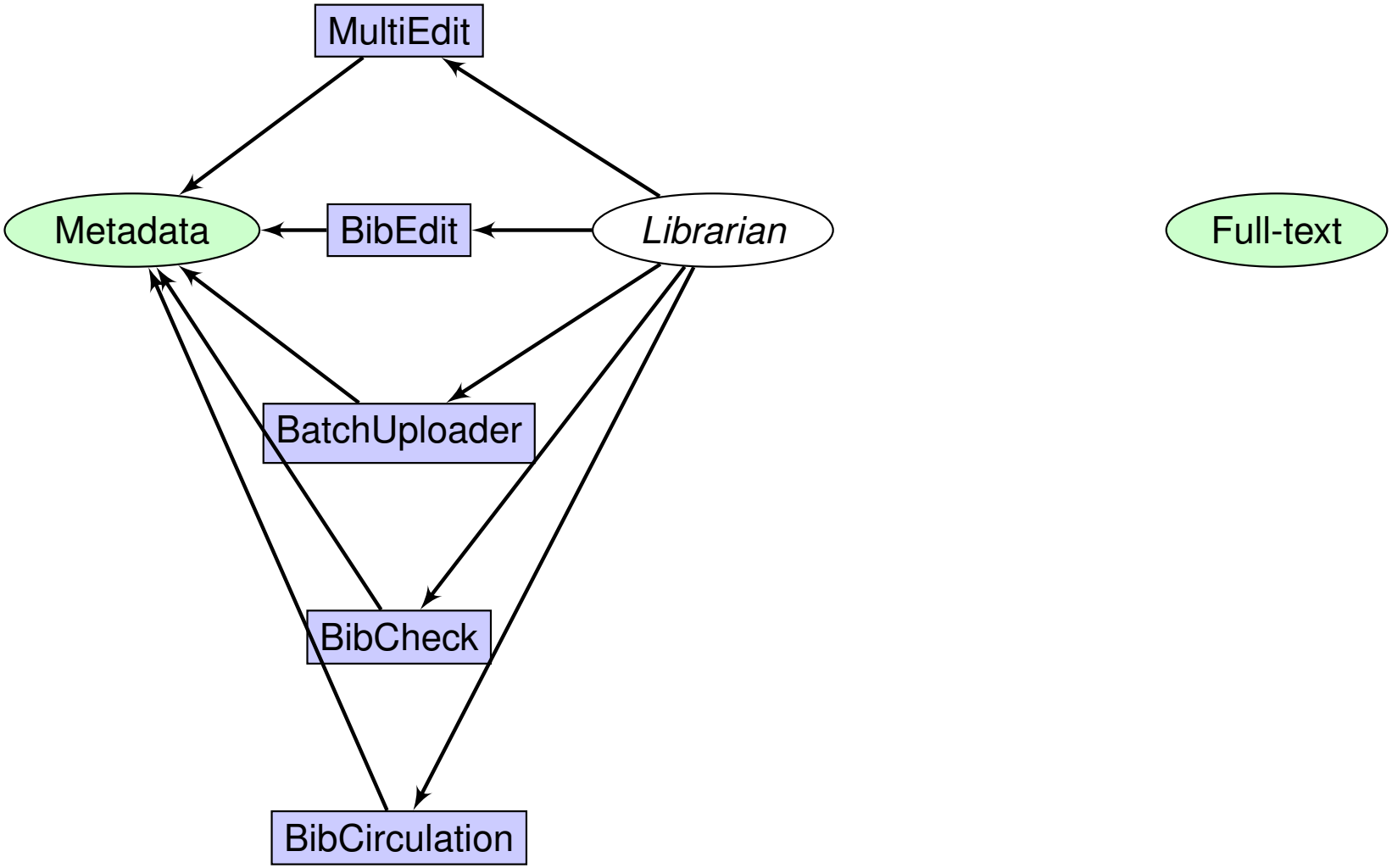
Invenio Modules: Curation



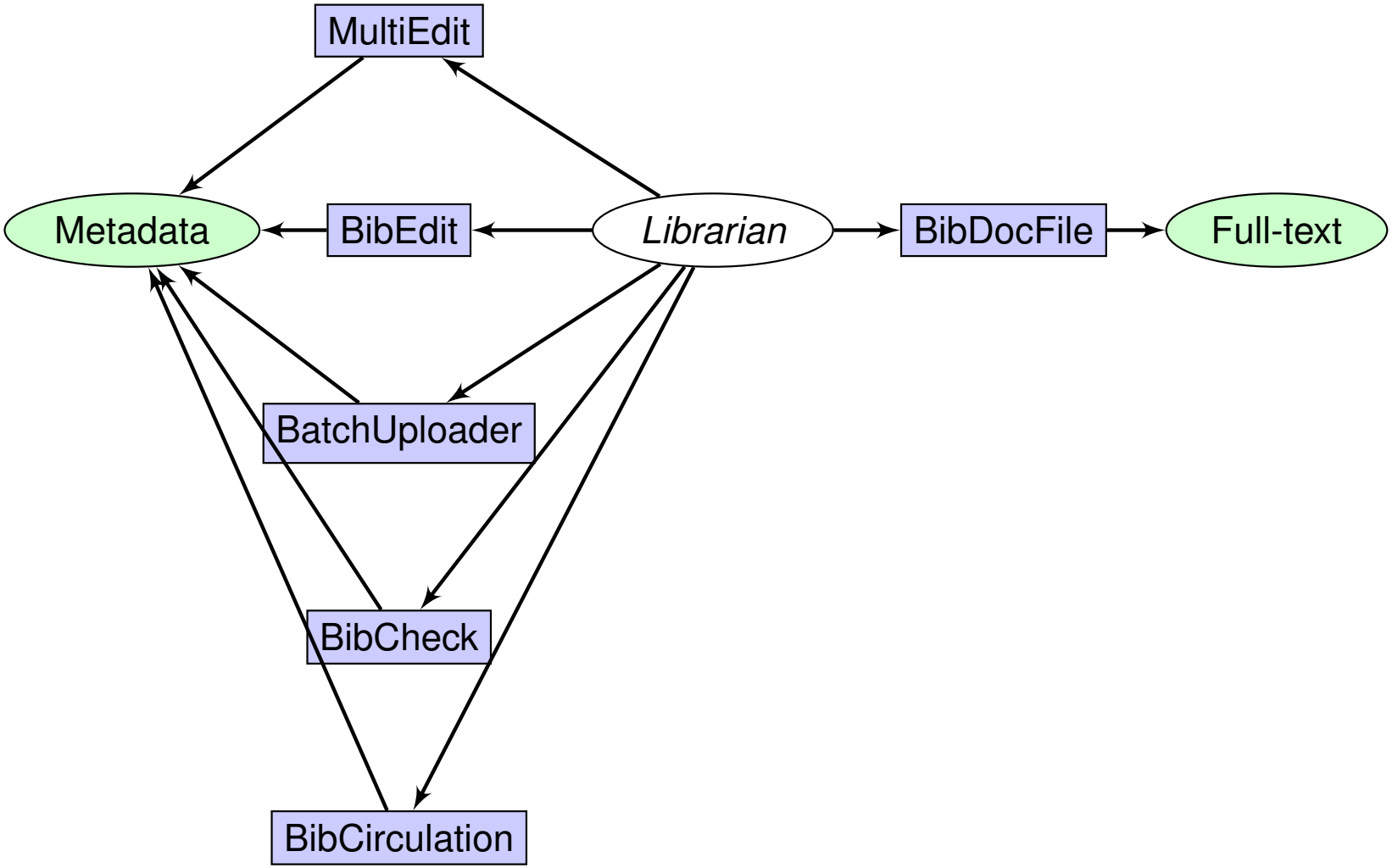
Invenio Modules: Curation



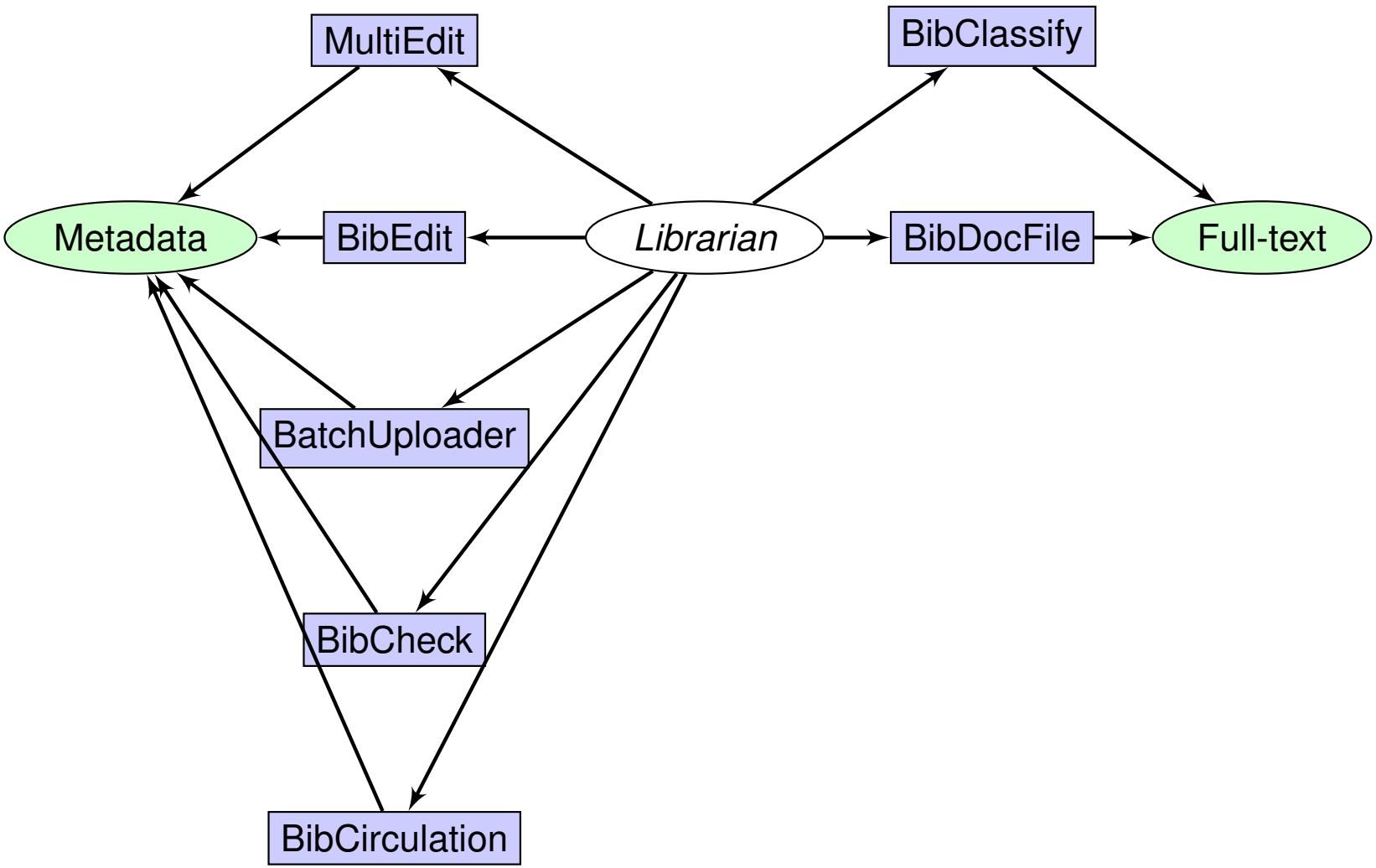
Invenio Modules: Curation



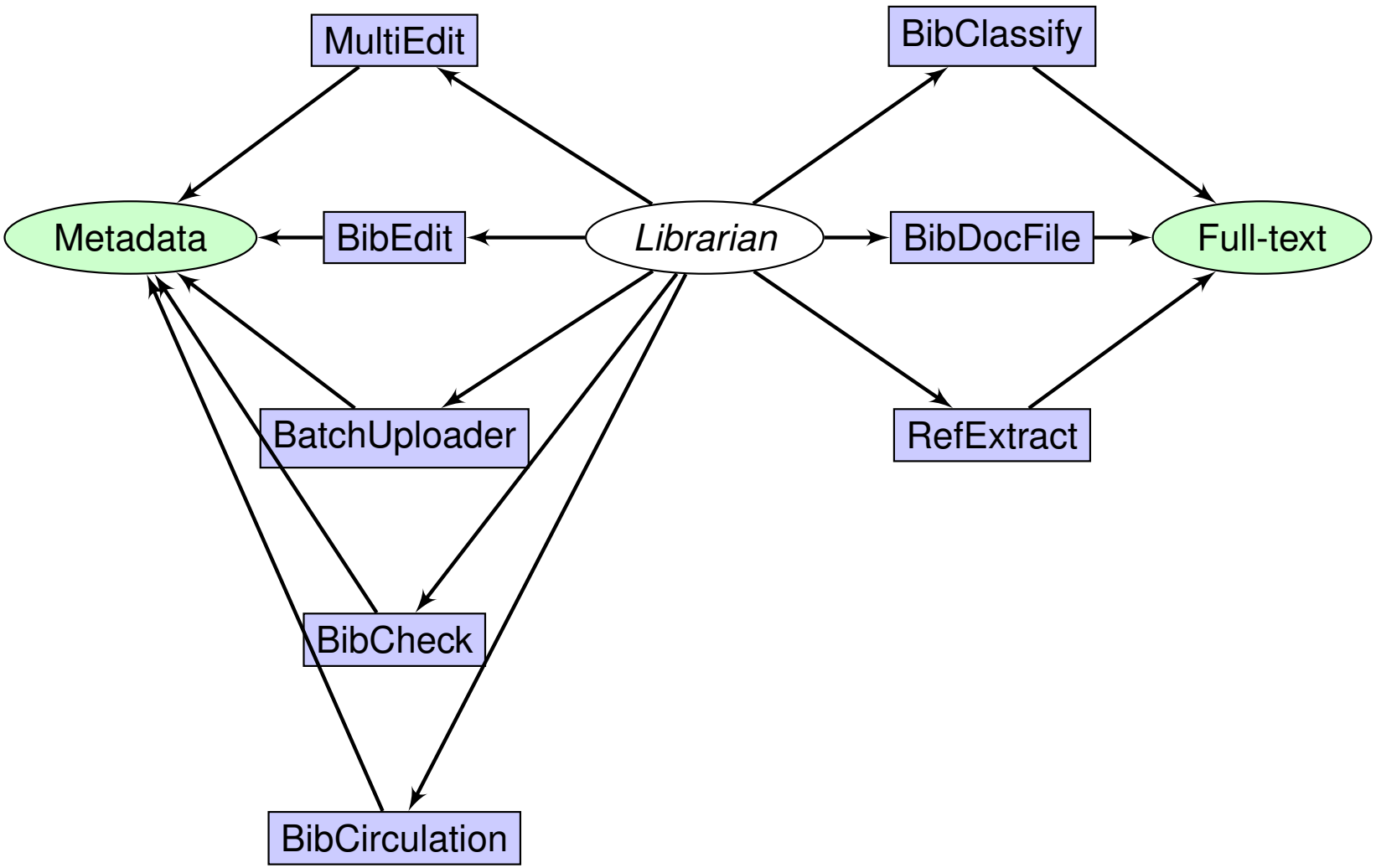
Invenio Modules: Curation



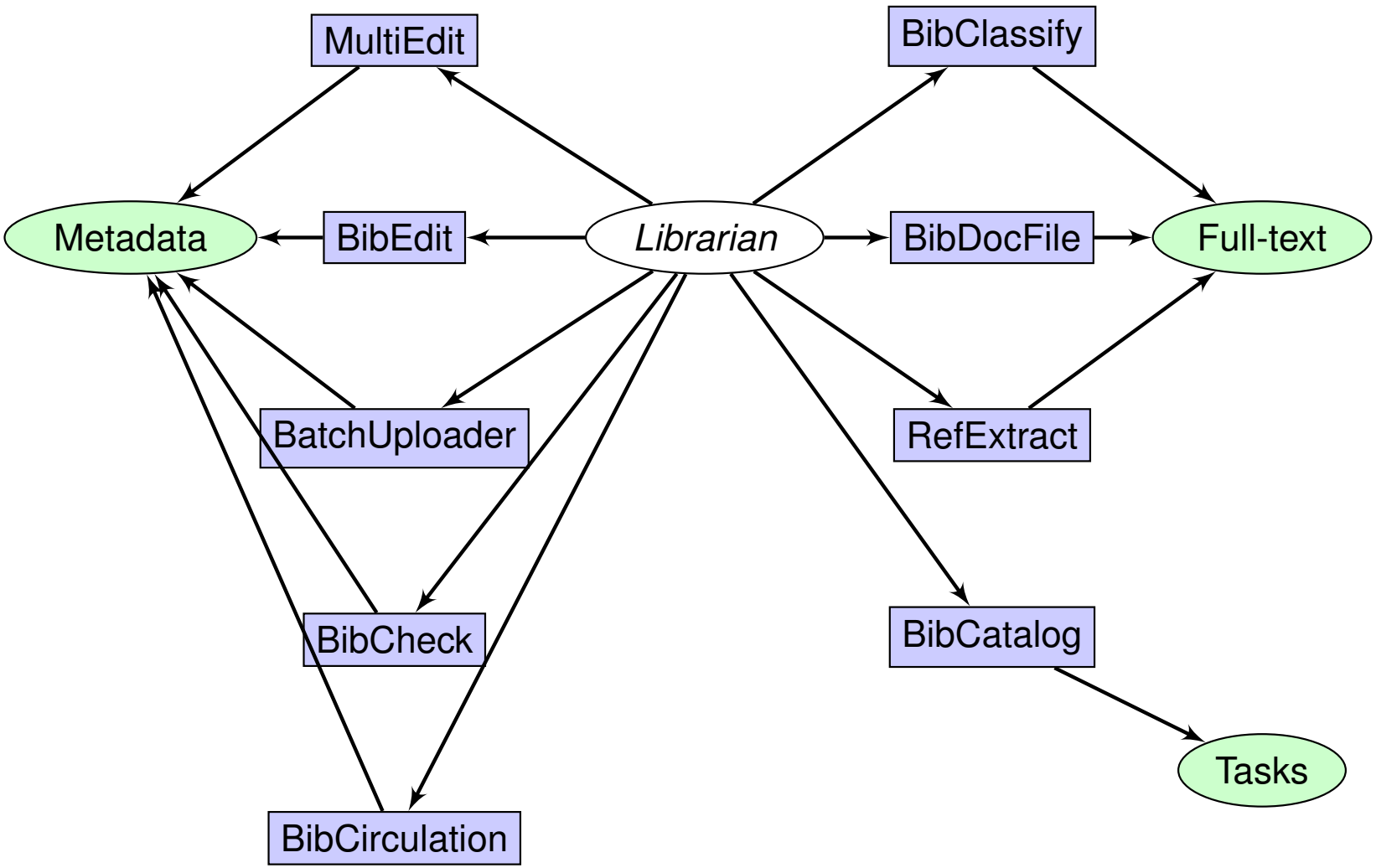
Invenio Modules: Curation



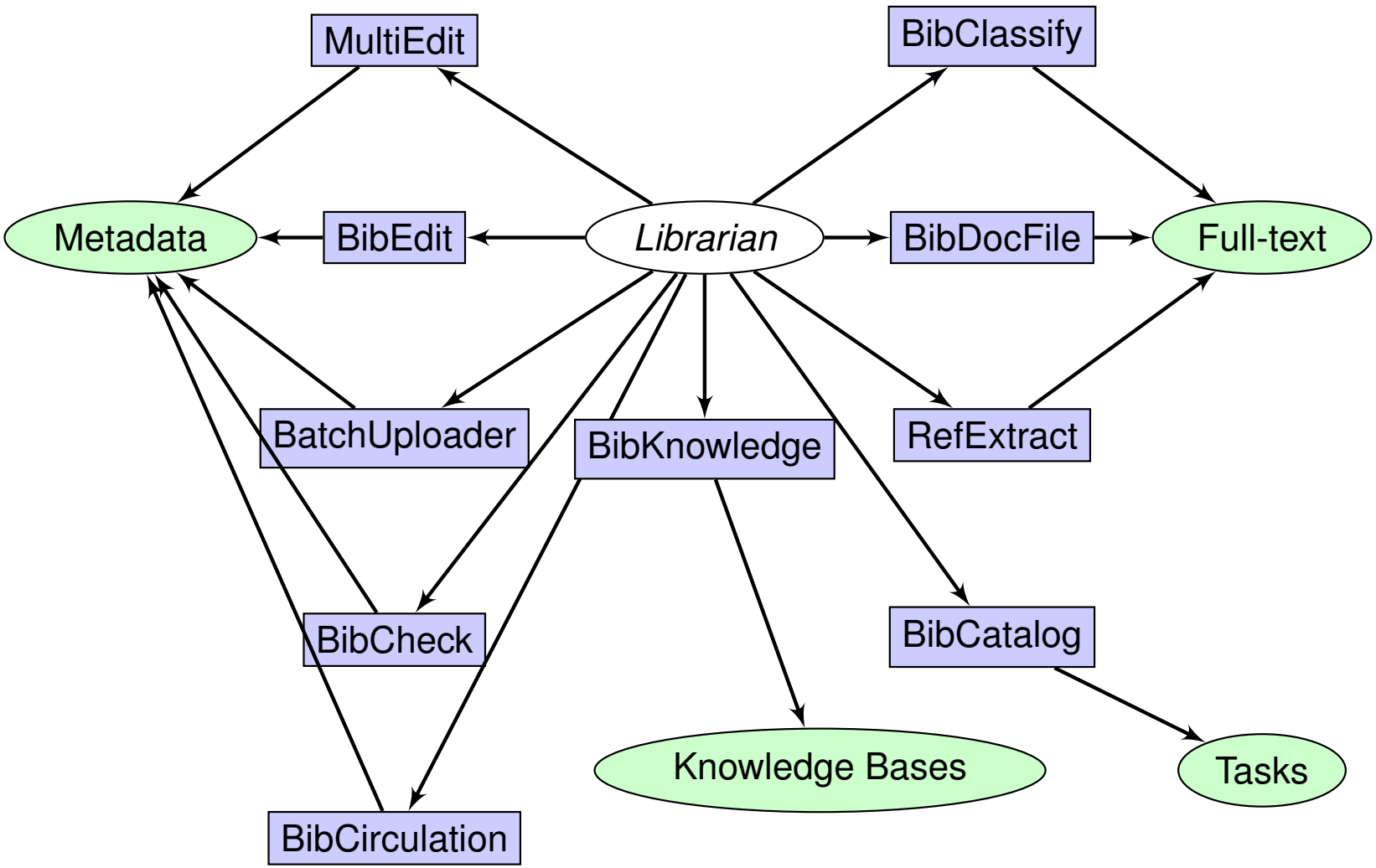
Invenio Modules: Curation



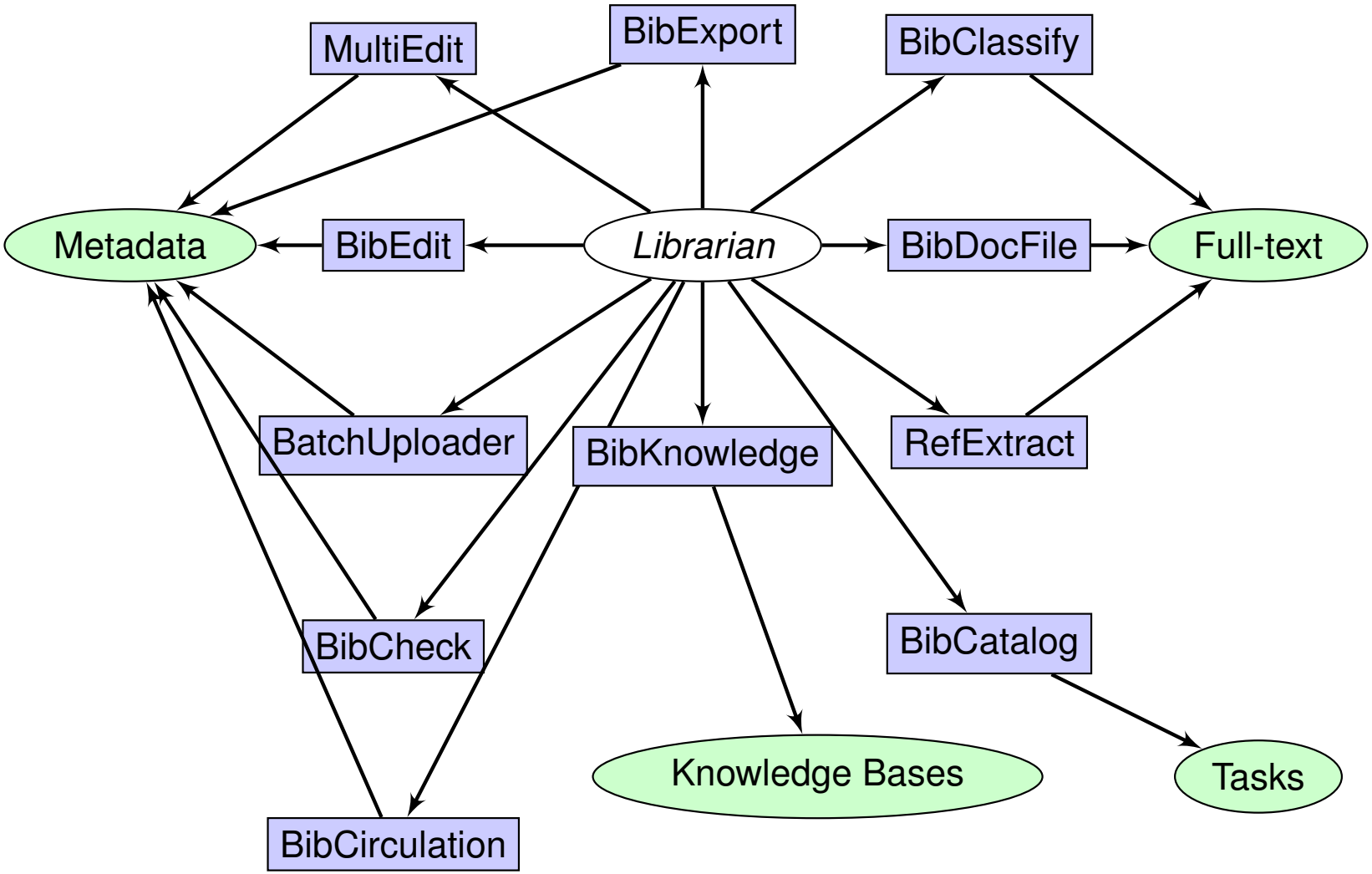
Invenio Modules: Curation



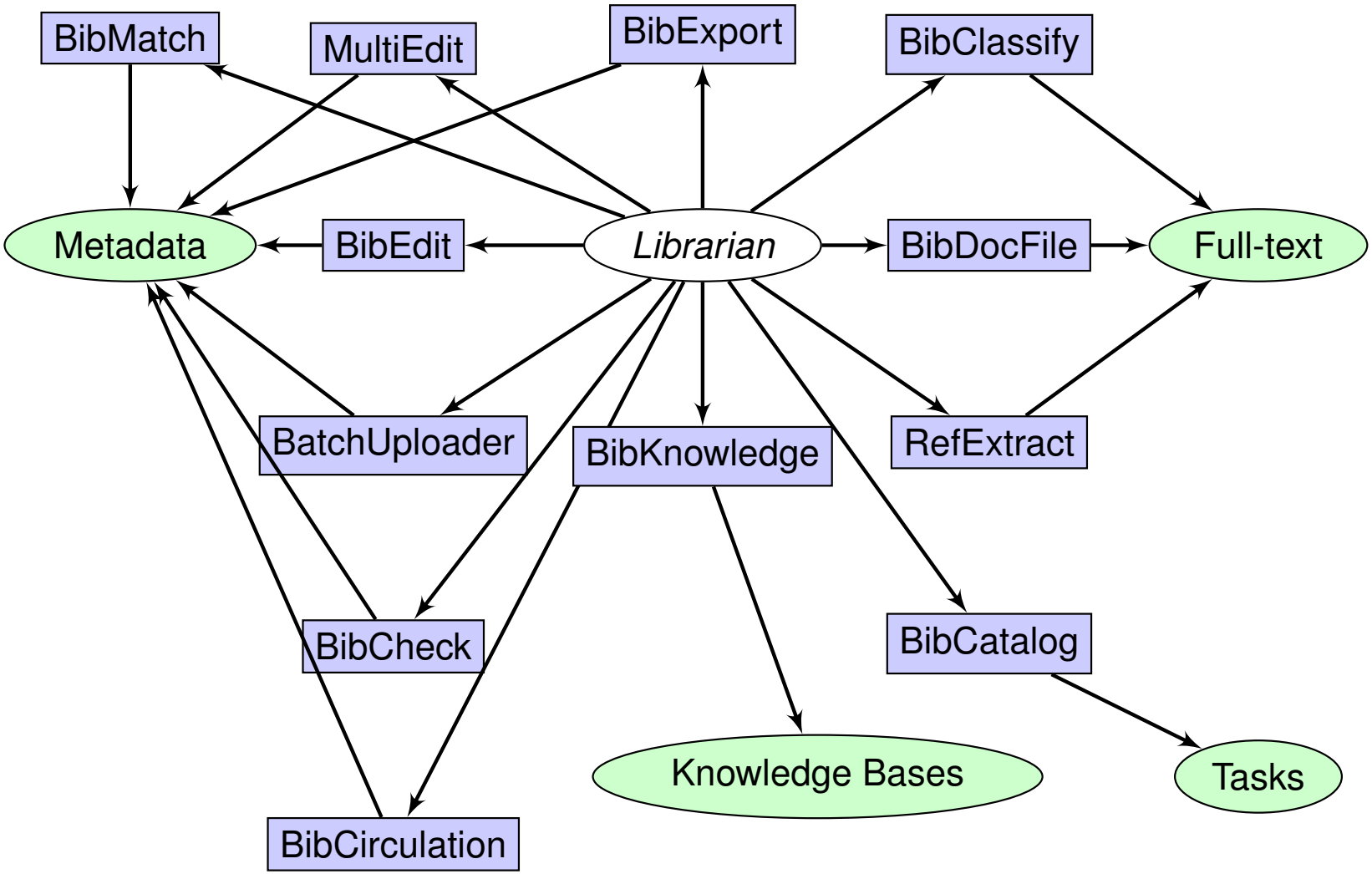
Invenio Modules: Curation



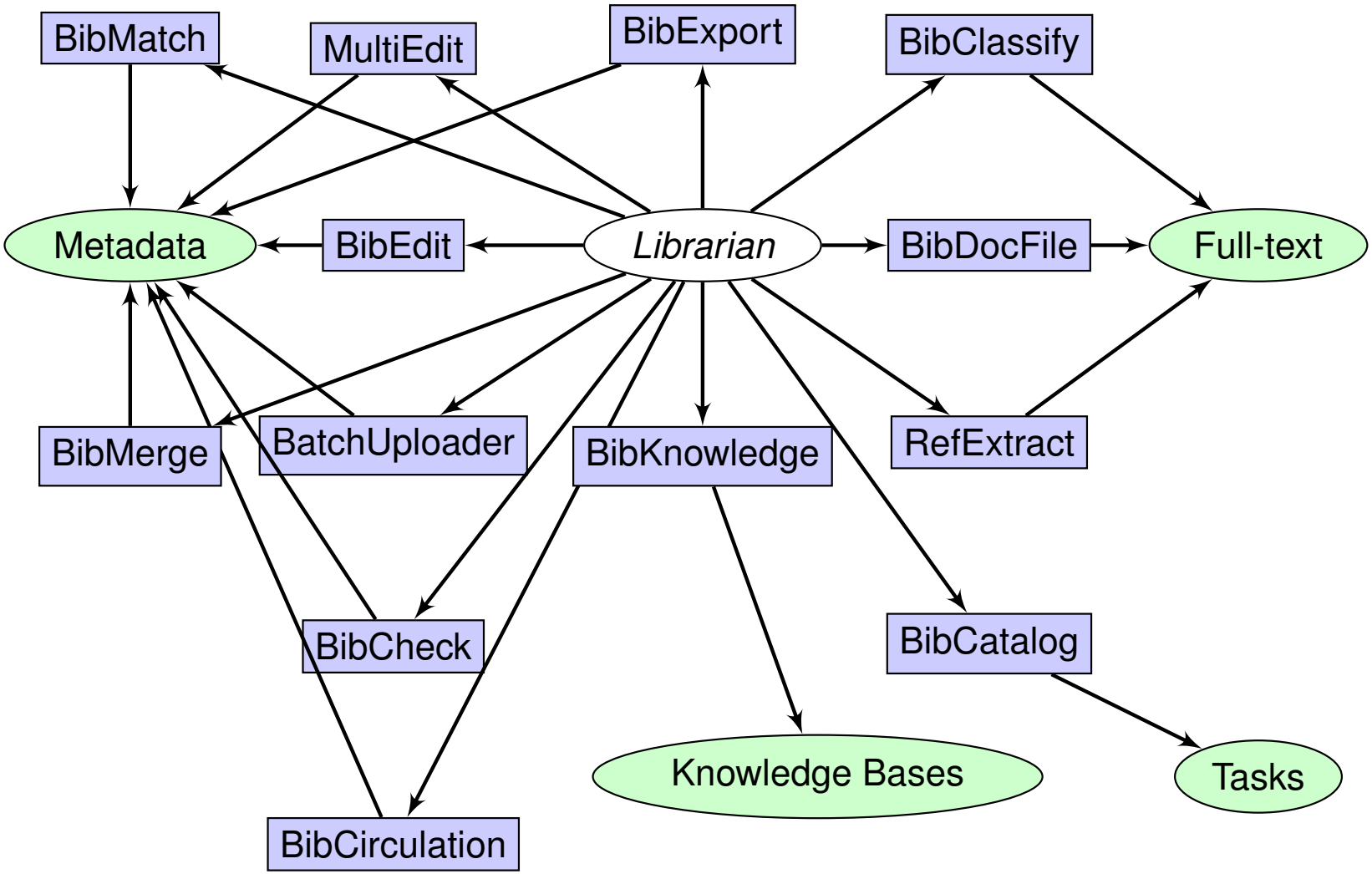
Invenio Modules: Curation



Invenio Modules: Curation



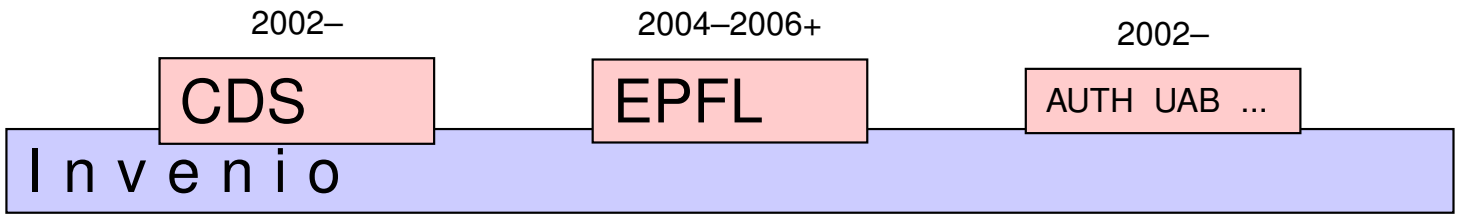
Invenio Modules: Curation



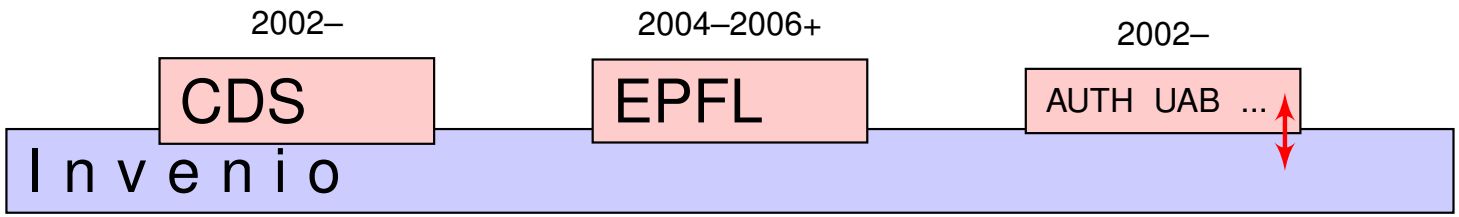
Invenio Modules: Summary

- ∼40 modules
- codebase
  - ∼350,000 lines of Python code
  - ∼15,000 lines of JavaScript code
  - ∼7,000 lines of XSL code
  - ∼8,000 lines of autotools code
- ∼120 authors and contributors since 2002
  - ∼48 authors and contributors in 2012 (18 new)
  - many short-term students, importance of *informal* coding standards
- ∼10 years of development
  - started at CERN, first release in 2002
  - now co-developed world-wide (EU, US)
- lego programming... but no silver bullet

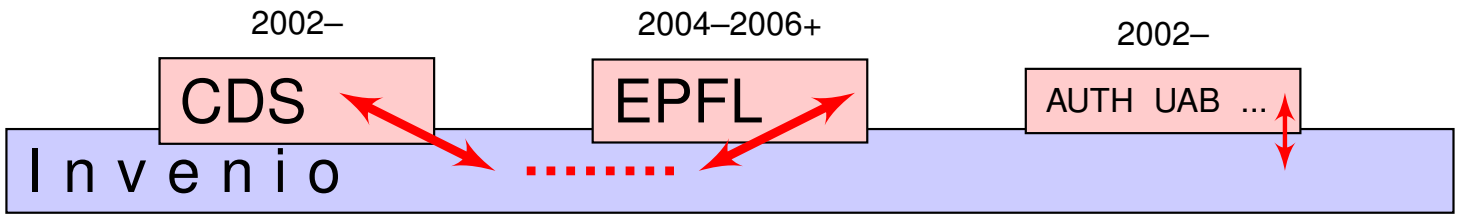
Developer Community



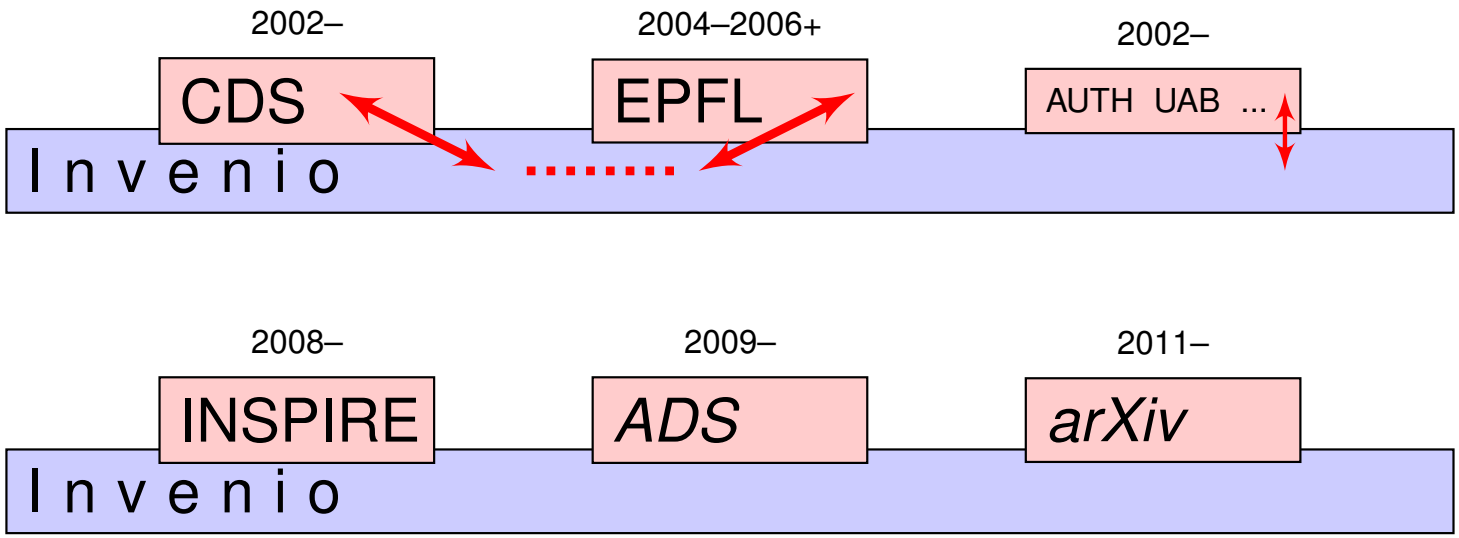
Developer Community



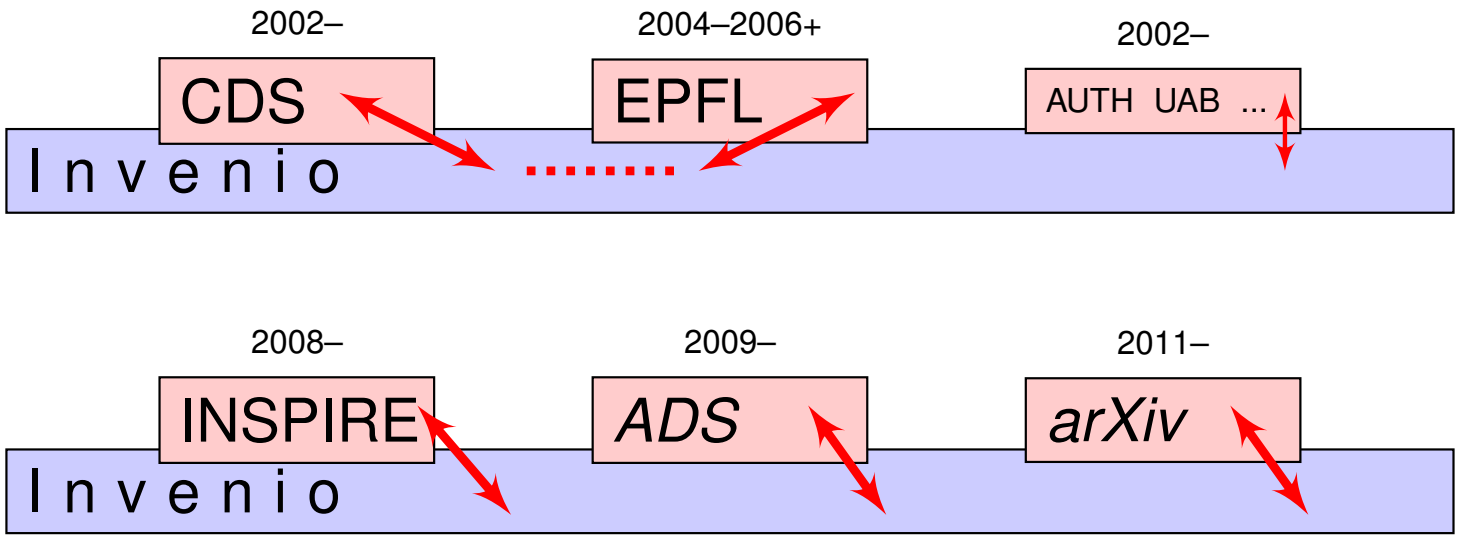
Developer Community



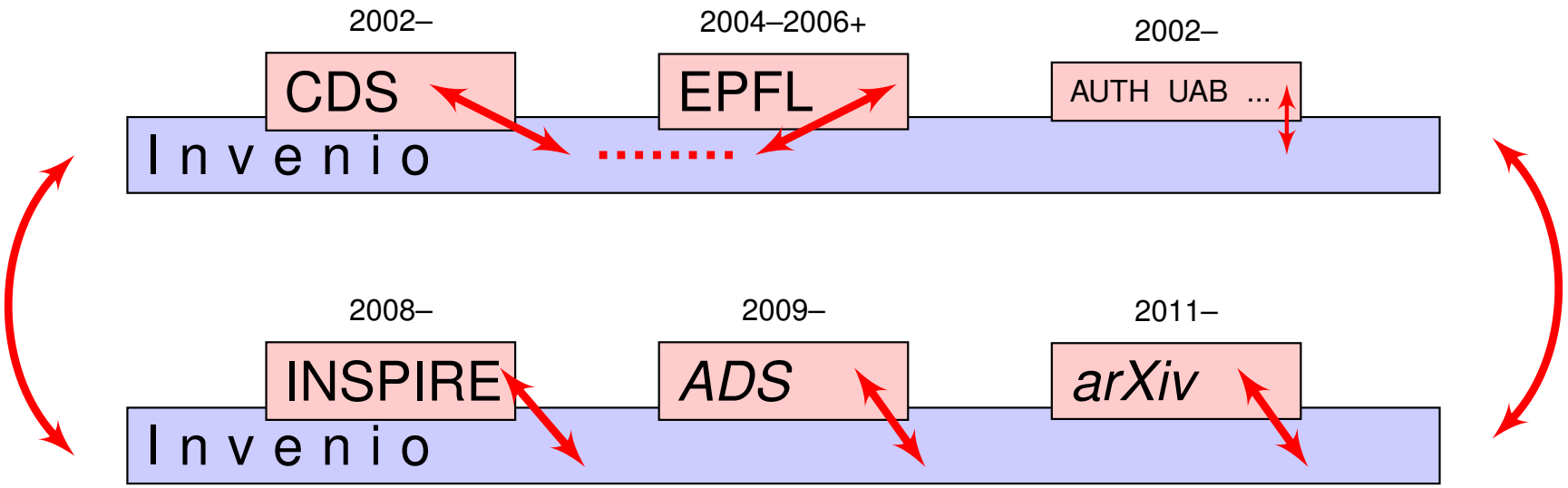
Developer Community



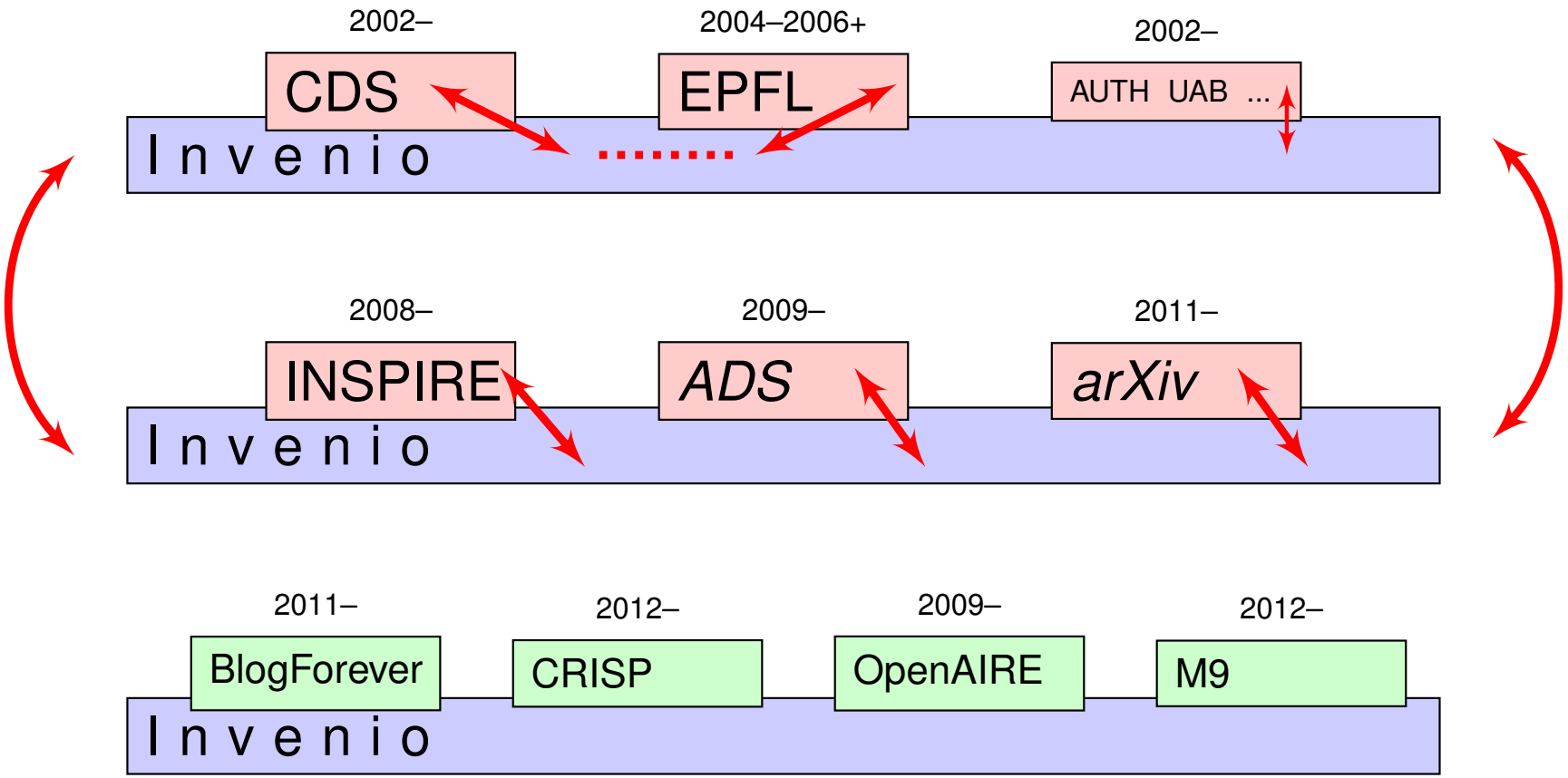
Developer Community



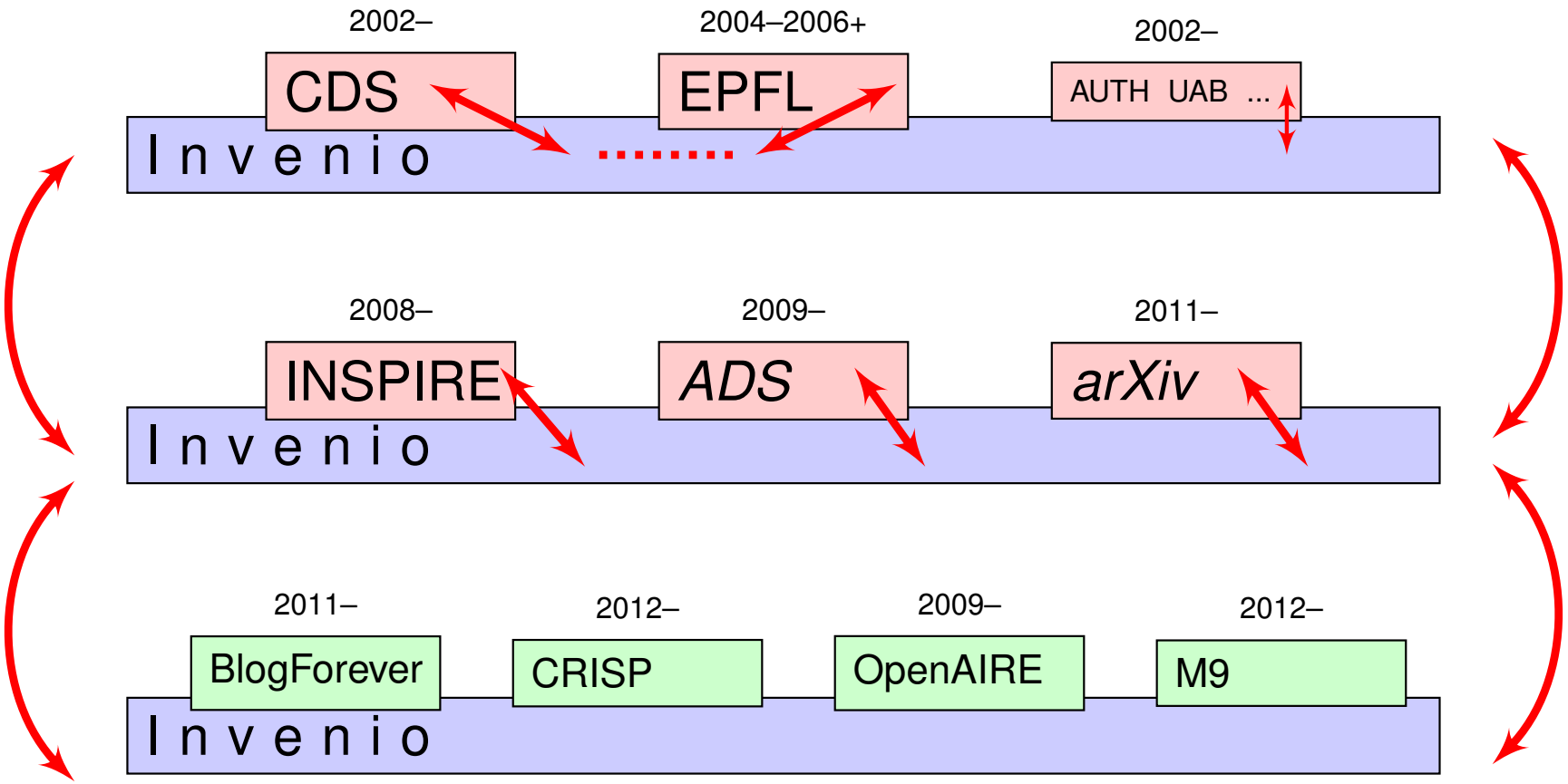
Developer Community



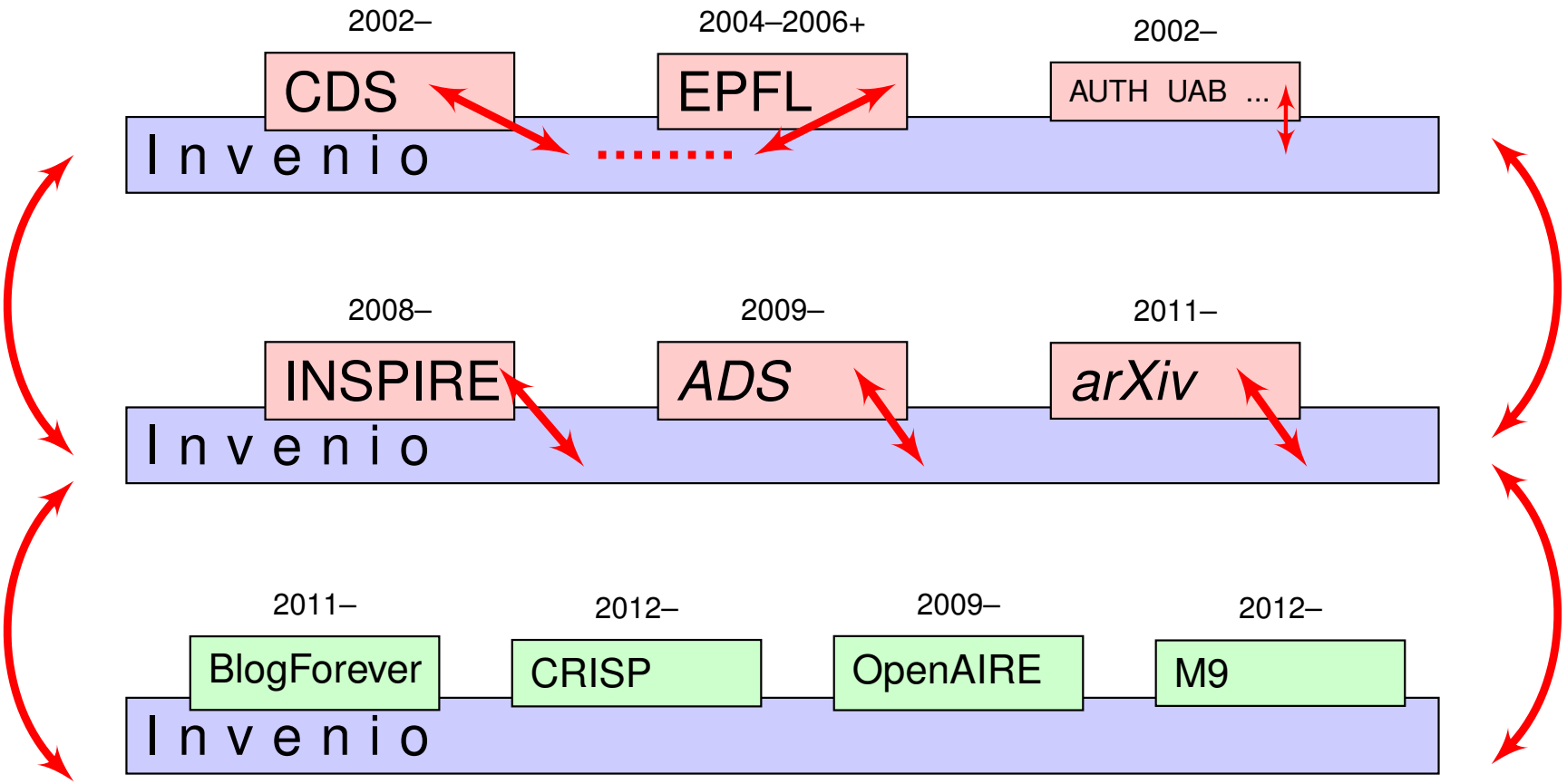
Developer Community



Developer Community



Developer Community



330k LOC - Invenio core sources

10k LOC - INSPIRE overlay sources

Outline

1 Introduction
- Digital Library
- Invenio

2 Case Studies
- Episode 1: Python
- Episode 2: Git
- Episode 3: Testing
- Episode 4: Building Efficient Indexes
- Episode 5: NIH
- Episode 6: Scalability

3 Conclusions

Outline

1 Introduction
- Digital Library
- Invenio

2 Case Studies
- Episode 1: Python
- Episode 2: Git
- Episode 3: Testing
- Episode 4: Building Efficient Indexes
- Episode 5: NIH
- Episode 6: Scalability

3 Conclusions

Why Python?

- easy to read and understand
  (good for many temporary developers)
- suitable for rapid prototyping
  (good for organic-growth software development model)
- *write code to throw it away*

Art of Ikebana

- Ikebana, “giving life to flowers”
- Japanese art of flower arrangement, “way of flowers”
- *“disciplined art form in which nature and humanity are brought together”*
- natural shapes, graceful lines
- minimalism

Art of Ikebana Programming

- example of anonymous functions

Java?

```
new Callable() {
    public Object call(Object x) {
        return x.times(k)
    }
}
```

Python!

```
lambda x: k * x
```

Art of Ikebana Programming

- example of anonymous functions

Java?

```
new Callable() {
    public Object call(Object x) {
        return x.times(k)
    }
}
```

Python!

```
lambda x: k * x
```

Speeding Up Python

- bytecode interpreted language: what about speed?
- **Cython** permits to write C extensions easily
- combining efficiency of C with high-levelness of Python

Example: intbitset.pyx

```
ctypedef unsigned long long int word_t

ctypedef struct IntBitSet:
    int size
    int allocated
    word_t trailing_bits
    int tot
    word_t *bitset
```

Outline

1 Introduction
- Digital Library
- Invenio

2 Case Studies
- Episode 1: Python
- Episode 2: Git
- Episode 3: Testing
- Episode 4: Building Efficient Indexes
- Episode 5: NIH
- Episode 6: Scalability

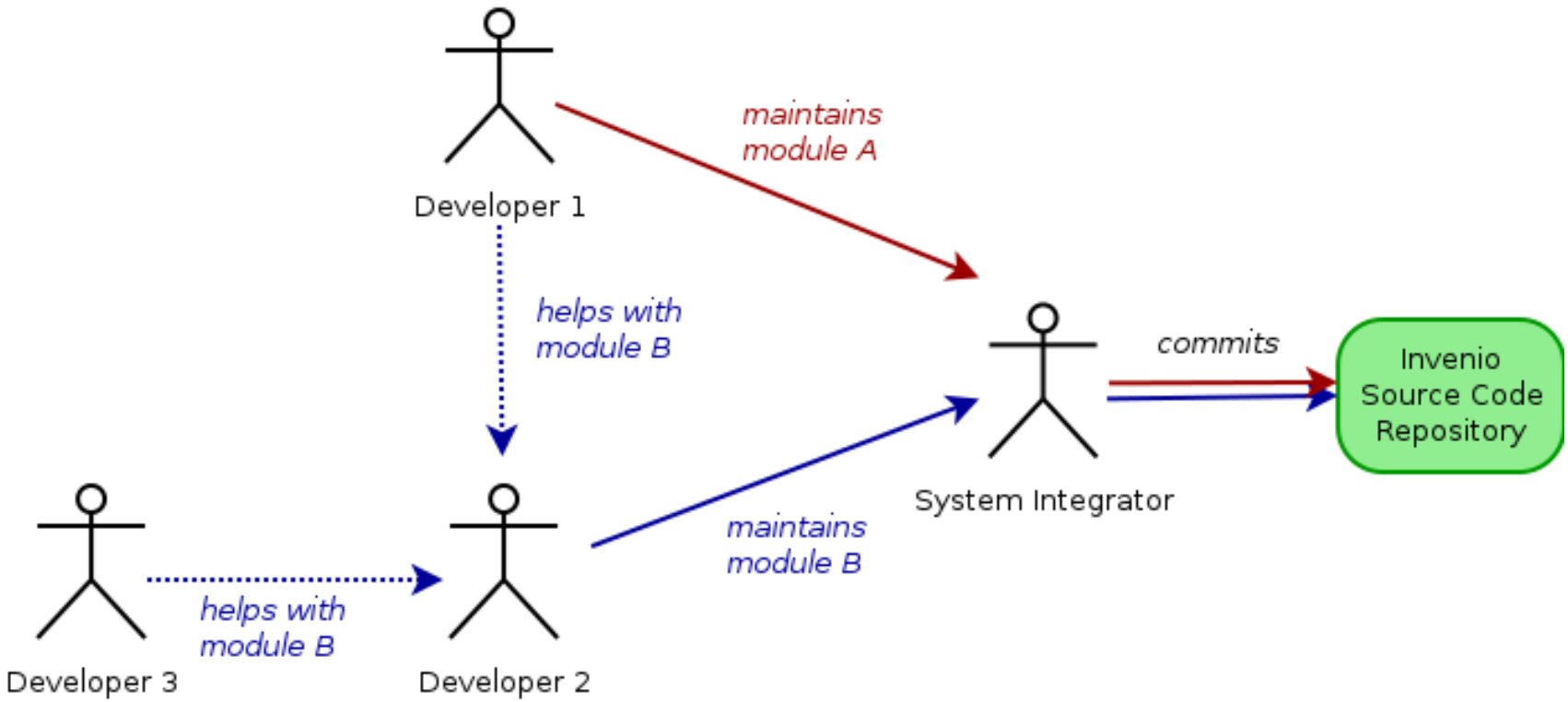
3 Conclusions

Why Git?

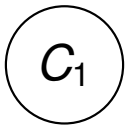
- good for distributed teams
- good for offline development
- powerful branching/merging, first class citizenship
- commit early, commit often
  (to private repositories)
- rebase and clean when ready
  (before pushing for public consumption)
- using pull-on-demand collaboration model
  (as opposed to shared-push collaboration model)

Git Collaboration

- **pull-on-demand** collaboration model
- inherent code review and QA processes before integration
- modules maintainers aka “integration lieutenants”

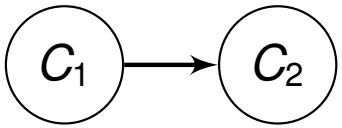


Git Branches



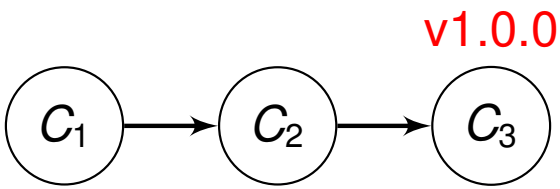
master

Git Branches



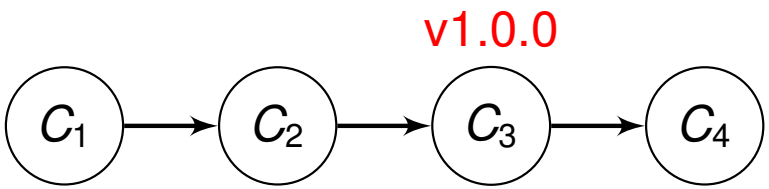
master

Git Branches

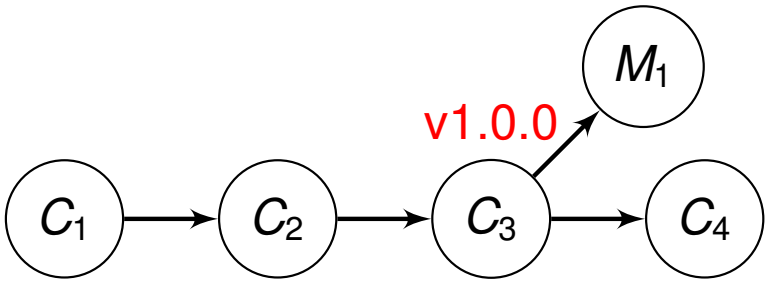


master

Git Branches



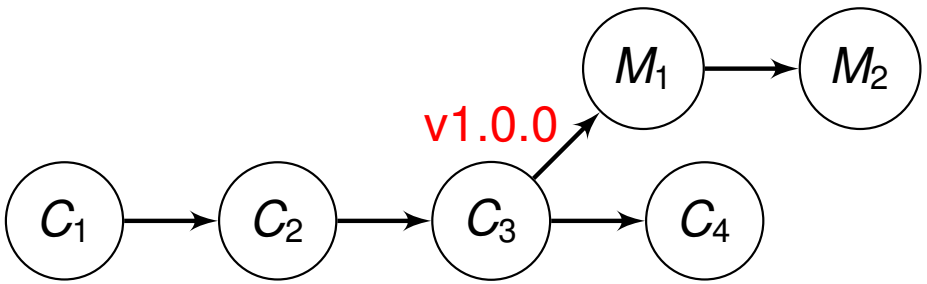
Git Branches



maint-1.0

master

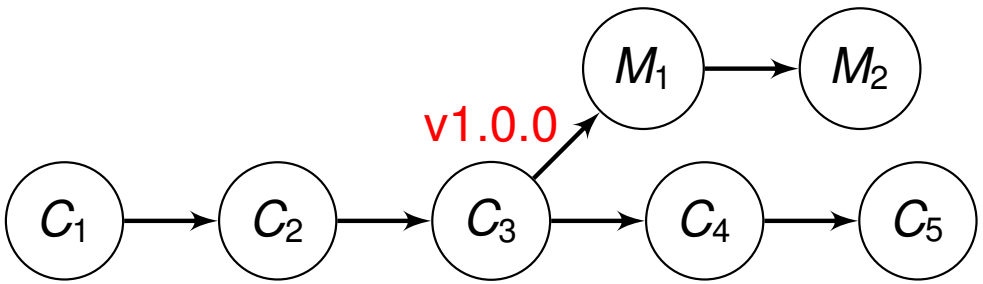
Git Branches



maint-1.0

master

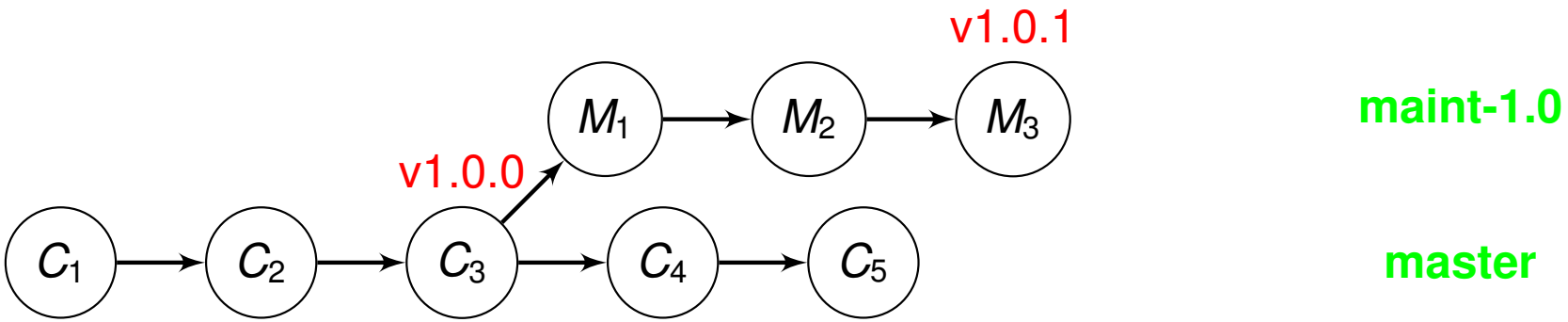
Git Branches



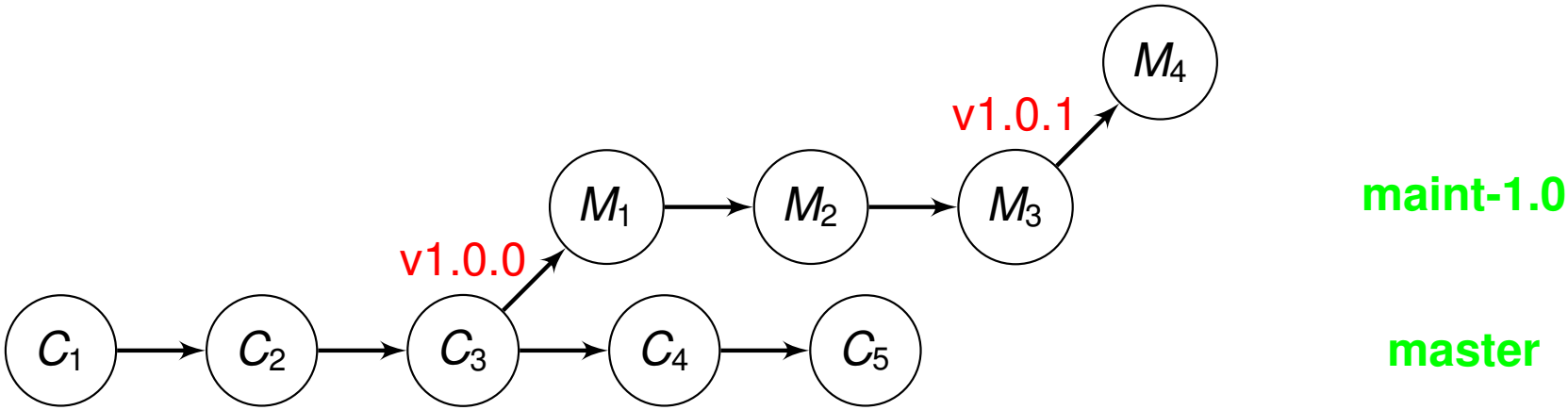
maint-1.0

master

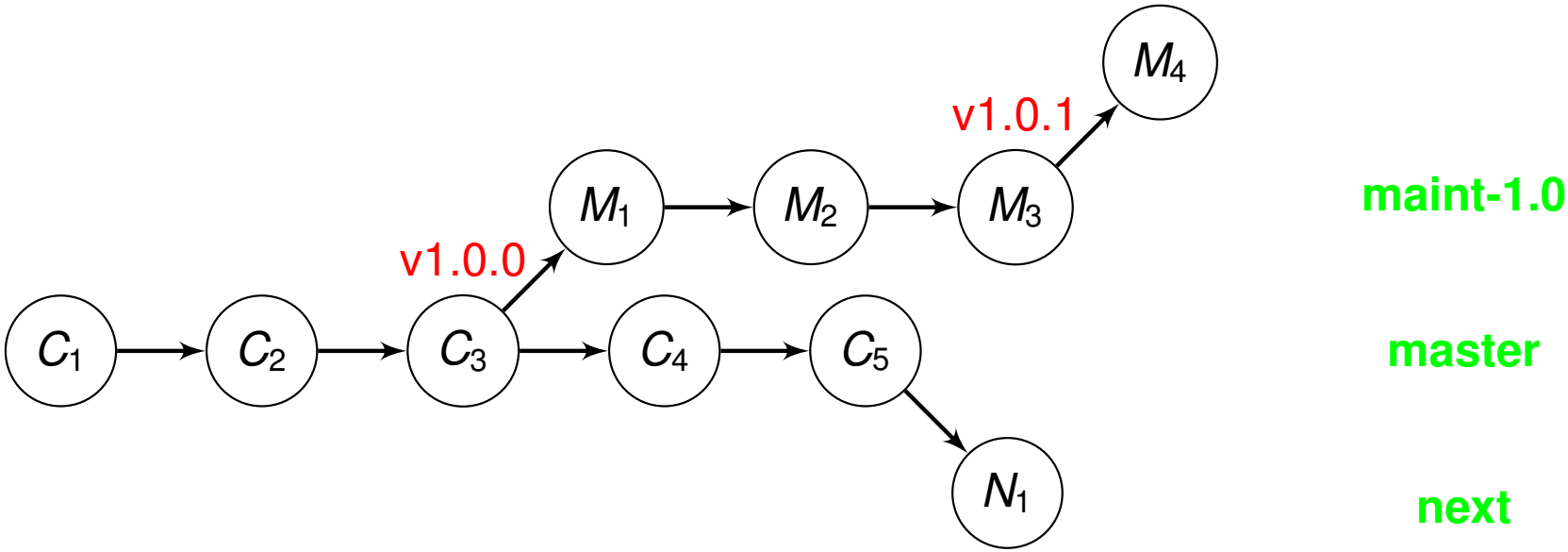
Git Branches



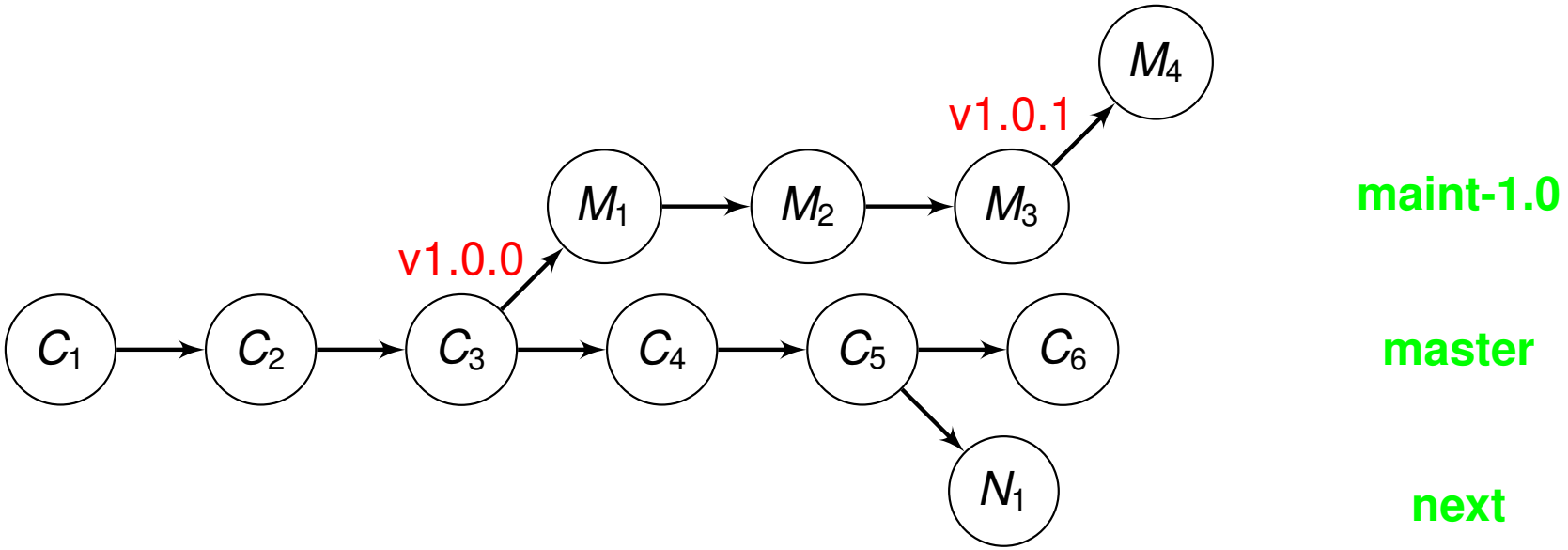
Git Branches



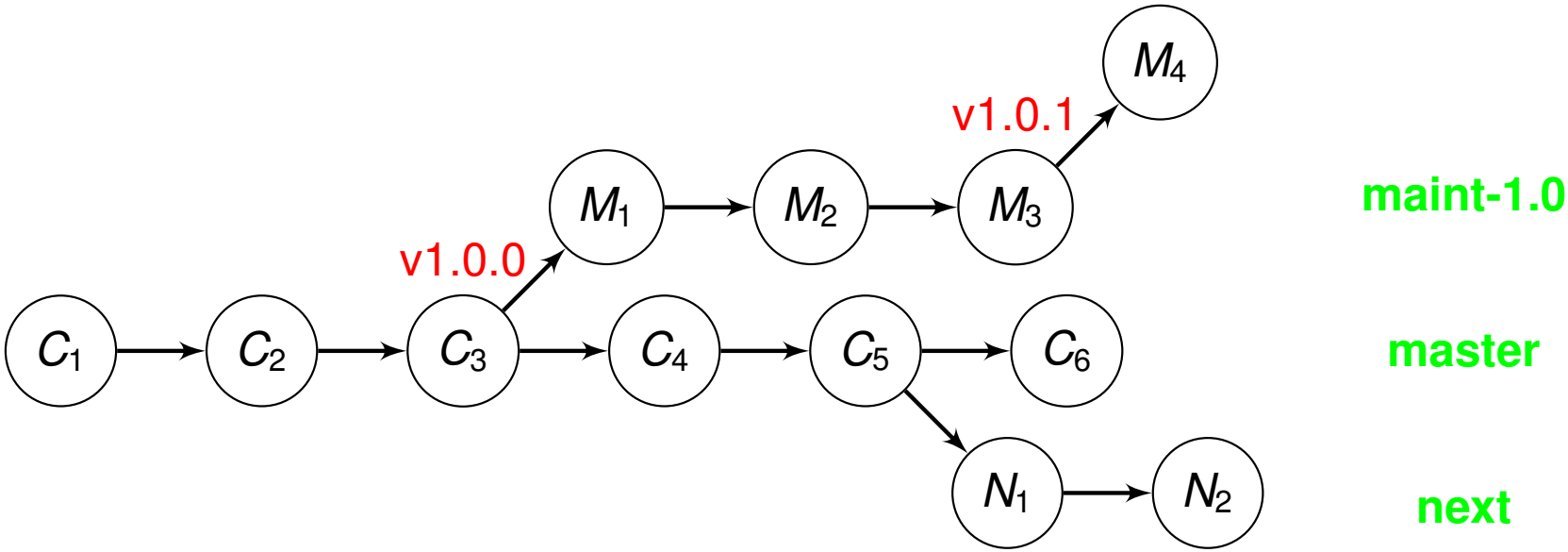
Git Branches



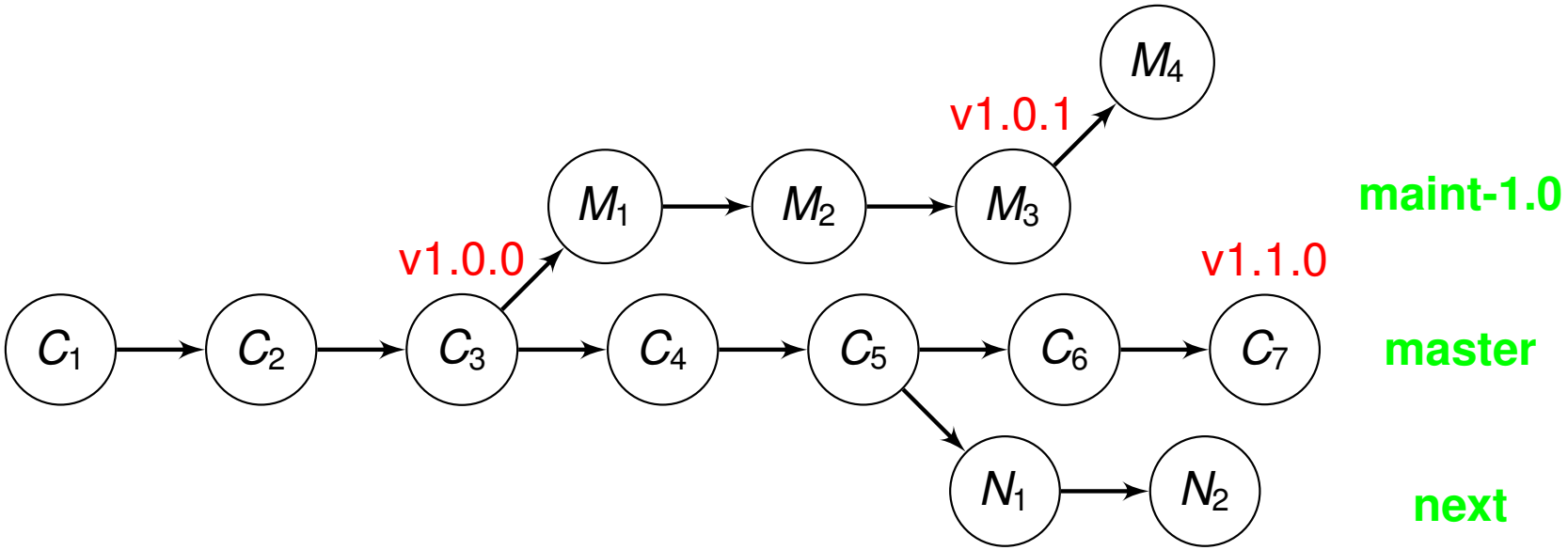
Git Branches



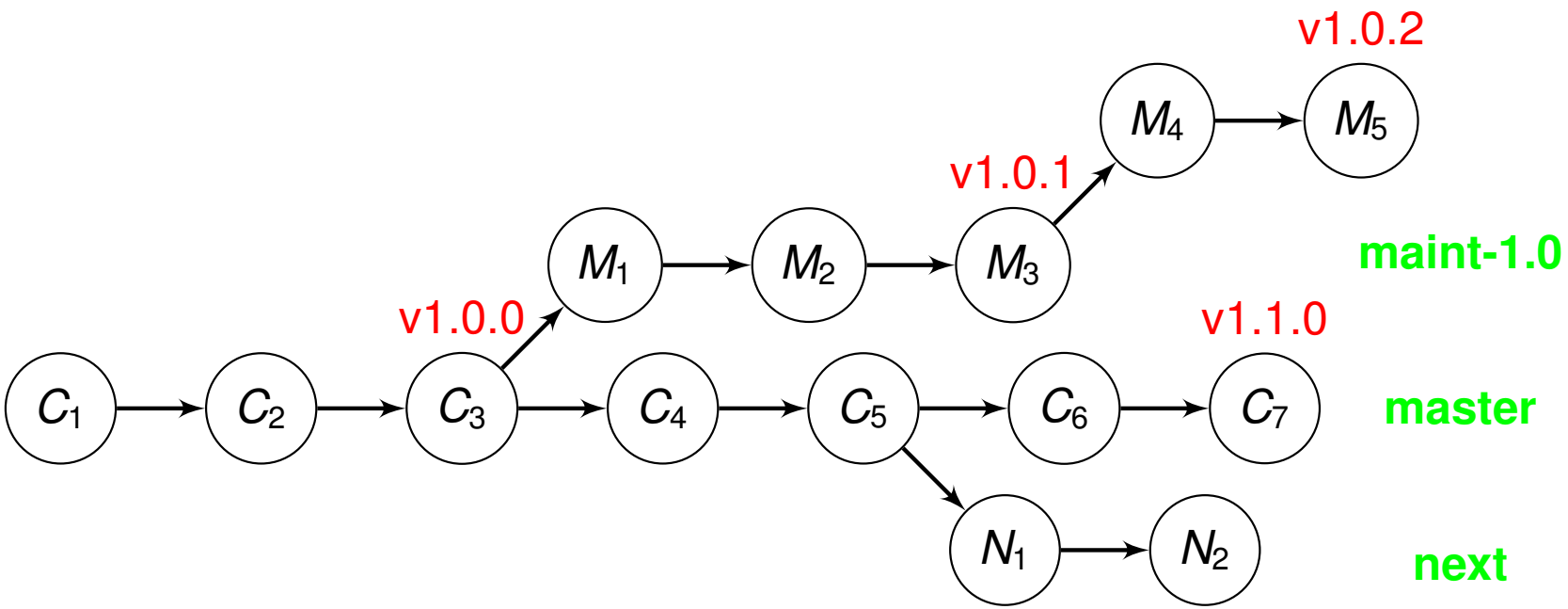
Git Branches



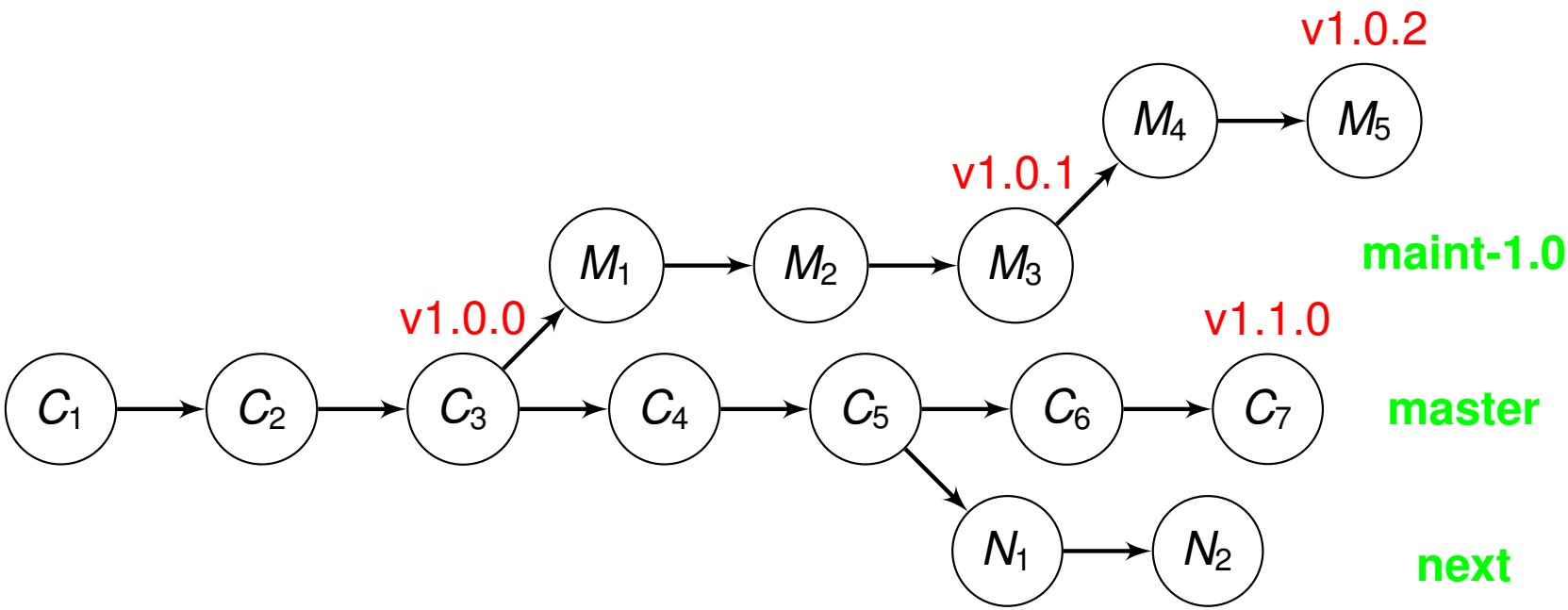
Git Branches



Git Branches

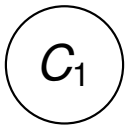


Git Branches



- **maint-X.Y** — release maintenance branches
- **master** — new feature branch
- **next** — things not yet release-ready

Git Development

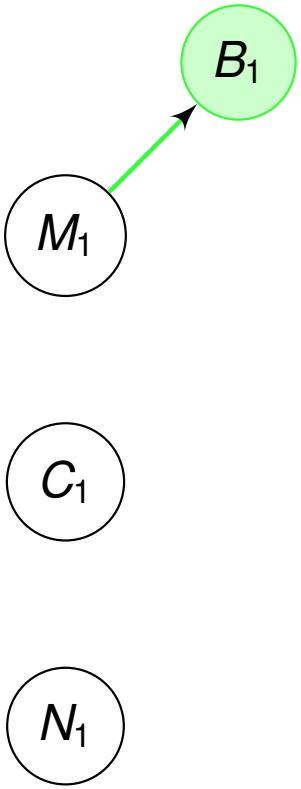


maint-1.0

master

next

Git Development



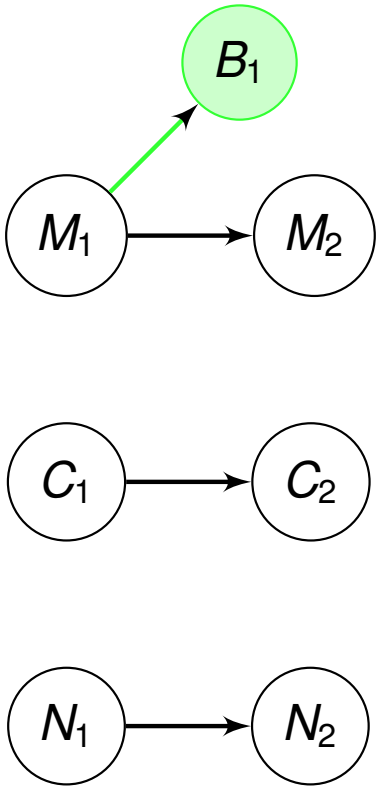
some-bugfix

maint-1.0

master

next

Git Development



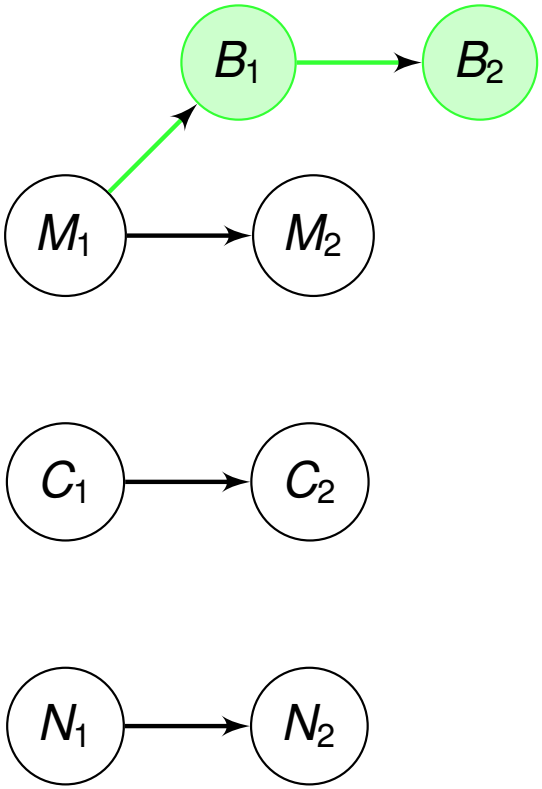
some-bugfix

maint-1.0

master

next

Git Development



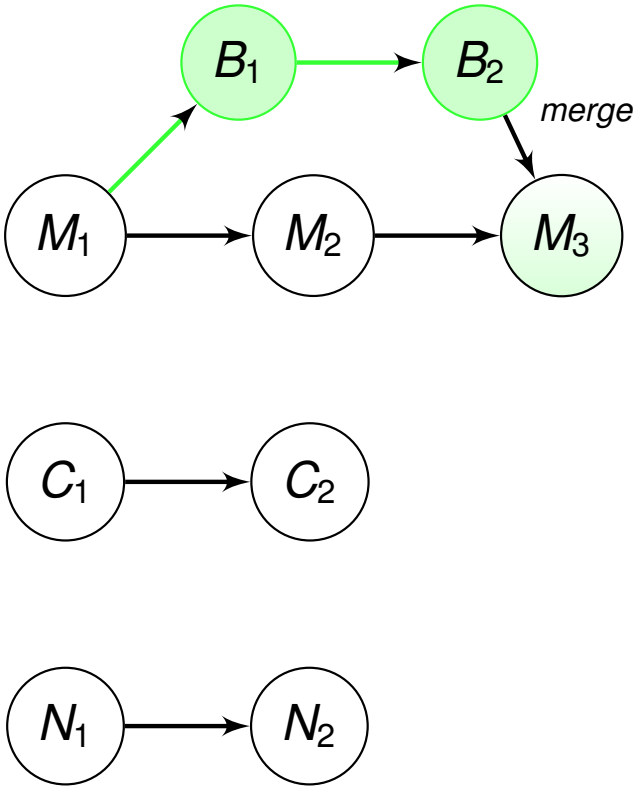
some-bugfix

maint-1.0

master

next

Git Development



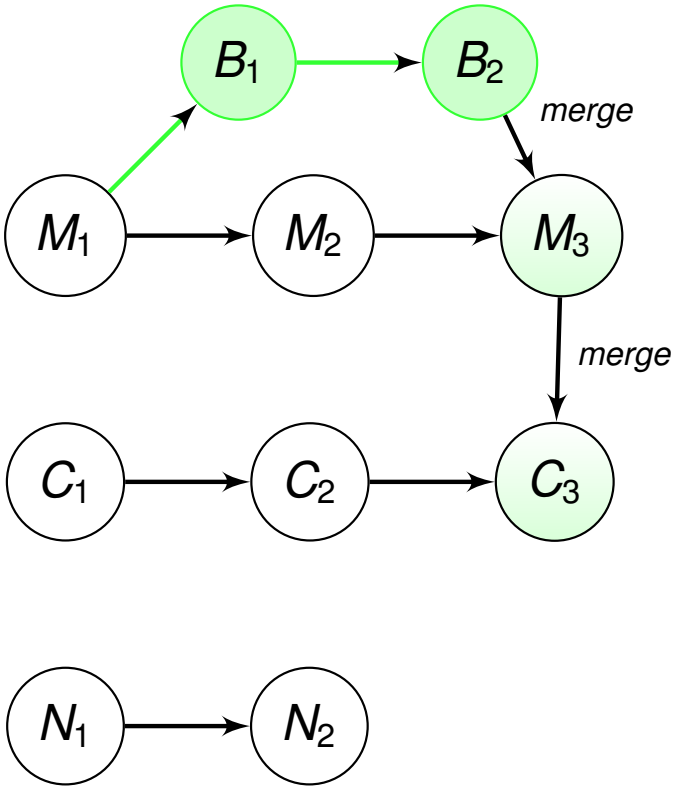
some-bugfix

maint-1.0

master

next

Git Development



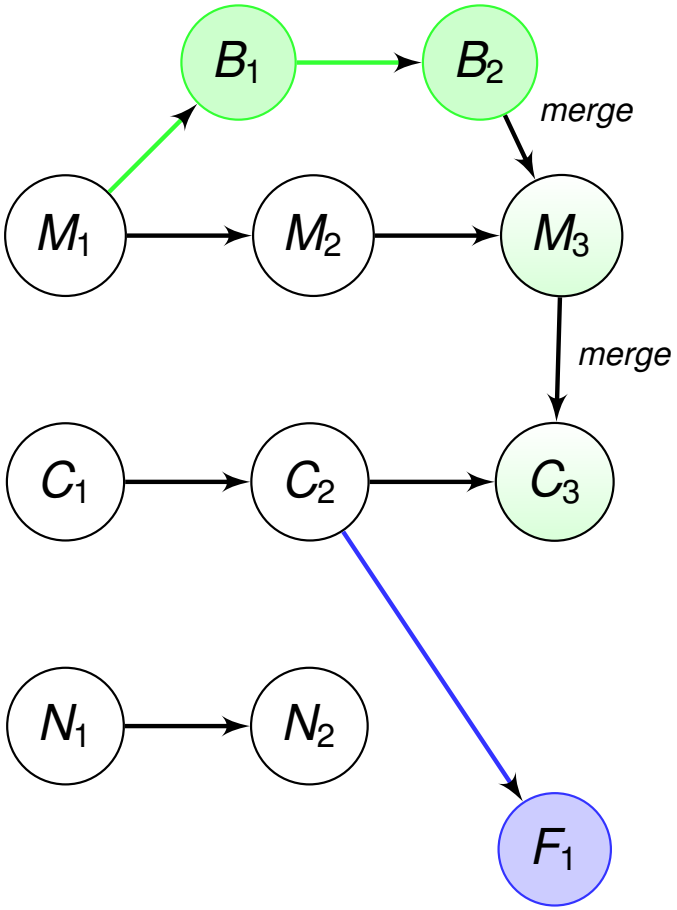
some-bugfix

maint-1.0

master

next

Git Development



some-bugfix

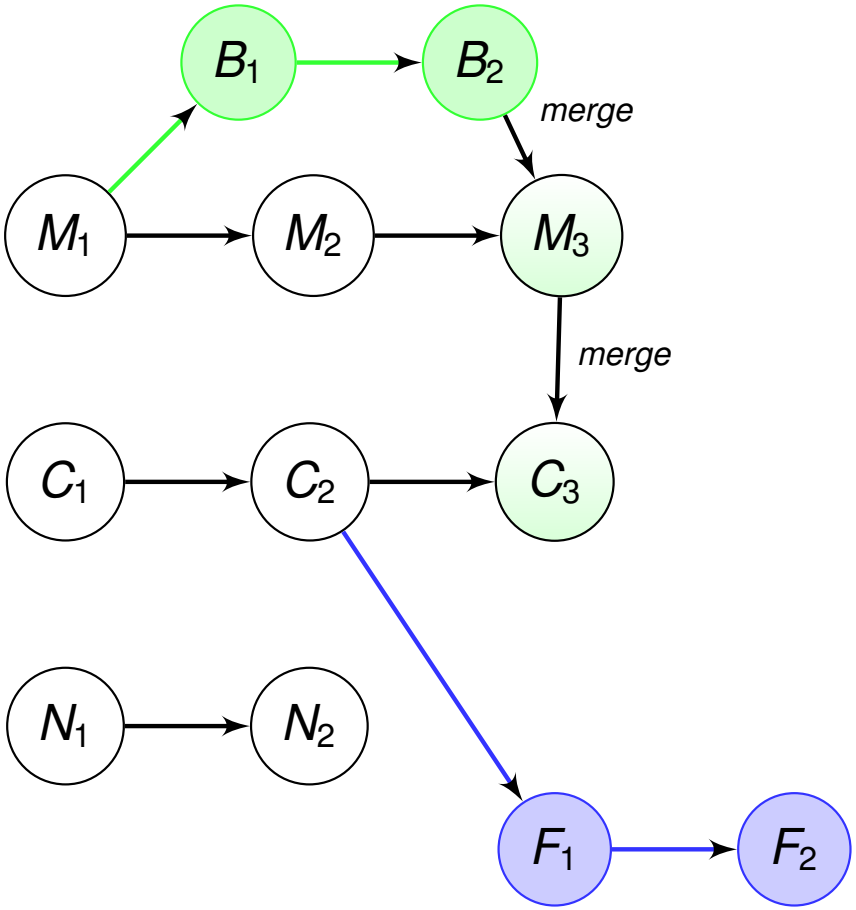
maint-1.0

master

next

some-new-feature

Git Development



some-bugfix

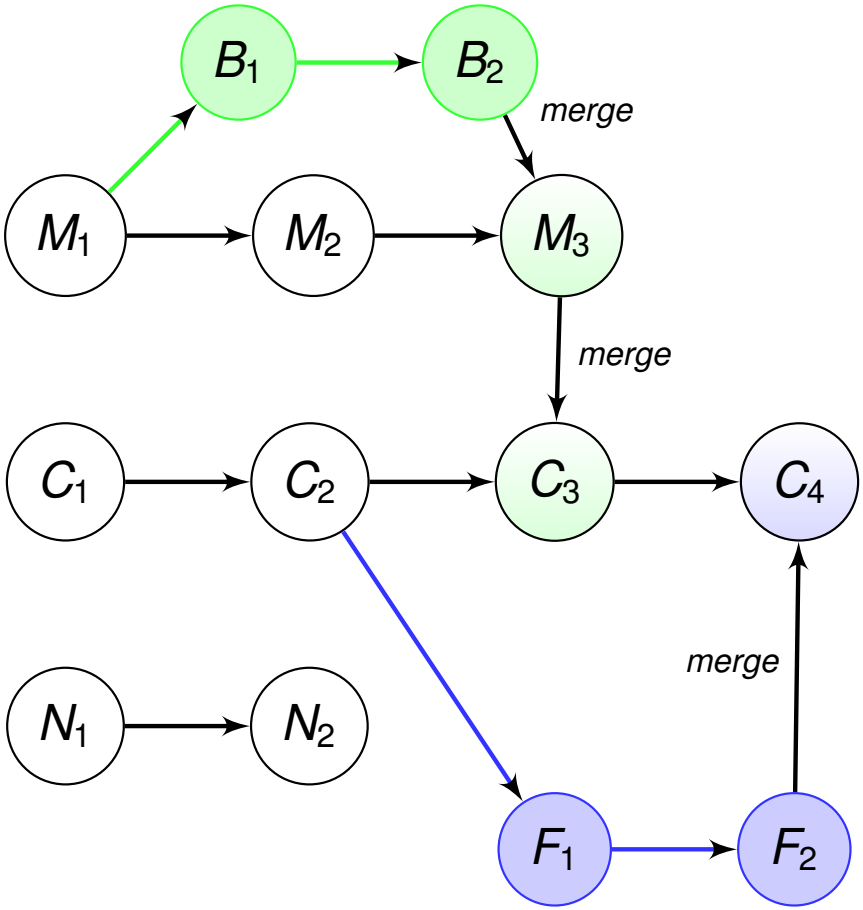
maint-1.0

master

next

some-new-feature

Git Development



some-bugfix

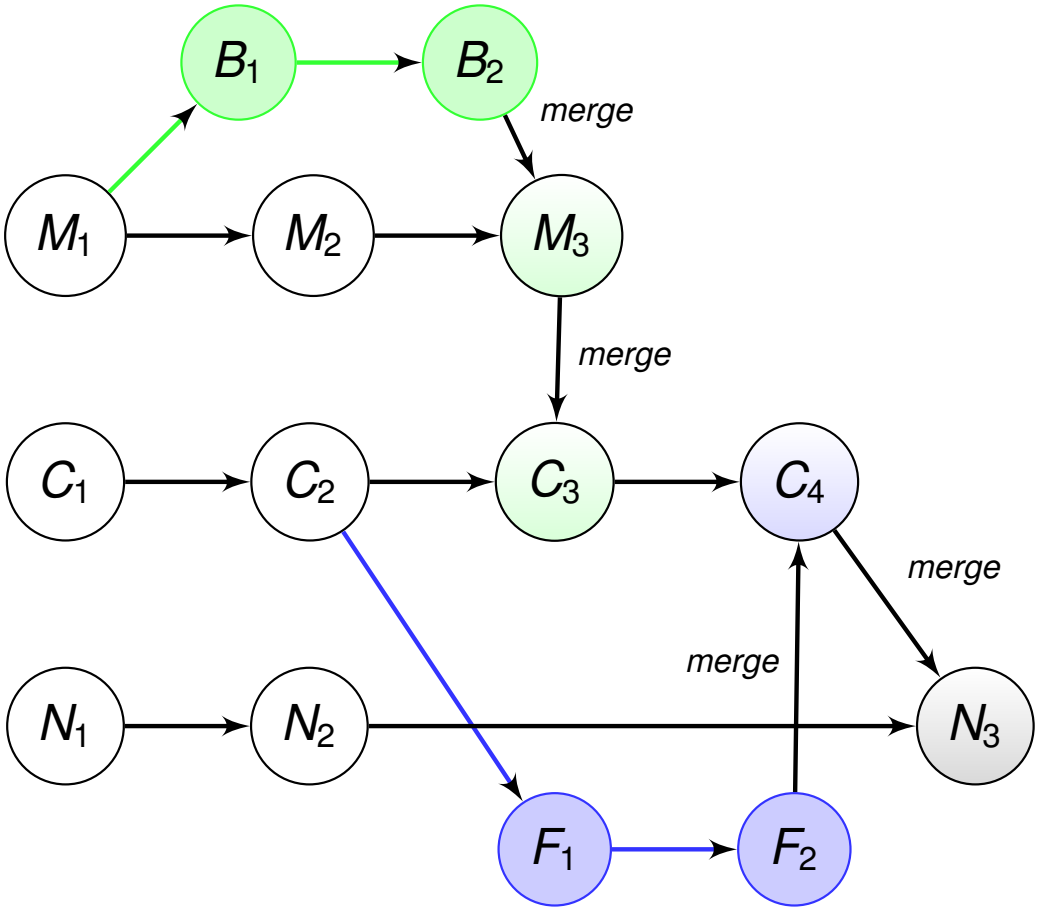
maint-1.0

master

next

some-new-feature

Git Development



some-bugfix

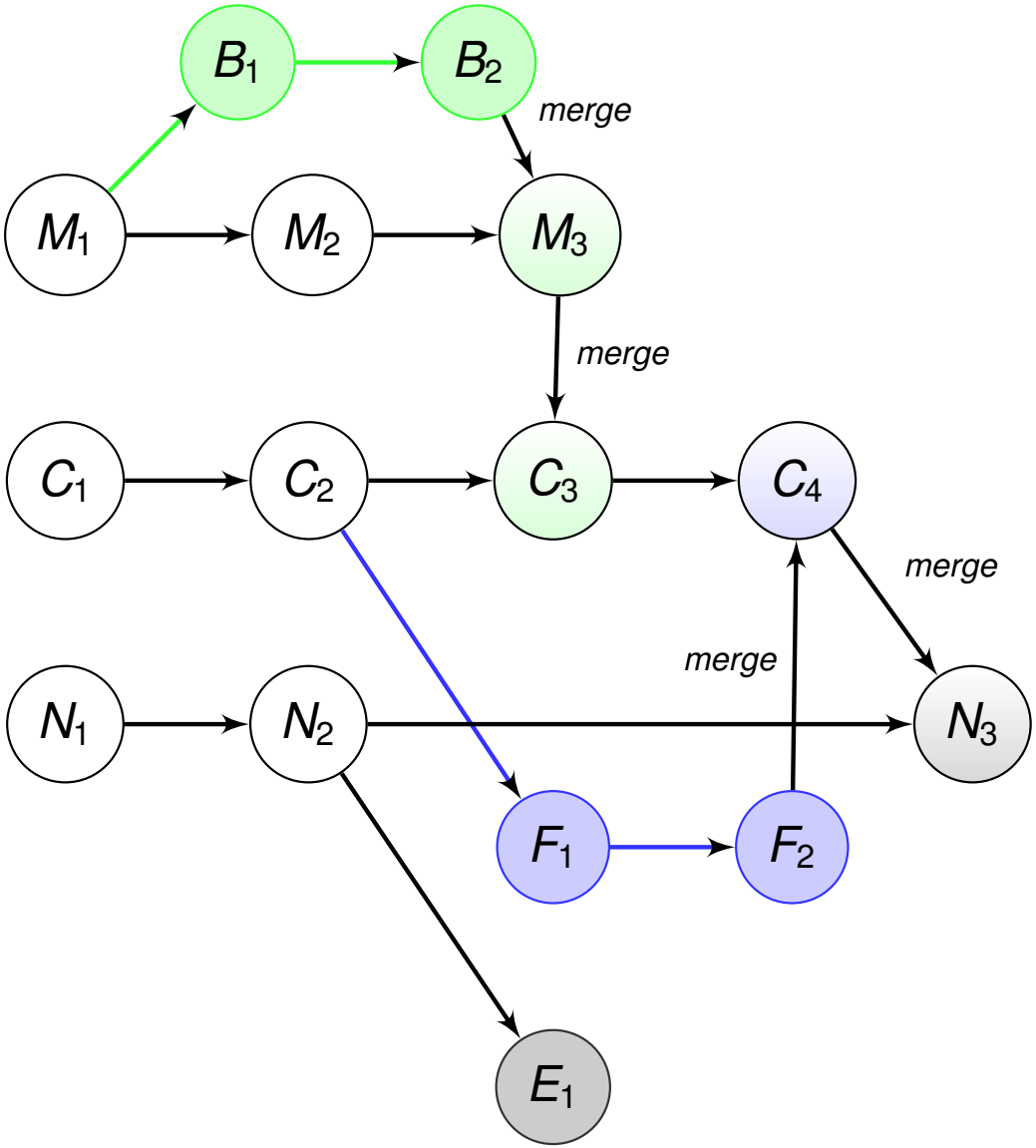
maint-1.0

master

next

some-new-feature

Git Development



some-bugfix

maint-1.0

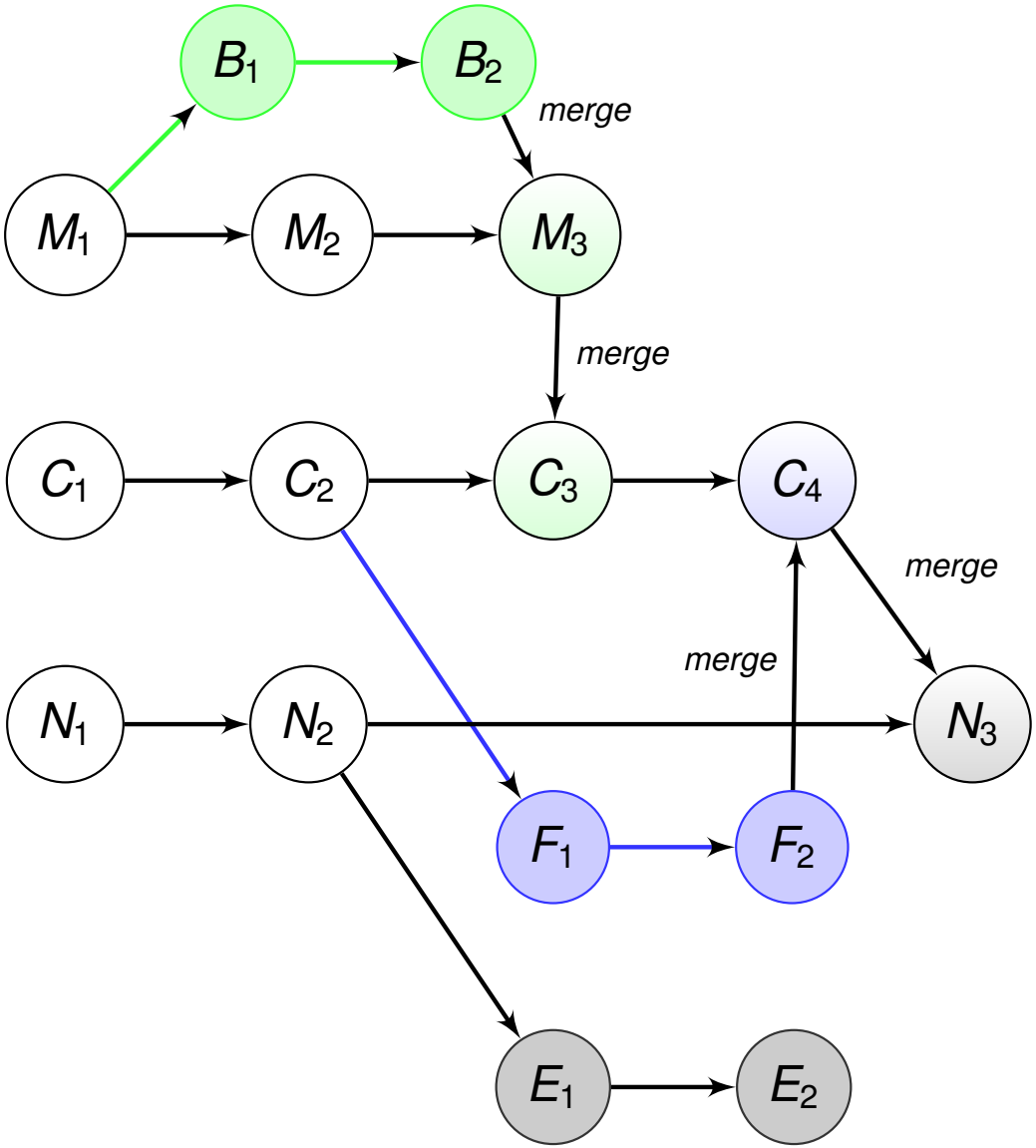
master

next

some-new-feature

some-experimental-feature

Git Development



some-bugfix

maint-1.0

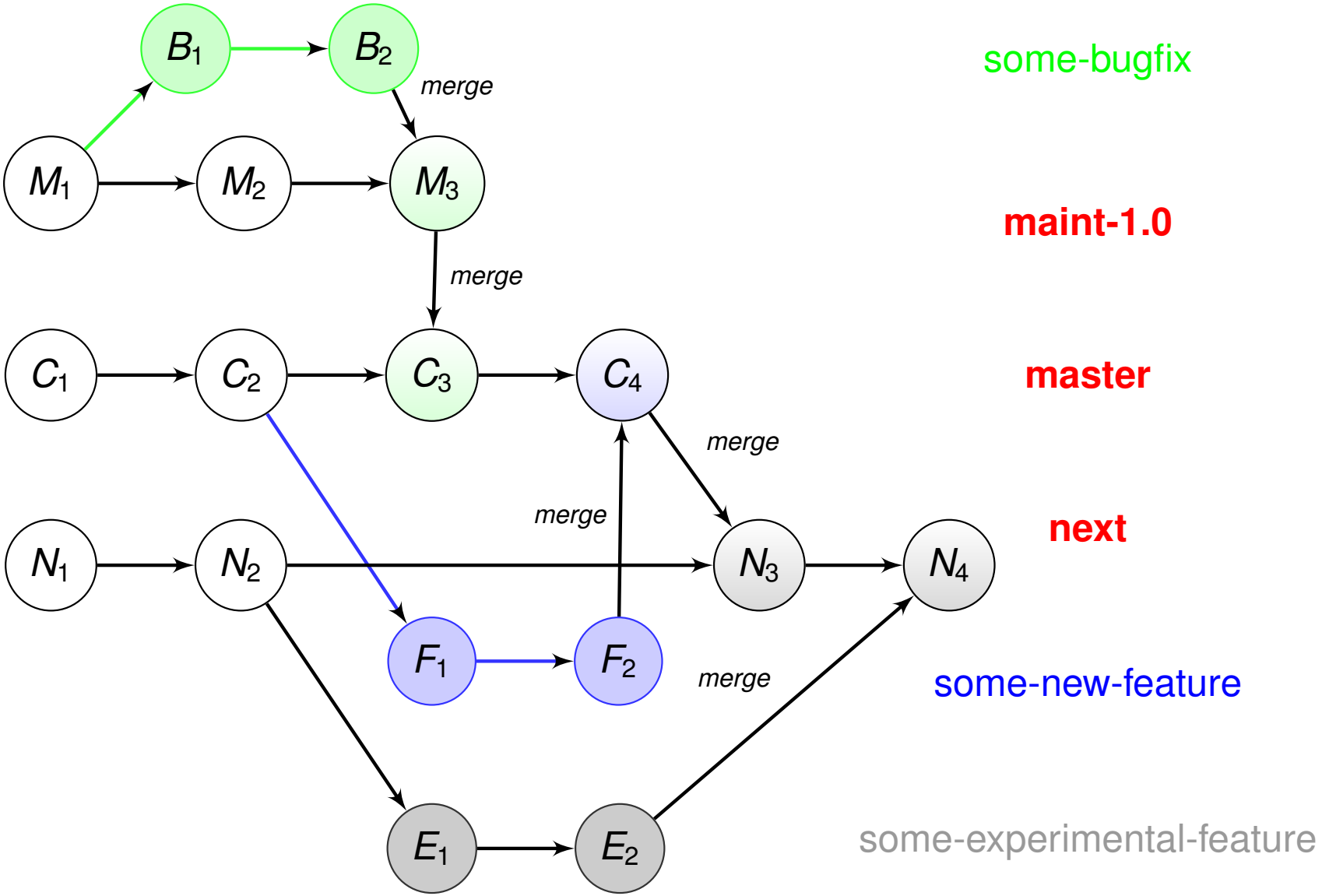
master

next

some-new-feature

some-experimental-feature

Git Development



Continuous integration

Outline

1 Introduction
- Digital Library
- Invenio

2 Case Studies
- Episode 1: Python
- Episode 2: Git
- Episode 3: Testing
- Episode 4: Building Efficient Indexes
- Episode 5: NIH
- Episode 6: Scalability

3 Conclusions

Unit testing

- **test-driven development** when appropriate
- e.g. before/while developing `strip_accents()`, write:

Example: search_engine_tests.py

```
class TestStripAccents(unittest.TestCase):
    """Test for handling of UTF-8 accents."""

    def test_strip_accents(self):
        """search engine - stripping of accented letters"""
        self.assertEqual("memememe",
             search_engine.strip_accents('mémêmëmè'))
        self.assertEqual("MEMEMEME",
             search_engine.strip_accents('MÉMÊMËMÈ'))
```

Functional testing

- functional/acceptance/regression testing
- testbed site (Atlantis of Institute Fictive Science)
- e.g. Python **mechanize** module to emulate browser

Example: websearch_regression_tests.py

```
class WebSearchSearchEnginePythonAPITest(unittest.TestCase):
    "Check typical search engine Python API calls on the demo data."

    def test_search_engine_python_api_for_failed_query(self):
        "websearch - search engine Python API for failed query"
        self.assertEqual([],
                         perform_request_search(p='aoeuidhtns'))

    def test_search_engine_python_api_for_successful_query(self):
        "websearch - search engine Python API for successful query"
        self.assertEqual([8, 9, 10, 11, 12, 13, 14, 15, 16, 17,
                          18, 47],
                         perform_request_search(p='ellis'))
```

Web testing

- sometimes we need to run tests in real browser
  - e.g. pages with heavy JavaScript
- using **Selenium** extension for Firefox
  - record and replay browser actions
  - test for text existence or non-existence on pages
  - test for link labels and targets

Example: websearch_web_tests.py

```
class InvenioWebSearchWebTests(InvenioWebTestCase):

    def test_search_ellis(self):
        """websearch - web test search for ellis"""
        self.browser.get(CFG_SITE_URL)
        p = self.browser.find_element_by_name("p")
        p.send_keys("ellis")
        p.submit()
        self.page_source_test(expected_text=[
            'Thermal conductivity of dense quark matter ' + \
            'and cooling of stars'])
```

Outline

1 Introduction
- Digital Library
- Invenio

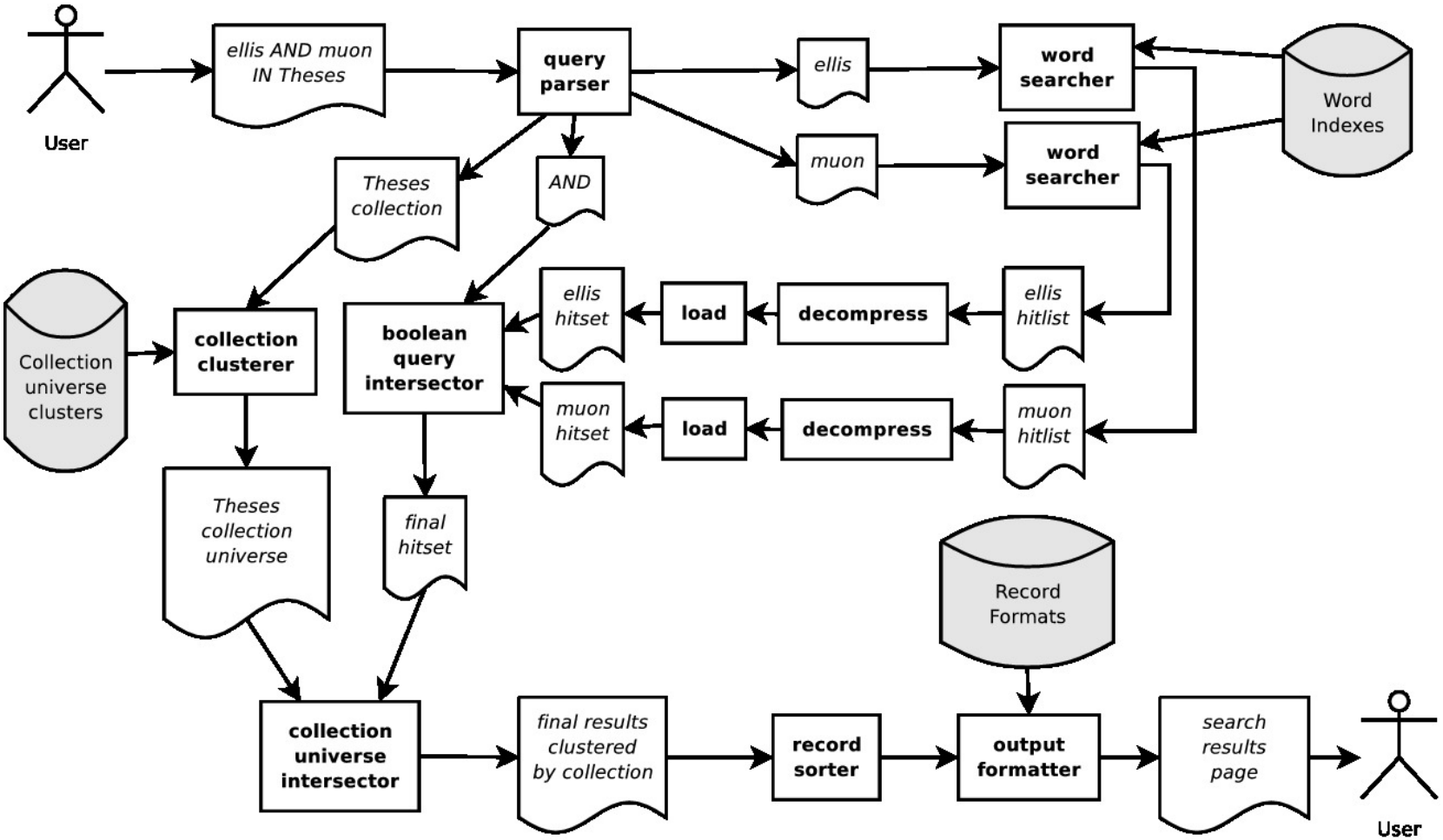
2 Case Studies
- Episode 1: Python
- Episode 2: Git
- Episode 3: Testing
- Episode 4: Building Efficient Indexes
- Episode 5: NIH
- Episode 6: Scalability

3 Conclusions

Designing A Search Engine

- **performance-driven design** assumptions:
  - high number of selects, low number of updates
  - fast searching, slow indexation
  - cache everything cacheable
- **search functionality:**
  - search for words, phrases, regular expressions
  - search in any field, authors, titles, etc
- **index design:**
  - forward indexes: $word1 \longrightarrow [rec1, rec2, \ldots]$
    $word2 \longrightarrow [rec2, rec7, \ldots]$
  - reverse indexes: $rec1 \longrightarrow [word1, word8, \ldots]$
    $rec2 \longrightarrow [word1, word2, \ldots]$
- **Zipf's law** on word frequency:
  - few words occur very often (e.g. *the*)
  - most words are infrequent (even e.g. *boson*)

Search Engine Under Cover



Measuring the Performance

- three important **speed factors** to consider:
  - speed of finding sets (DB Server)
  - speed of demarshaling sets (DB $\leftrightarrow$ Web App Server)
  - speed of intersecting sets (Web App Server)

Example: speed of various parts (2002, before optimization)

```
action / query:        "CERN 2002"   "of the this"
-----------------------------------------------
fetching                  0.28 sec      0.34 sec
demarshaling              0.78 sec      1.10 sec
adding colls              0.37 sec      0.63 sec
intersecting              0.64 sec      1.19 sec
-----------------------------------------------
total search time         2.07 sec      3.22 sec
```

Optimizing Data Structures

- **data structures** tested:
  - 'sorted' (lists, Patricia trees)
  - 'unsorted' (hashed sets, binary vectors)
- **fast prototyping**: (Python, Lisp in 2002)
  - throw-away coding to test ideas

Example: lists vs dicts, 350K sets in 800K universe

```
marshaling lists ..... 532616+532571 bytes in 1.33 sec
demarshaling lists ... 350000+350000 items in 0.10 sec
merging lists ........ 546965 items in 0.34 sec
intersecting lists ... 153035 items in 0.35 sec

marshaling dicts ..... 576491+576450 bytes in 0.87 sec
demarshaling dicts ... 350000+350000 items in 0.36 sec
merging dicts ........ 546965 items in 0.09 sec
intersecting dicts ... 153035 items in 0.15 sec
```

. . . and the winner is:

- **binary vectors** found the best compromise!
  - using `Numeric` Python module (in 2002)
  - typical search time gain: 4.0 sec $\rightarrow$ 0.2 sec (in 2002)
  - typical indexing time loss: 7 hours $\rightarrow$ 4 days (in 2002)
  - mostly spare data modelled via mostly dense data structure?
  - free your mind, think critically
- further optimisation:
  - `Numeric` module not addressing real bits, only bytes
  - so home-made `intbitset` C extension (2007)
    - addressing real bits, saving factor of 8 already
    - saving space, saving (indexing) time
  - use of external information retrieval tools (2011)
    - Solr, Xapian

Outline

1 Introduction
- Digital Library
- Invenio

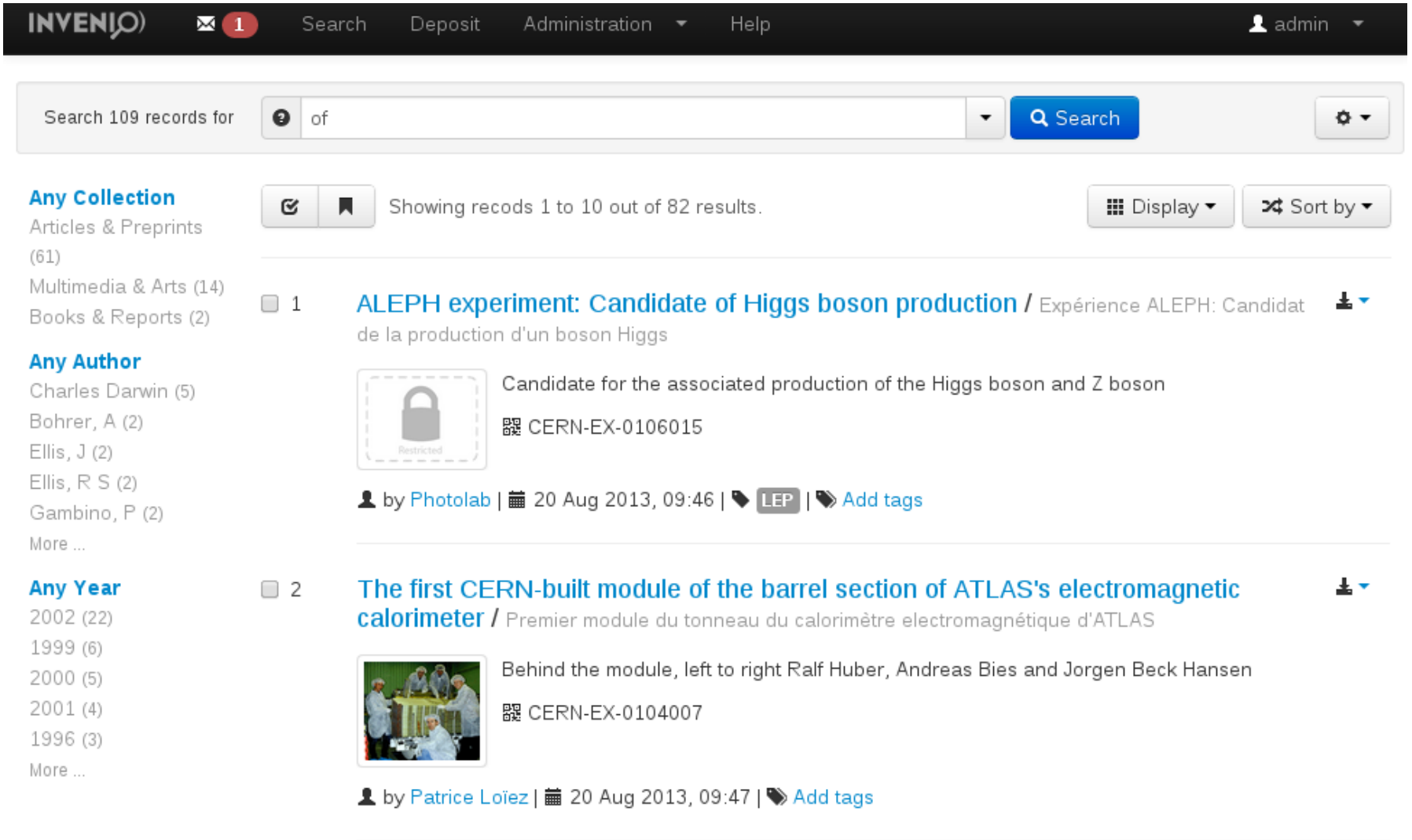
2 Case Studies
- Episode 1: Python
- Episode 2: Git
- Episode 3: Testing
- Episode 4: Building Efficient Indexes
- Episode 5: NIH
- Episode 6: Scalability

3 Conclusions

Not Invented Here

- **technology overview**:
  - load balancing: HAProxy
  - web application: Apache, WSGI, Python, Flask, Jinja, Cython
  - database: SQLAlchemy, MySQL/PostgreSQL/SQLite, MongoDB
  - indexing: Solr, Xapian
  - caching: Memcached, Redis
  - UI: Twitter Bootstrap, jQuery
  - mobile app: Apache Cordova
  - tools: Git, Trac, Jenkins, Selenium

Example: Invenio “next” branch UI



Example: ZENODO

zenodo

Research. Shared.

Search Upload Get started

Email Password Sign in

Home / Publications /
Improved photovoltaic performances by post-deposition acidic treatments on tetrapod shaped colloidal nanocrystal solids

10 July 2012

Journal article Embargoed access

Improved photovoltaic performances by post-deposition acidic treatments on tetrapod shaped colloidal nanocrystal solids

Mastria, Rosanna ; Rizzo, Aurora ; Nobile, Concetta ; Kumar, Susmit ; Maruccio, Giuseppe ; Gigli, Giuseppe

(show affliations)

The ligand exchange reaction with pyridine is the standard procedure for the integration of colloidal semiconductor nanocrystals (NCs) in photovoltaic devices; however, for large sized and irregularly shaped branched NCs, such as CdSe@CdTe tetrapods, this procedure can lead to a considerable waste of materials and the aggregation of NCs in the colloidal solution, therefore

Publication date
10 July 2012
Embargoed
Files available as Open Access after 10 July 2013
DOI
10.1088/0957-4484/23/30/305403
Report number(s):
OpenAIRE-ESCORT-2012-008
Published in:
Nanotechnology: 23 (2012) no. 30,
Funded by:
ESCORT - Efficient Solar Cells based on Organic and hybrid Technology (261920)
Collections:
Publications > Journal articles

Outline

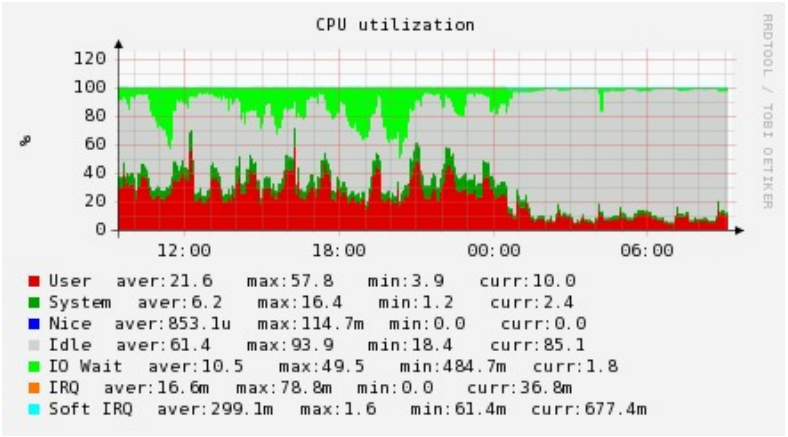
1 Introduction
- Digital Library
- Invenio

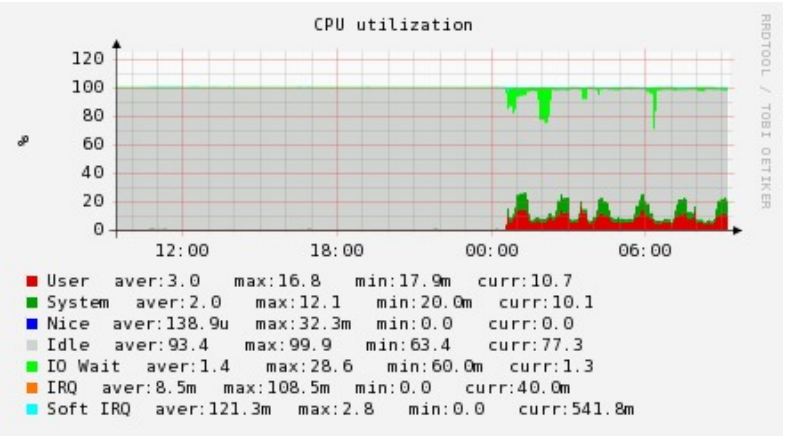
2 Case Studies
- Episode 1: Python
- Episode 2: Git
- Episode 3: Testing
- Episode 4: Building Efficient Indexes
- Episode 5: NIH
- Episode 6: Scalability

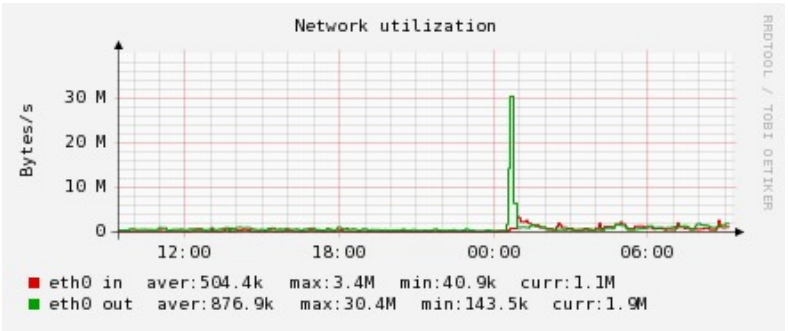
3 Conclusions

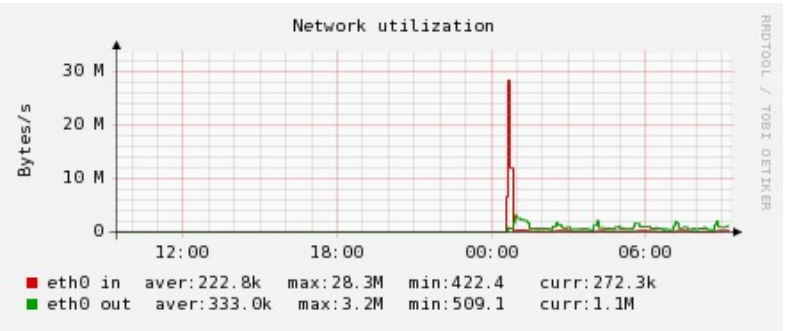
Splitting Web App Server and DB Server

- load of CDS Web and DB servers at the split time:





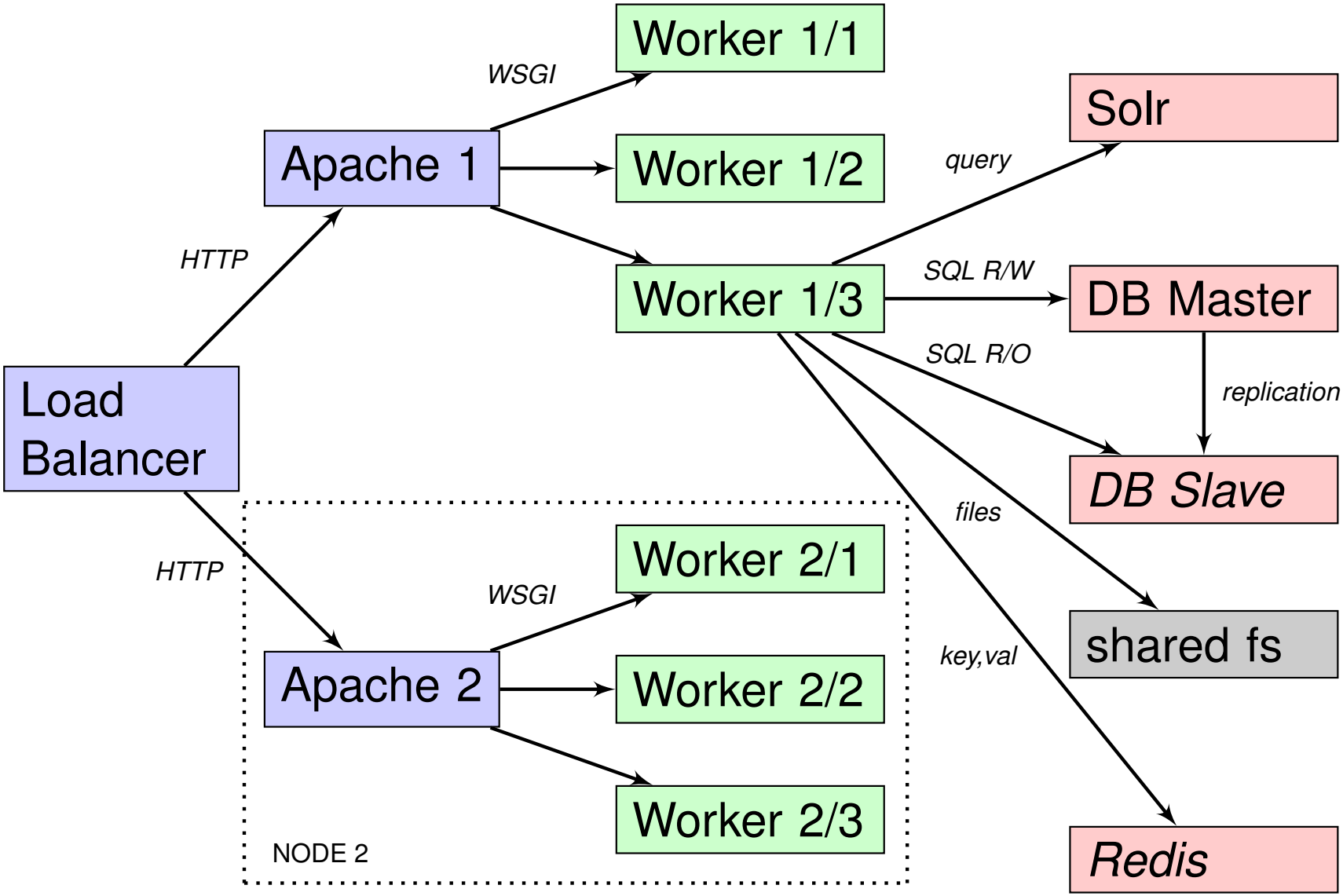




- split leads to efficient use of OS resources by lone, non-competing Web and DB daemon processes

Multi-Node Architecture

- 800 hits per sec on CDS during Higgs seminar July 4th



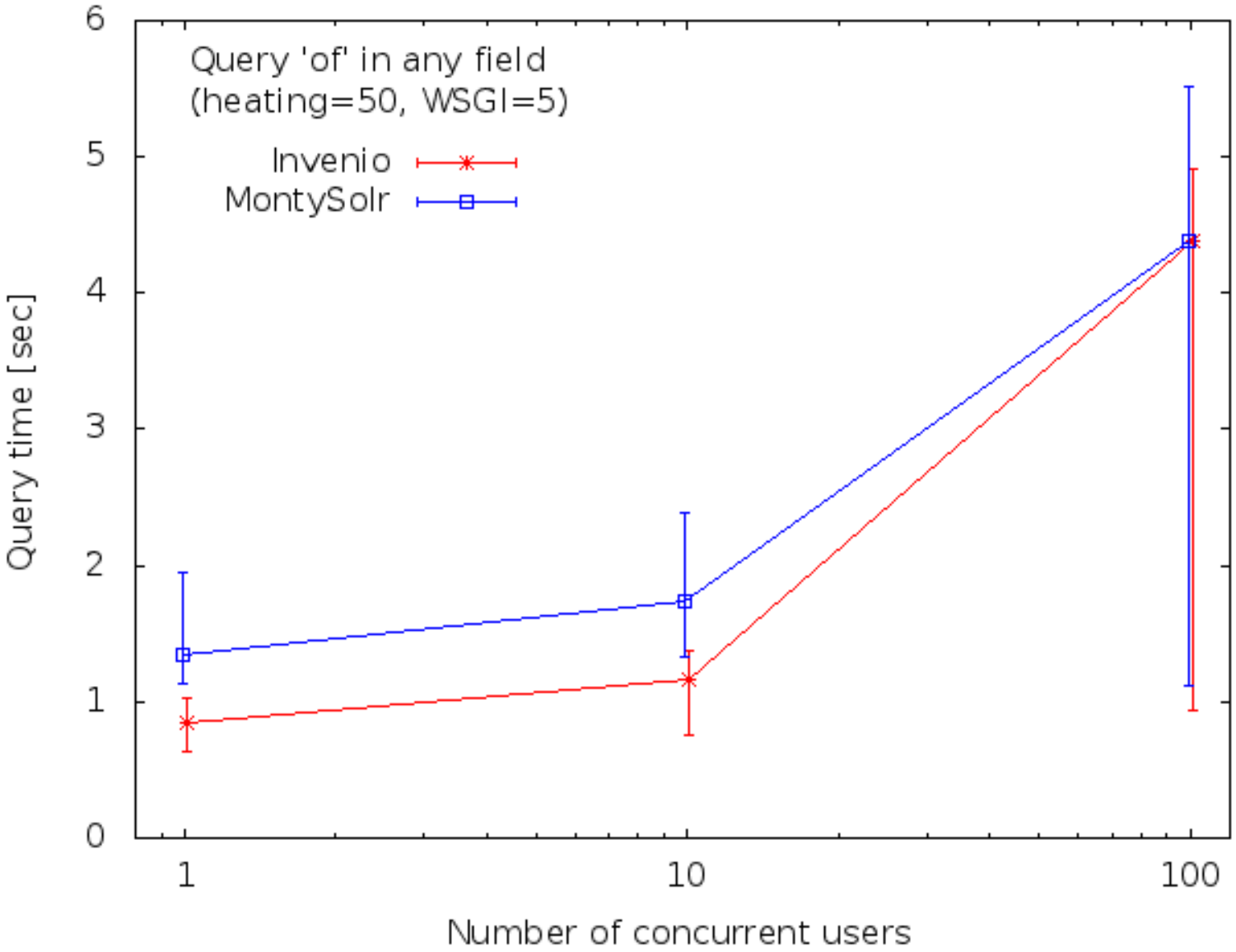
Measuring Scalability

- using **siege** and **ab** to simulate concurrent users and to measure throughput on a sample of typical URLs

Example: inspirehep.net under gentle siege

```
$ siege -d 1 -c 20 -t 1m -f inspirehep_urls.txt
Transactions:                   1329 hits
Availability:                 100.00 %
Elapsed time:                  60.23 secs
Data transferred:              37.12 MB
Response time:                  0.41 secs
Transaction rate:              22.07 trans/sec
Throughput:                     0.62 MB/sec
Concurrency:                    8.96
Successful transactions:        1329
Failed transactions:               0
Longest transaction:            3.05
Shortest transaction:           0.01
```

Measuring Scalability: “ab” on top of “siege”



Outline

1 Introduction
- Digital Library
- Invenio

2 Case Studies
- Episode 1: Python
- Episode 2: Git
- Episode 3: Testing
- Episode 4: Building Efficient Indexes
- Episode 5: NIH
- Episode 6: Scalability

3 Conclusions

Conclusions

- selected lessons from building a digital library system
  - 350,000+ LOC from 110+ authors over 10+ years
- selected technology:
  - load balancing: HAProxy
  - web application: Apache, WSGI, Python, Flask, Jinja
  - database: SQLAlchemy, MySQL/PostgreSQL/SQLite, MongoDB
  - caching: Memcached, Redis
  - UI: Twitter Bootstrap, jQuery
  - project tools: Git, Trac, Jenkins, Selenium
- morale from selected anecdotes?
  - value of rapid prototyping
  - value of organic-growth software development model
  - value of coding aesthetics and minimalism
  - *"Never Lose A Holy Curiosity"* (A. Einstein)