Invenio v2.0: A Pythonic Framework for Large-Scale Digital Libraries

Jiří Kunčar, Lars Holm Nielsen, and Tibor Šimko, for Invenio collaboration

Department of Information Technology, CERN, Geneva, Switzerland

February 10, 2014

Abstract

We describe the new major version of Invenio digital library software. The new version v2.0 comes after a major rewrite of the whole software platform. The main goal of the rewrite was to ease non-trivial platform customisation for large-scale repository installations. We present an overview of used technologies, followed by description of the most notable user-facing novelties. The release of Invenio v2.0 can be considered as a major milestone in converting a fairly tightly integrated digital library platform towards a coherent framework of communicating, adaptable and independent Pythonic digital library components.

1 Introduction

Invenio is a free software suite to run a digital library or a document repository on the web[1]. The technology offered by the software covers all aspects of digital library management, from document ingestion through classification, indexing, raking and curation up to document dissemination[2, 3]. Invenio has been originally developed at CERN to run the CERN document server[4], managing over 1,300,000 bibliographic records in high-energy physics since 2002, covering articles, books, journals, photos, videos, and more. Invenio is nowadays co-developed by an international collaboration comprising institutes such as CERN, DESY, EPFL, FNAL, SLAC and is being used by many more scientific institutions worldwide.

Invenio is especially suitable for big repositories such as CERN document server[4] or INSPIRE high-energy physics information system[5]. The use of Invenio for both the institution repository use case and the domain repository use case permits taking advantage of synergies and experimenting with new concepts such as "hosted collections" of mutually collaborating digital repository instances[6].

A growing use of Invenio for big repositories outside of the high-energy physics domain, such as ILO Labordoc[7], as well as growing needs to thoroughly modernise and customise the platform and its site-specific functionality and workflows, have motivated a thorough refactoring of the platform. The process started a couple of years ago and led to the birth of Invenio v2.0 that we shall briefly describe here.

2 Invenio v2.0

An outline of technologies used in Invenio v2.0 refactoring process has been described previously[8]. Since then, Invenio became a Pythonic framework for building data repositories using a flexible architecture on top of Flask[9] micro web development framework:

- New **source code organisation** has been introduced to logically split utilities from core bibliographic modules, to ease the separation of reusable components and extensions that can be put to use in multiple (even non-library-related) projects.

  (i) `base` — contains a lightweight application factory and functions for loading default configuration, extensions and modules.

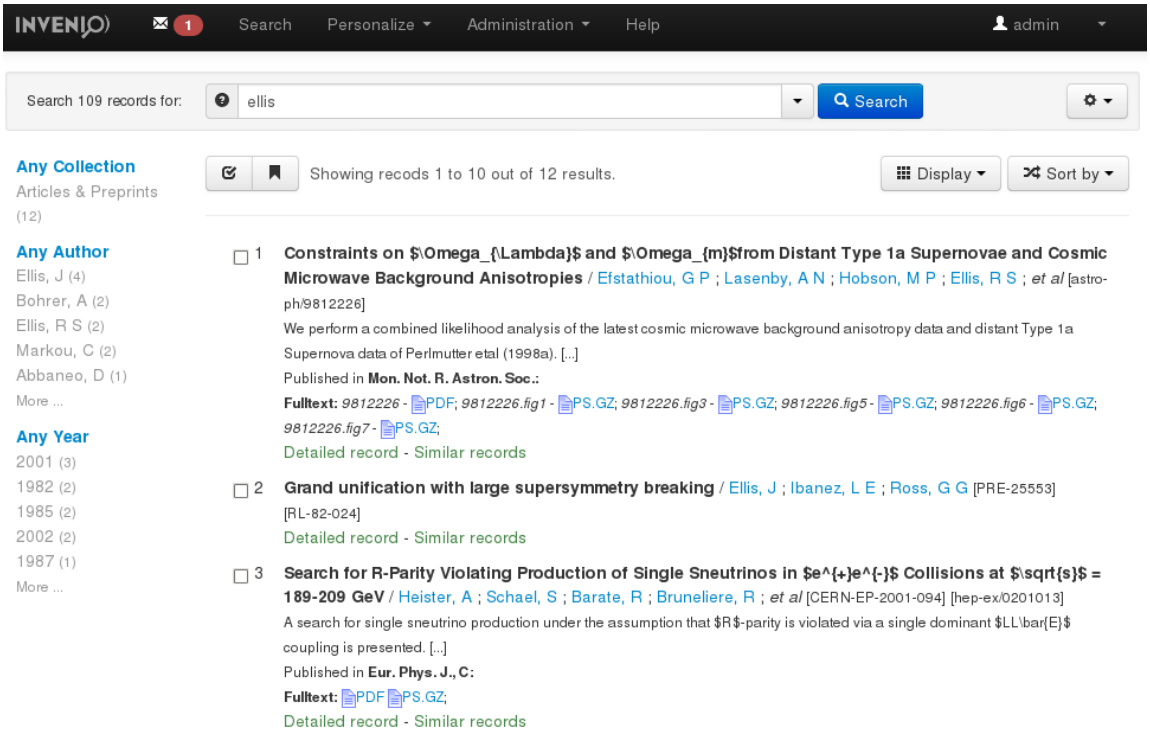

Figure 1: Example of Invenio v2.0 search page with faceted search on the lhs.

(ii) `ext` — links application-specific configuration and initialisation to general Flask extensions (e.g. login, internationalisation) and adds custom application tweaks for features that cannot be achieved otherwise (e.g. integration of legacy code and configuration).

(iii) `utils` — wraps various utility functions and packages to accomplish specific needs (e.g. DOI minting, plot extraction).

(iv) `modules` — defines bibliographic components and functions as well as common application functionality (e.g. searching, depositing, tagging, account management, etc).

- Focus on creating **fully Pythonic installation** process, removing all legacy remainders.
- New **upgrade system** containing per-module upgrade scripts and dependency graph handling.
- External **JavaScript** and **CSS** dependencies are installed using Bower[10] and Grunt[11] tools, easing the packaging process.
- New **SQLAlchemy** object-relational mapper[12] to achieve database independence, e.g. introducing support for PostgreSQL backend[14].
- Efficient **JSON** storage and indexing via MongoDB[13] and PostgreSQL[14].
- Fast **distributed caching** via Redis[15].

The rewrite led to the creation of standalone Flask extensions and other refactored standalone packages, including:

- **Flask-Registry** extension was created to permit easy dynamic inclusion of custom pluggable components, without having to modify the core Invenio source code in any way.
- **Flask-Menu** and **Flask-Breadcrumbs** extensions were created to enrich default UI with custom site-specific menu items and navigation bread crumbs.
- **Flask-SSO** extension was created to ease the integration of Shibboleth single-sign-on authentication method.

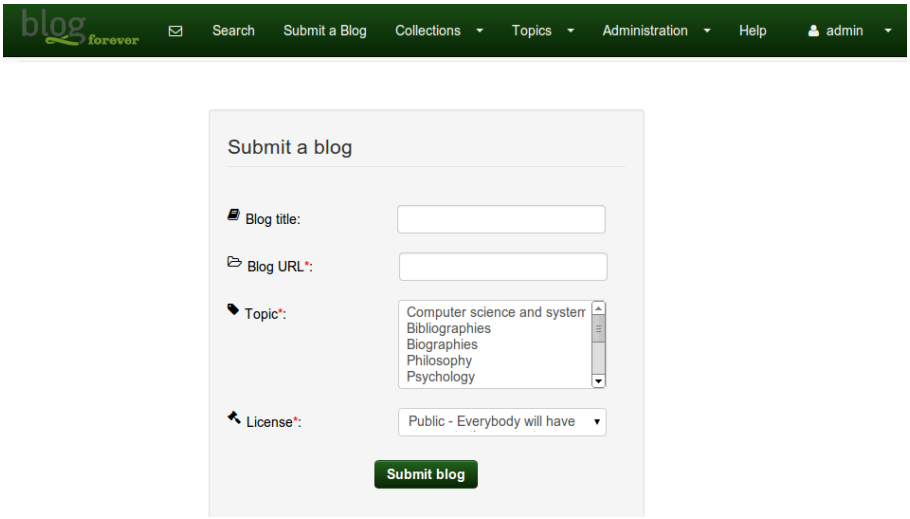

Figure 2: BlogForever simple submission interface: an example of site customisation.

- **invenio-demosite** source code overlay was separated out of the Invenio source code tree. This permits to rapidly clone a new custom site overlay repository for new installations.
- **intbitset** is Python C-based extension implementing fast integer bit set operation. It was developed within Invenio since many years and has now been separated to an independent package, permitting its use outside of the digital library scope.

Invenio v2.0 brings completely new look and many new functionalities. Among the most noticeable new features are:

- **modern web UI** based on Twitter Bootstrap framework[16];
- customisable **search UI** featuring **hierarchical facets**;
- extensible **document deposition system** featuring complete **REST API**;
- customisable **ingestion workflow engine** developed within the INSPIRE project[5];
- support for **multiple master formats** for records, permitting logical field abstraction and going "beyond MARC";
- **file storage abstraction** using cloud connectors, paving the way towards heterogeneous large-scale backends storing petabytes of research data;
- modern **tagging** and **annotation** user interface.

Figure 1 demonstrates the new search UI and the pluggable architecture for facet components that support various backends for building and rendering. Figure 2 demonstrates the flexibility of the platform in customising deposition workflows. Figure 3 demonstrates extending platform functionality via a new "community collections" concept example.

3 Conclusions

In this paper we have presented Invenio v2.0, a new major release of the Invenio digital library platform. After listing underlying technologies and basic principles of the new source code architecture, we have briefly discussed the most notable new features brought by the release. A thorough rewrite of the platform marks the evolution of Invenio towards a Pythonic framework of adaptable and communicating digital library components. The new system is already in production on several digital repository instances such as ZENODO[17] and BlogForever[18].

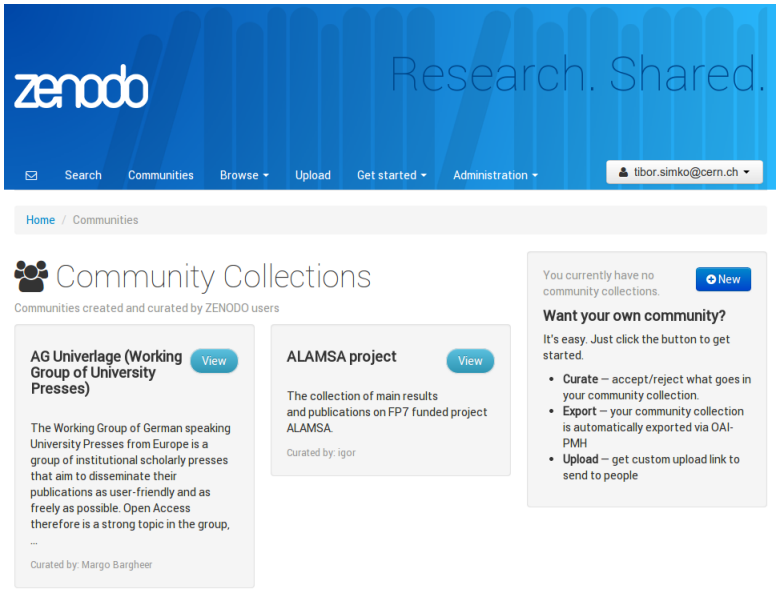

Figure 3: ZENODO community collections: an example of site specific functionality.

References

[1] Invenio digital library platform. `http://invenio-software.org/`

[2] S. Kaplun, *Invenio: A Modern Digital Library System*, Open Repositories 2010, Madrid, Spain, 6–9 July 2010.

[3] P. Glauner, J. Iwaszkiewicz, J.-Y. Le Meur, T. Šimko, *Use of Solr and Xapian in the Invenio document repository software*, Open Repositories 2013, Charlottetown, Prince Edward Island, Canada, 8–12 July 2013.

[4] CERN document server. `http://cds.cern.ch/`

[5] INSPIRE. `http://inspirehep.net/`

[6] N. Kasioumis, *External and Hosted Collections*, Invenio User Group Workshop 2012, Geneva, Switzerland, 7–9 May 2012.

[7] ILO Labordoc. `http://labordoc.ilo.org/`

[8] J. Kunčar and T. Šimko, *New Features and Technologies in Current and Future Invenio Versions*. Invited Talk at the 5th Seminar on Providing Access to Grey Literature, Prague, Czech Republic, 23 October 2012.

[9] Flask web development framework. `http://flask.pocoo.org/`

[10] Bower, a package manager for the web. `http://bower.io/`

[11] Grunt, a JavaScript task runner. `http://gruntjs.com/`

[12] SQLAlchemy, a Python SQL Toolkit and Object Relational Mapper. `http://www.sqlalchemy.org/`

[13] MongoDB document database. `http://mongodb.org/`

[14] PostgreSQL relational database. `http://postgresql.org/`

[15] Redis, an advanced key-value store. `http://redis.io/`

[16] Twitter Bootstrap web development framework. `http://getbootstrap.com/`

[17] ZENODO. `http://zenodo.org/`

[18] BlogForever. `http://blogforever.eu/`