QLectives – Socially Intelligent Systems for Quality
Project no. 231200

Instrument: Large-scale integrating project (IP)
Programme: FP7-ICT

Deliverable D4.2.4

Deployed QScience living lab v4

Submission date: 2013-03-01

Start date of project: 2009-03-01 Duration: 48 months

Organisation name of lead contractor for this deliverable:
University of Fribourg

Project co-funded by the European Commission within the Seventh Framework Programme (2007-2013)		
Dissemination level		
PU	Public	**X**
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidental, only for members of the consortium (including the Commission Services)	

Document information

1.1 Author(s)

Author	Organisation	E-mail
Matúš Medo	University of Fribourg	matus.medo@unifr.ch

1.2 Other contributors

Name	Organisation	E-mail
Sébastien Arnold	ETH Zurich	seba1511@tooski.ch
Stefano Balietti	ETH Zurich	stefano.balietti@gess.ethz.ch
Liao Hao	University of Fribourg	jamesliao520@gmail.com
Tamas Kojedzinszky	University of Szeged	tamas.kojedzinszky@gmail.com
David Rozas	University of Surrey	david.rozas@gmail.com
Xiao Rui	University of Fribourg	xrfind@gmail.com
Christian Schulz	ETH Zurich	cschulz@student.ethz.ch
Christoph Schwirzer	ETH Zurich	cschwirz@student.ethz.ch
Sam Sulaimanov	ETH Zurich	sam@bitmorse.com
Yanbo Zhou	University of Fribourg	nbboob@gmail.com

1.3 Document history

Version#	Date	Change
V1.0	19 February, 2013	First draft
V1.1	12 March, 2013	Approved version to be submitted to EU (small corrections)

1.4 Document data

Keywords	complex networks, distributed algorithms
Editor address data	matus.medo@unifr.ch
Delivery date	01 03, 2013

1.5 Distribution list

Date	Issue	E-mail
	Consortium members	QLECTIVES@list.surrey.ac.uk
	Project officer	Roumen.BORISSOV@ec.europa.eu
	EC archive	INFSO-ICT-231200@ec.europa.eu

QLectives Consortium

This document is part of a research project funded by the ICT Programme of the Commission of the European Communities as grant number ICT-2009-231200.

University of Surrey (Coordinator)
Department of Sociology/Centre
for Research in Social Simulation
Guildford GU2 7XH
Surrey
United Kingdom
Contact person: Prof. Nigel Gilbert
E-mail: n.gilbert@surrey.ac.uk

University of Fribourg
Department of Physics
Fribourg 1700
Switzerland
Contact person: Prof. Yi-Cheng Zhang
E-mail: yi-cheng.zhang@unifr.ch

Technical University of Delft
Department of Software Technology
Delft, 2628 CN
Netherlands
Contact Person: Dr Johan Pouwelse
E-mail: j.a.pouwelse@tudelft.nl

University of Warsaw
Faculty of Psychology
Warsaw 00927
Poland
Contact Person: Prof. Andrzej Nowak
E-mail: nowak@fau.edu

ETH Zurich
Chair of Sociology, in particular
Modelling and Simulation
Zurich, CH-8092
Switzerland
Contact person: Prof. Dirk Helbing
E-mail: dhelbing@ethz.ch

Centre National de la Recherche Scientifique, CNRS
Paris 75006,
France
Contact person: Dr. Camille ROTH
E-mail: camille.roth@polytechnique.edu

University of Szeged
MTA-SZTE Research Group on
Artificial Intelligence
Szeged 6720, Hungary
Contact person: Dr Mark Jelasity
E-mail: jelasity@inf.u-szeged.hu

Institut für Rundfunktechnik GmbH
Munich 80939
Germany
Contact person: Dr. Christoph Dosch
E-mail: dosch@irt.de

QLectives introduction

QLectives is a project bringing together top social modelers, peer-to-peer engineers and physicists to design and deploy next generation self-organising socially intelligent information systems. The project aims to combine three recent trends within information systems:

- **Social networks** - in which people link to others over the Internet to gain value and facilitate collaboration
- **Peer production** - in which people collectively produce informational products and experiences without traditional hierarchies or market incentives
- **Peer-to-Peer systems** - in which software clients running on user machines distribute media and other information without a central server or administrative control

QLectives aims to bring these together to form Quality Collectives, i.e. functional decentralised communities that self-organise and self-maintain for the benefit of the people who comprise them. We aim to generate theory at the social level, design algorithms and deploy prototypes targeted towards two application domains:

- **QMedia** - an interactive peer-to-peer media distribution system (including live streaming), providing fully distributed social filtering and recommendation for quality
- **QScience** - a distributed platform for scientists allowing them to locate or form new communities and quality reviewing mechanisms, which are transparent and promote

The approach of the QLectives project is unique in that it brings together a highly inter-disciplinary team applied to specific real world problems. The project applies a scientific approach to research by formulating theories, applying them to real systems and then performing detailed measurements of system and user behaviour to validate or modify our theories if necessary. The two applications will be based on two existing user communities comprising several thousand people – so-called "Living labs", media sharing community tribler.org; and the scientific collaboration forum EconoPhysics.

Executive summary

During the last year of the QLectives project, the work related to QScience has focused on the following areas:

1. Consolidation of the current codebase. All QScience components have been revisited, extensively debugged, and improved. Semantic validation of patterns files and refactoring of the javascript codebase in the VisualScience module are just two examples. Finally, the continuous integration of the Patterns module on Travis-ci proved to be an added value in this process.

2. Development of the QScience PDF parser. This new feature guarantees the extraction of metadata from PDF files. The extracted metadata is immediately inserted in the local database of the QScience instance, and at the same time uploaded to a central server for aggregated statistics.

3. Redesign of the VisualScience module. The VisualScience module has been completely overhauled to offer a better user experience. The interface has been simplified, and new features, such as relationships graphs, and comparison between users, have been added.

4. Further integration with Living Science. The integration with Living Science has been strengthened, and the API greatly improved. The goal for the coming months is to make the communication between Living Science and QScience bidirectional, in order to provide user generated data—such as paper ratings—to the Living Science server. The new information is then used to generate better statistics, rankings and visualizations, and to feed in the Living Archive.

5. Rationalizing the DrupalToDrupal module. As requested by the reviewers during last project meeting, the effort in this area has been focused in making the module more user-friendly, without hindering the security layer, and in supplying well-defined primitives for sharing data among QScience instances.

6. Launch of the QScience version of the Econophysics forum. A new QScience Econophysics forum has been created with new design and new features, such as the newly implemented Quality Trust Reputation (QTR) algorithm. The site will be launched in the coming weeks.

7. Launch of the Patterns central server web site. A central repository for user-generated patterns was created and it is going to be launched in the next weeks.

8. Dissemination within the Drupal community. We participated and promoted QScience and QLectives in several meetings and events relevant for the Drupal community. The positive effect of these activities is testified in the increasing use of the Patterns 7.x module.

9. Dissemination within the scholar community. For promoting QScience in the scholarly community, we created two additional templates that are ready to install: (i) the European project web site, and (ii) the research group home page. They will be used to replace the CRESS (`http://cress.soc.surrey.ac.uk/web/home`), and the QLectives (`http://www.qlectives.eu/`) web sites.

The QScience project is a real example of a successful collaboration between four universities: ETH Zurich, University of Frubourg, University of Surrey, and University of Szeged.

Contents

Chapter 1

Introduction

During the last year of the QLectives project, the work related to QScience has focused on the following areas:

1. Consolidation of the current codebase. All QScience components have been revisited, extensively debugged, and improved. Semantic validation of patterns files (Sec. 2.2.2) and refactoring of the javascript codebase in the VisualScience module (Sec. 2.6.2) are just two examples. Finally, the continuous integration of the Patterns module on Travis-ci (Sec. 2.2.5) proved to be an added value in this process.

2. Development of the QScience PDF parser. This new feature guarantees the extraction of metadata from PDF files (see Sec. 2.5). The extracted metadata is immediately inserted in the local database of the QScience instance, and at the same time uploaded to a central server for aggregated statistics.

3. Redesign of the VisualScience module. The VisualScience module has been completely overhauled to offer a better user experience. The interface has been simplified, and new features, such as relationships graphs, and comparison between users, have been added (see Sec. 2.6).

4. Further integration with Living Science. The integration with Living Science has been strengthened, and the API greatly improved. The goal for the coming months is to make the communication between Living Science and QScience bidirectional, in order to provide user generated data—such as paper ratings—to the Living Science server (see Sec. 2.7). The new information is then used to generate better statistics, rankings and visualizations, and to feed in the Living Archive.

5. Rationalizing the DrupalToDrupal module. As requested by the reviewers during last project meeting, the effort in this area has been focused in making the module more user-friendly, without hindering the security layer, and in supplying well-defined primitives for sharing data among QScience instances (see Sec. 2.4).

6. Launch of the QScience version of the Econophysics forum. A new QScience Econophysics forum has been created with new design and new features, such as

the newly implemented Quality Trust Reputation (QTR) algorithm (see Sec. 2.3). The site will be launched in the coming weeks.

7. Launch of the Patterns central server web site. A central repository for user-generated patterns was created and it is going to be launched in the next weeks (see Sec. 2.2.4).

8. Dissemination within the Drupal community. We participated and promoted QScience and QLectives in several meetings and events relevant for the Drupal community as described in Sec. 2.8. The positive effect of these activities is testified in the increasing use of the Patterns 7.x module.

9. Dissemination within the scholar community. For promoting QScience in the scholarly community, we created two additional templates that are ready to install: (i) the European project web site, and (ii) the research group home page. They will be used to replace the CRESS (`http://cress.soc.surrey.ac.uk/web/home`), and the QLectives (`http://www.qlectives.eu/`) web sites.

The QScience project is a real example of a successful collaboration between four universities: ETH Zurich, University of Fribourg, University of Surrey, and University of Szeged.

Chapter 2

Description of work

2.1 The Econophysics Forum

There are some new improvement of Econophysics Forum (EPF), include a new design and integrating with QTR and Recommendation module. The new EPF is now available at `http://134.21.2.151/drupal-7.14`, the online version available on the Fribourg University servers at `http://unifr.ch/econophysics/` will be updated soon.

2.1.1 New design

The new design is implemented using the theming system of Drupal. Drupal provides a very flexible theming system. With a theme, you can change the interface of a Drupal site. Here we present some screenshots of the new designed Econophysics Forum. In the home page, the main types of contents are displayed as blocks in the main column, and the other contents lists (top papers, recommendation list, similar users, recent comments) are shown in the left column. See Fig. 2.1 for the EPF front page as shown to a user who is logged in.

In the new EPF, every user has a profile block to show his basic profile information (see the top of the right column in Fig. 2.1). In the user's profile page, a chart is added to show the user's reputation history (this uses the newly developed QTR algorithm which is described in 2.3)—see Fig. 2.2 for an illustration.

New content is now added in pop up boxes instead of on separate pages (see Fig. 2.3 for an illustration).

2.1.2 Migration of the data

Since the new Econophysics Forum is based on the open CMS Drupal, the database structure of the new EPF is totally different from the old one. We have to migrate the EPF data from the old database to the new database. In Drupal, there is a migrate module that provides a flexible framework for migrating content into Drupal from other sources. The Migrate module is a developer's tool. The primary purpose is to provide the infrastructure for managing complex, large-scale site migrations. It supports the creation of core Drupal objects such as nodes, users, files, terms, and comments,

Figure 2.1: Front page of the redesigned Econophysics Forum.

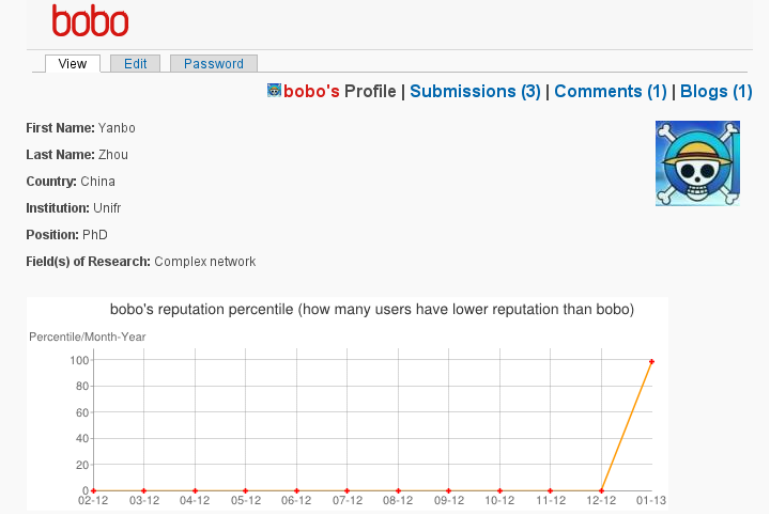

Figure 2.2: Personal page of a user registered in the Econophysics Forum: basic information, links to the content submitted by this user and a chart of this user's reputation are shown here.

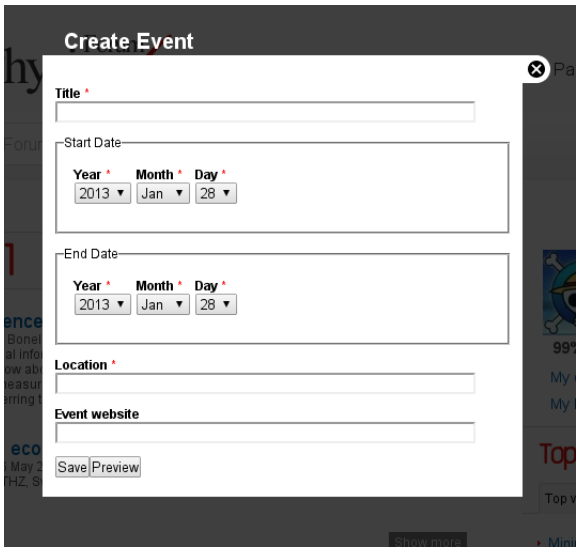

Figure 2.3: Pop up window through which new content is submitted (in this case, a new event).

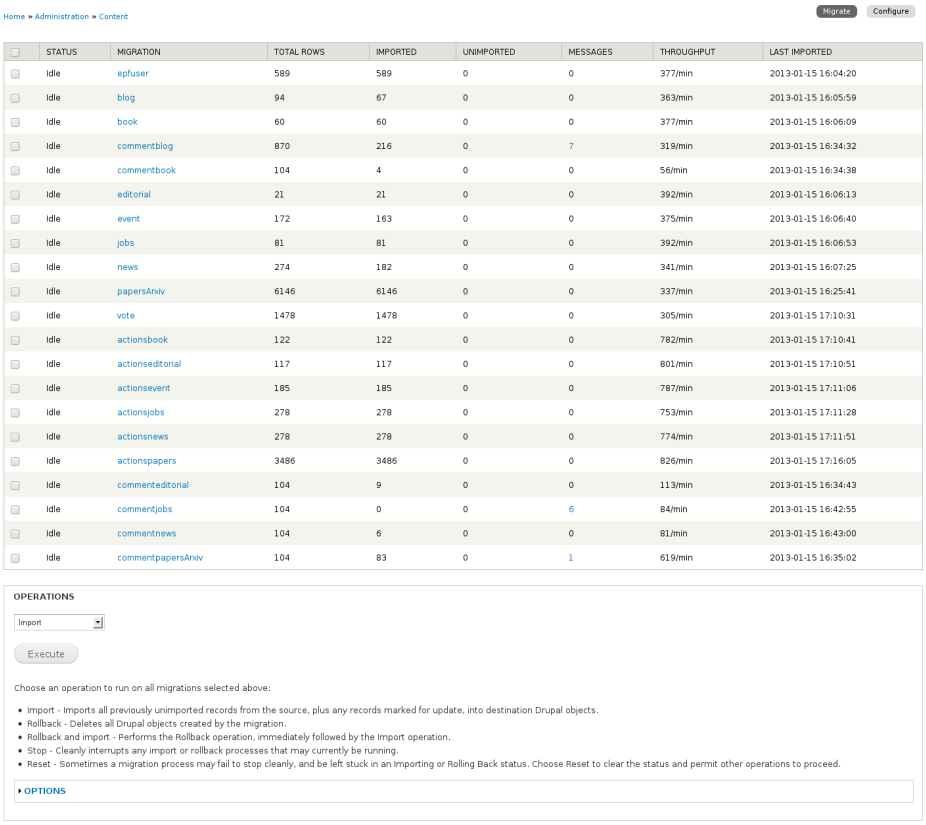

Figure 2.4: Screenshot of the migration interface.

and it can be easily extended for migration of other kinds of content. Content can be imported and rolled back using a bundled web interface (`Migrate UI` module).

Using the migrate module, we add maps from the old EPF database to the new database for every object, such as users, contents, comments and even the view,vote and download actions. And then the old EPF data is migrated to the new database correctly according to the maps. After the migration, user passwords can also be kept, which means that the old EPF users can login to the new Drupal version of EPF using their original user name and password without any extra modification. The screenshot of migration interface is shown in Fig. 2.4.

2.1.3 Recommendation module

Recommendation is an important function of the EPF. A recommendation system can help users navigate to pages that might be of interest to them, and users can get useful information efficiently from large quantities of information. In the recommendation module, the user-based Collaboration Filtering algorithm is used. With this algorithm, all users have vectors to value their preferences for every content item (papers, news, events, etc.). The preference value is calculated from the user's action history. Different actions have different weights. For example, the vote action has more weight than the view action. Then, the Pearson Correlation Coefficients of different users' preference vectors are calculated to measure the similarity of different users. Finally,

Figure 2.5: Presentation of recommendations to a user.

the recommendation content list for a user will be obtained from the other users most similar to him or her.

As there are many content items and actions in the EPF, it may take a long time to calculate the recommendation result. However, the user's preferences do not change frequently. Thus to improve the user experience, rather than calculate the recommendation result in real time, we save the users' similarities in the database, and refresh it every day. With the recommendation module, user can see a recommended contents list and a similar users list block on the right column of the site after login. A screenshot of the recommendation results shown to a user is presented in Fig. 2.5.

2.1.4 QScience module

QScience module can recreate the basic features of the Econophysics forum on a general level. The QScience module will:

- Add the following fields to the user profile:
  - First Name,
  - Last Name,
  - Country,
  - Institution (autocomplete tag from a list of institution),
  - Field(s) of research (economics, computer science, etc.)
  - Position (PhD candidate, professor, post-doc, etc.),
  - Picture (a picture of the person),
  - Link to a home page or blog,
- Create the following content types, and related fields:
  - Paper: title, author, abstract, journal, external URL,

- Paper form arXiv: serial number,
- Book Review: title, author, reviewer, review,
- Institution: title, country, city, web site,
- Journal: title, field,
- Event: title, start date, end date, location, web site,
- News: title, prefix (call for papers, announcement, job opening, award, conference report, others), body, web site,
- Job Opening: title, body, web site.

2.2 Patterns module

2.2.1 Patterns Progress

During the past months the architecture of Patterns has been extended to enhance the automatic export capabilities and to provide a new level of semantic validation in order to offer assistance in the task of solving the possible conflicts between the entities generated by the components. There has also been an effort on providing an extensive technical documentation and on disseminating the project by participating in several international Drupal meetings in order to receive the feedback and attract the attention of the Drupal community.

Improvement of the export functions and components extension

The new version of Patterns provides a set of enhancements to export automatically the configuration of a site. All the main components possess now automatic export capabilities, and depending on the type of component new options to export it are offered. Two different use cases are distinguished:

1. Export the configuration as a pattern consisting of a set of CREATE actions which allows users the creation of patterns for a "fresh installation" of certain features,

2. Export the configuration as a pattern consisting of a set of MODIFY actions which allows users the creation of patterns to override the current settings (e.g., to update the features developed in a testing site to a production site). The patterns generated ensure idempotency in the operations.

An analysis was performed in order to classify the required export configuration processes according to the semantic of the tag itself. For most of the components exporting the configuration as a set of CREATE or MODIFY actions makes sense semantically for both cases, but there are some exceptions, e.g., for the colour component only the MODIFY case makes sense, since it is always present in the system. Table 2.1 summarizes the required export configuration processes for each tag.

On the other hand, some of the components have been extended as well to provide a richer syntax and to divide some complex actions into a set of more basic ones, keeping with the principles of modular design. E.g., the block component now offer especific tags for the settings related to the scope in terms of content type or role.

Component: tag / process	CREATE actions	MODIFY actions
Menu: menu	X	X
Menu: menu_item	X	X
Block: block	X	X
Block: block_node_type	X	
Block: block_role	X	
Colour: colour		X
Field: field	X	X
Field: instance	X	X
Content Type: content_type	X	X
Content Type: node	X	X
Taxonomy: vocabulary	X	X
Taxonomy: term	X	X
User: role	X	X
User: permission		X
User: user	X	X
User: roles_user	X	
Pathauto: settings		X
Pathauto: patterns		X
Shortcut: shortcut_set	X	X
Shortcut: shortcut_link	X	X
Shortcut: shortcut_set_user		X
System: variables		X
System: modules	X	
System: theme		X
Toolbar: toolbarrole		X

Table 2.1: Required configuration processes for each tag.

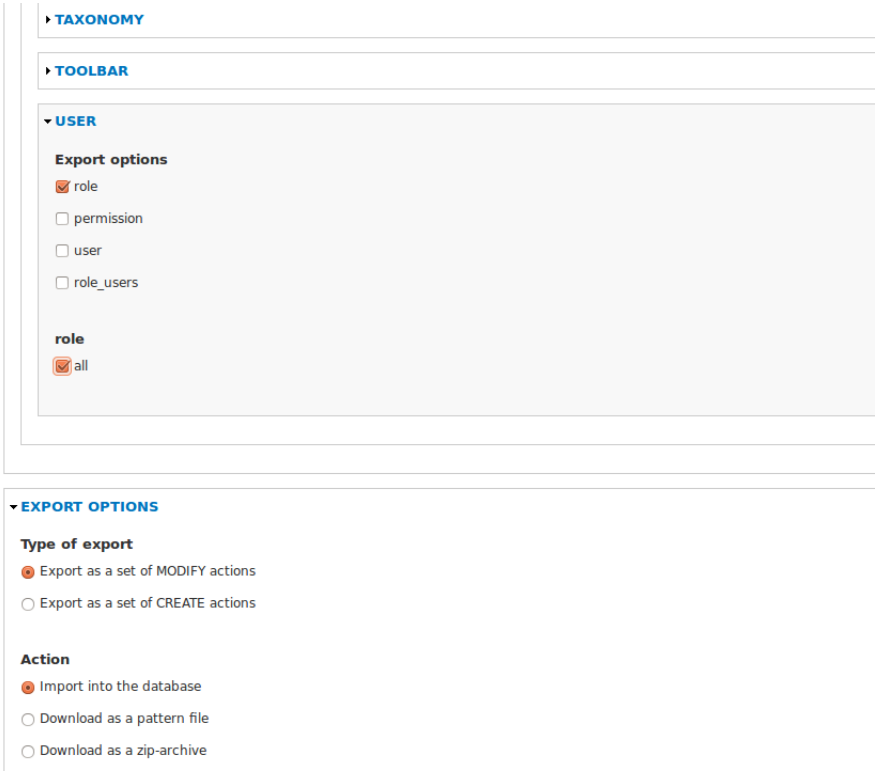

Figure 2.6: Detail of the new UI employed to export configuration automatically.

2.2.2 Syntactic and semantic analysis

The problems of naming collision and resolving dependencies between different entities have been addressed by several techniques employed by compilers while transforming the source code written in a programming language into a binary form. In the case of Patterns it was necessary to address a similar problem due to its inner nature of transforming a file storing a set of configuration into a set of values interpretable by the CMS. This is especially important while re-using patterns exported automatically as the ones described in the previous section, since the state of a website might not fulfil the requirements to make use of the configuration of a pattern exported by another website.

The new version of Patterns allows the components to implement two separated layers of syntactic and semantic validation, providing a very valuable feedback to the user that is going to run a pattern in case it is necessary to solve any possible conflicts. It was necessary to analyze and classify all these conflicts in order to establish a set of principles that ensure a pattern is syntactically and/or semantically valid. As a rule of thumb we will consider syntactic errors/warnings the ones related to wrong grammar statements, and semantic warnings the ones that would provoke execution errors related to the meaning of the pattern itself. The relationship is inclusive: a pattern semantically valid will always be syntactically valid, but not the other way around. The next subsection provides a set of examples of the possible errors and warnings we have currently discovered and classified.

```
info:
info:
  title: Create roles
  category: Users
  description: Creates a role for researchers and students
  version: 1.0
  core: 7.x
  author: QScience
  author_website: http://qlectives.eu/
modules:
  - user
role:
  -
    create:
      tag: role
      name: researcher
      fakeatt: 17

  -
    create:
      tag: role
      name: student
```

Figure 2.7: Example of a syntactic warning. The fake attribute "fakeatt" in this pattern will not be interpreted by the component, but it will not provoke any collateral consequences.

Syntactic warnings

They refer to wrong grammar statements that will not provoke any execution error. Examples of syntactic warnings are attributes that will not be used by that tag, or grammar mistakes in optional attributes—see Fig. 2.7.

Syntactic errors

They refer to wrong grammar statements that will provoke an execution error. Examples of syntactic errors are misspellings in required attributes or actions that do not apply on that component—see Fig. 2.8.

Semantic warnings

They refer to the meaning of the pattern itself, and they might provoke execution errors if they are not solved. Currently we have identified the following types:

1. *Already defined element:* This refers to the problem of creating an element that it is already present in the system and whose scope is within the component itself. E.g., while creating a menu that it is already stored in the system.

```
info:
  title: Create roles
  category: Users
  description: Creates a role for researchers and students
  version: 1.0
  core: 7.x
  author: QScience
  author_website: http://qlectives.eu/
modules:
  - user
role:
  -
    create:
      tag: role
      namenn: researcher

  -
    create:
      tag: role
      name: student
```

Figure 2.8: Example of a syntactic error. The mandatory attribute “name” has been misspelled as “namenn” and this will provoke an error during the execution.

2. *Element undefined:* This refers to the problem of trying to perform changes in an element that is not present in the system yet and whose scope is on tags created by the component itself. E.g., while modifying a system variable that does not existin the system.

3. *Unmet dependency:* This refers to the problem of creating an element that depends on the existence of elements generated by other tags that are not within the scope of that component. E.g., while trying to define a shortcut set (shortcut component) for an user that does not exist yet (user component), or define a permission (user component, tag permission) for a role that has not been defined yet (user component, tag role).

4. Remaining dependencies: This refers to the problem of deleting a part of the configuration that affects elements defined by other components. E.g., Deleting a content type leaves the nodes of that type as "undefined".

5. Not unique alias: This refers to the problem of modifying or creating an element that belongs to a set that identifies uniquely certain entity, being a similar constraint as the Unique constraint in Databases. E.g., Creating a new block whose delta and info values (alternative ways to identify it besides the Primary Key element) already exist in the system.

6. Inconsistent operation: This refers to performing an action that, although syntactically valid, Drupal core does not support. E.g., Delete a block that has been not defined through the Block component.

See Fig. 2.9 for an example.

2.2.3 Technical documentation

Patterns is a relatively big Drupal module and a significant effort has been made to provide a detailed technical documentation in order to encourage the support from the Drupal community. The documentation was officially released at Drupal.org, including a whole new section for contributors (`http://drupal.org/node/1822906`) that offers an outer to inner overview, in a similar way we do while debugging the code. It includes several subsections describing the execution flow for the most important functionalities, such as running a pattern, importing a pattern from several sources, etc. In order to make the documentation easy to follow a set of flowcharts were included, whose source files can be obtained from Github at `https://github.com/QScience/patterns-flowcharts`.

2.2.4 Patterns server

The role of the Patterns server is to act as a hub for sharing Patterns files among Drupal users. The key attributes we aimed for are: (1) simplicity, and (2) security. Given these goals, we built a DrupalToDrupal addon to upload patterns, and exchange information between clients and the Patterns server (see section 2.4). The client extension ensures transfer through a secure, encrypted channel where user credentials are

```
info:
  title: Create roles
  category: Users
  description: Creates a role for researchers and students
  version: 1.0
  core: 7.x
  author: QScience
  author_website: http://qlectives.eu/
modules:
  - user
role:
  -
    create:
      tag: role
      name: researcher

  -
    create:
      tag: role
      name: student
```

Figure 2.9: Example of a semantic warning. The pattern is syntactically valid, but in case the roles already exist, it will raise two semantic warnings of type "Already defined element".

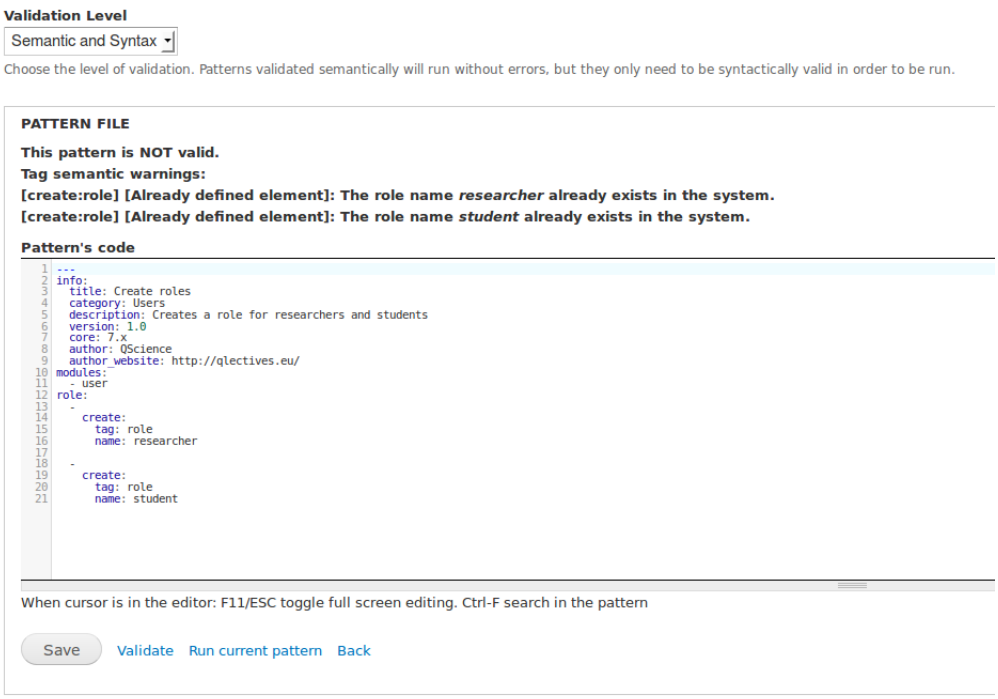

Figure 2.10: Example of a Pattern syntactically valid, but raising semantic warnings for the current instance—graphical user interface.

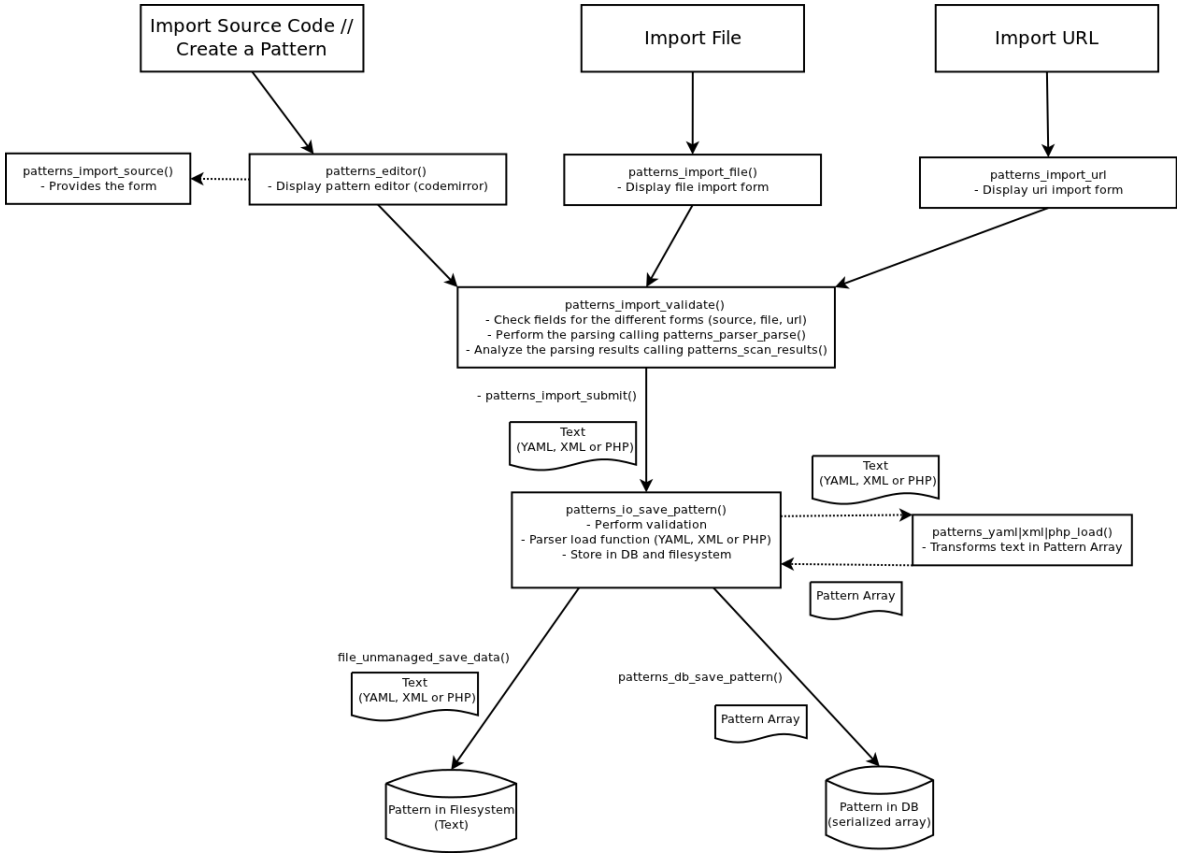

Figure 2.11: A flowchart describing pattern import processes.

verified each time an attempt to post patterns files is made. This way, a 1 to 1 association between patterns creator and patterns users, and statistics can be generated as described in section below.

Currently, the procedure to upload patterns is twofold. Patterns can be pushed to a user-specified server from the UI, or via the Drupal drush utility (command-line). We expect that experienced developers will find themselves more at ease with the command line utility, while new users—or site administrators—will find more useful the buttons integrated in the graphical interface. Interestingly, both the Patterns server code, and the client extension are released as open source modules for Drupal—patterns-server and patterns-client—and can be customised and adapted by other Drupal developers. At the moment of writing, the Patterns server site `http://www.drupal-patterns.org` is undergoing a period of beta testing, and it is expected to be launched in the forthcoming weeks.

Patterns server details

The Patterns server is designed as a Drupal module. For now, it can be downloaded from `https://github.com/QScience/patterns_server`. The main page of Patterns Server site is shown in Fig. 2.12. It has three parts: search box, pattern list, and pattern information. Users can easily post their comment on individual patterns. Through the search function (see Fig. 2.13), users can access each pattern file in an easy way.

The list of patterns shows the ten most recent pattern files (see Fig. 2.12 again). Each row of the table shows information for one pattern with six columns:

1. Category: the category that pattern file belongs to,
2. Title: the title gives a short description of this pattern. When user clicks on the link of title, the pattern information part will show on the bottom of this page which include all information and content of this pattern,
3. Author: the user who uploads the pattern,
4. Upload time: the time when this pattern was uploaded,
5. Download: download link of this pattern file,
6. Downloads: how many times the pattern has been downloaded.

2.2.5 Continuous integration on Travis-ci

In order to guarantee the fast development of high quality code, we have integrated our main code repository with the continous integration platform Travis-ci (`http://travis-ci.org`). Each time the code is pushed to the github repository `http://github.com/QScience/Patterns`, a battery of automatic SimpleTests is run on the continuous integration server Travis-ci. A history of previous builds is kept for reference and the results of the tests are emailed to the commit author. Given the distributed nature of the development of the QScience platform, the continuous integration proved to be an important milestone for the project. We are planning to extend the continuous

Figure 2.12: Front page of the patterns server.

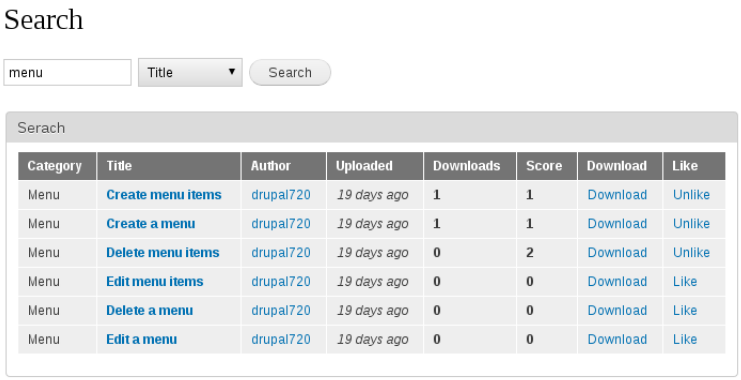

Figure 2.13: Search interface of the patterns server.

integration to all the other components of the QScience platform in the forthcoming months.

2.3 QTR module

The quality, trust and reputation (QTR) module was developed as a new Drupal module implementing our new QTR model for science. This model is now being developed in cooperation by the University of Surrey, University of Fribourg, and the Warsaw University. The main goal of the model is to process the data on online activity of scientists (by activity we mean writing a new paper, submitting a new blog post, commenting, and so forth) into quality of scientific objects (papers/blogs/meetings/...), reputation of individuals, and trust relations between individuals (see the deliverable D1.2.1 for details). We now implemented it into a Drupal module (`https://github.com/QScience/qtr`) which is available to use in any QScience instance. The QTR Drupal module also can be changed and adapted by other interested Drupal developers.

We have already used this module in a new testing version Econophysics Forum (`http://physpc140.unifr.ch/drupal-7.14/`) which will replace the old version Econophysics Forum soon. In our released new version Econophysics Forum, we initialize the QTR module as the default module of the site. The rank of papers is displayed in the right column under the tab "top value" in the Top Papers box (see Fig. 2.14).

There are three types of rank method, which are top value, most downloads, and most votes. The top value rank obtained from QTR results through analysis of users' activities in the Econophysics Forum, the another two methods are based on popularity methods. As a consequence, a user action can be either uploading, downloading, or viewing a paper. To obtain the dataset of interactions, we run-time automatically analysis database related to user interactions. The rank information can changed from time to time because updating the ranking information depends on the changes of this website. So QTR module can be run manually or automatically.

Among the three types of user access considered (uploading, downloading and viewing a paper), the first is obviously the more demanding, whereas the second reflects the user's interest in the paper much better than the latter. Other content type activities in the website also can be considered to calculate the result of QTR. Hence

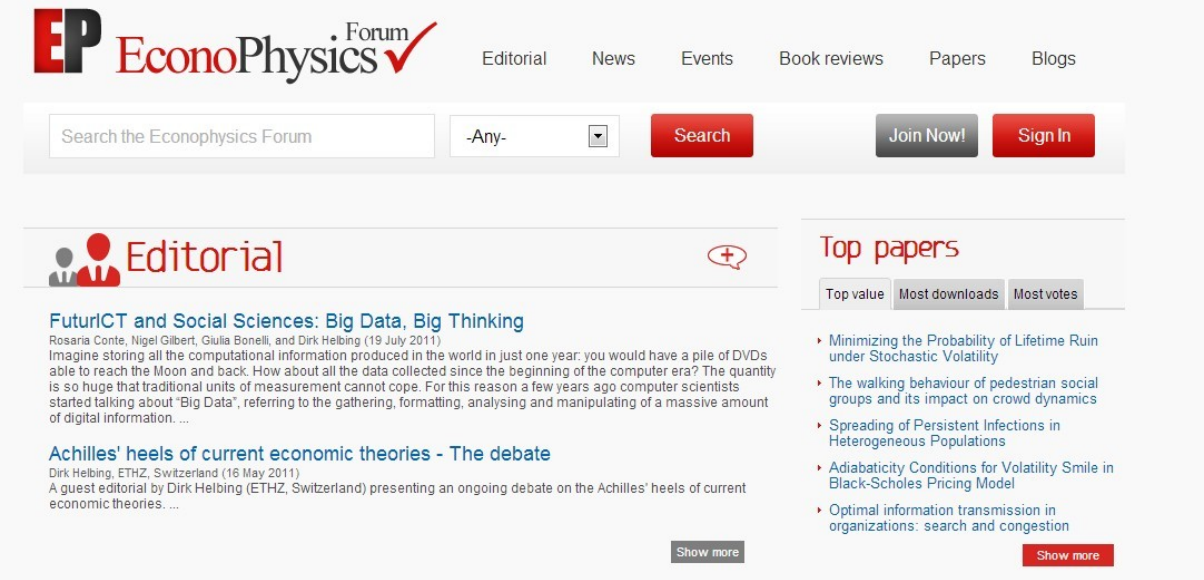

Figure 2.14: Papers scoring best in the QTR algorithm are now displayed on the Econophysics Forum front page.

we can associate to each action a different weight. Through simulation and data training experiments, we set $w = 1.0$ for upload actions, $w = 0.1$ for download actions and $w = 0.05$ for abstract view actions. Of course this is just a particular choice, which we consider as reasonable for Econophysics forum. Indeed, in order to make our developed module more flexible to apply to other QScience instances, the website administrator can configure the weight of the actions through the webpage (see Fig. 2.15).

We have currently set the default configuration of QTR module for Econophysics forum to obtain the best situation where the scores are distributed more evenly, top users have a non-negligible number of contributions and top papers have on average more citations than in the other settings. Every important parameter in QTR module can be manually configured for the specific requirement of the domain.

2.4 Building a QScience network

Drupal websites hosting QScience communities should be able to communicate with each other. Drupal natively supports XML RPC, a standard for remote procedure calls, encoding a call with XML and sending it over HTTP. Communication via XML RPC can be used almost similarly as a PHP-function-call, providing arguments to a remote function and getting back a return value. While being easy to use, neither authenticity nor secrecy can be guaranteed by these remote procedure calls. The Drupal-to-Drupal-module solves these issues. It allows to built up a QScience Network, a peer-to-peer network of QScience communities where authentic and secret communication can be reached using strong cryptography. Communities can establish friendship which allows the sharing of exclusive contents with particular other communities, a network of QScience communities is built.

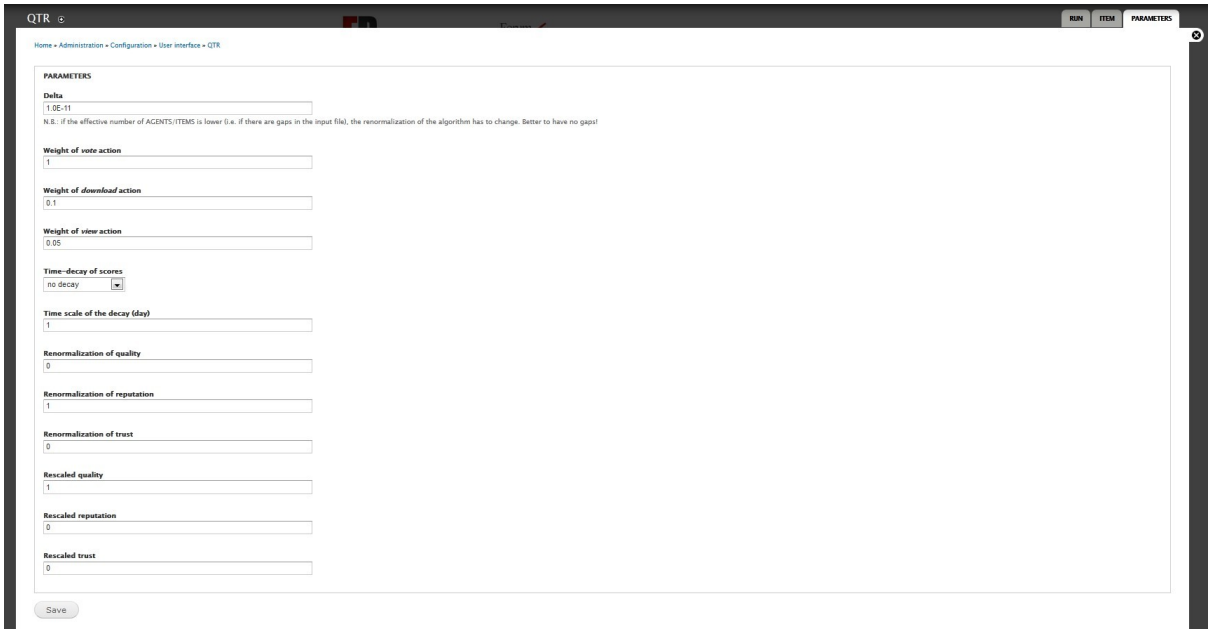

Figure 2.15: Configuration interface of the QTR module.

After having a working version of Drupal-to-Drupal and doing some testing of the communication part (being run on several servers in Switzerland, Hungary and China), we decided to re-factor parts of Drupal-to-Drupal to make the module easier to use in practice as well as reducing the amount of work necessary to maintain a running copy of Drupal-to-Drupal. However, this refactoring requires some structural changes in the module as well as overcoming technical and practical issues and is therefore very time consuming. Moreover, as part of the refactoring, a lot of tests are being done, not only tests that can be run offline, but also tests including several installations of the Drupal-to-Drupal-module. Beside that, we have implemented and further developed several add-ons and show cases for Drupal-to-Drupal.

The possibility to automatically share content between QScience communities is about to being implemented and tested. Therefore, we develop an add-on to distribute and synchronize databases among Drupal websites. This add-on will allow content being synchronized not only between Drupal instances that are directly connected but also among Drupal instances that are connected in the QScience network over several hops. After distributing content, this content remains accessible, i.e. over the QScience communities it has been shared with, even if the QScience community originally sharing it is temporarily or permanently offline. As a showcase of interest for the QScience communities, we will use this add-on to share publications as well as respective user ratings among the network of QScience communities using this add-on.

Another related use case is the sharing data of content types without previously distributing it. Certain data, e.g. a publication together with author information, description etc. can be marked as public. A search query by a user of one QScience community is then not only answered using the data which is locally available but also forwarded to other directly or indirectly connected communities which then send back their results for this search. Such a result could also include a link to download a

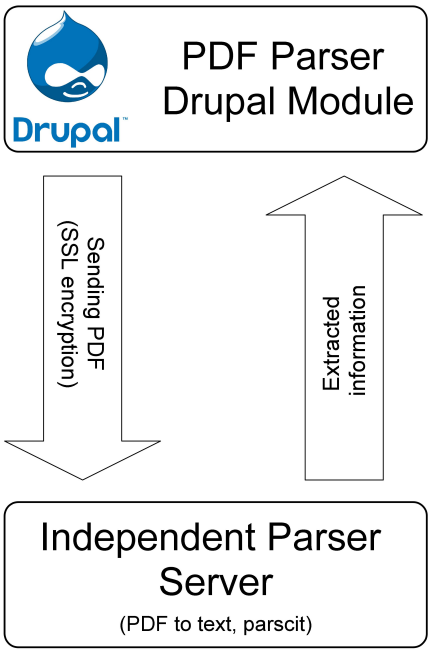

Figure 2.16: Work diagram of the pdf parser.

paper in case of the content type being a publication.

Finally, we wrote an add-on for the Drupal-to-Drupal module to publish Patterns, i.e. to share it with other QScience communities and therefore make it available to a large number of potential users. Note that a QScience community might not only provide the collected Patterns to its registered users but also make it available via public download. While the basic functionality of this sub module has already been implemented, we are about to extend it further to make it usable in practice. Such future functionality includes deletion of previously published Patterns, defining and resolving dependencies among Pattern. Finally, we want to track the evolution of Patterns: is a newly published Pattern based on previously published Patterns? How did a Pattern make its way from its first publication to the current server? This work is not finished yet but progress is expected in the next few months.

2.5 PDF parser

The PDF Parser module is based on the QScience Drupal module. The PDF Parser module's goal is to help uploading scientific papers. The working mechanism (see 2.16) is very simple: Once the PDF Parser module is installed and enabled, a "PDF Parser" menu is available at the main menu bar, where the user (page owner) has to send its own public key (which was generated automatically during installation). After sending the key, the page is free to use the parser server. This procedure is illustrated in 2.17.

After these processes if a user uploads a scientific paper, the Drupal server sends the uploaded, SSL encrypted PDF to the parser server, the server tries to extract this

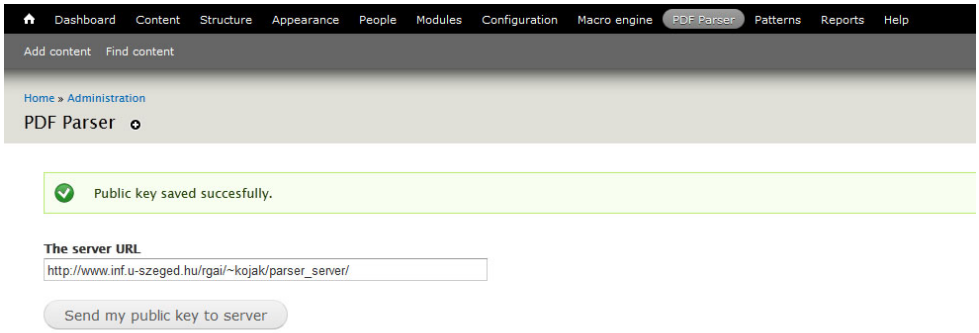

Figure 2.17: Setting the public key for the pdf parser.

information:

- paper title,
- paper abstract,
- authors,
- citations (including their authors),

and then returns them to the Drupal server (to the PDF Parser module). After accepting the results the server displays the extracted information (see 2.18), so the user can check and modify them if he wants.

2.5.1 PDF Parser server

The PDF Parser server is a Drupal independent server. Theoretically it could be any kind of server which can accept HTTP requests. The job of the PDF Parser server is very simple, the steps of processing are the next:

1. receive the PDF,
2. verify the PDF with the sender's public key,
3. convert the PDF to TXT with pdfbox (see `http://pdfbox.apache.org/`),
4. run parscit (see `http://aye.comp.nus.edu.sg/parsCit/`),
5. run some further heuristics on author names to extract them correctly,
6. collect the extracted information,
7. send back to the Drupal server.

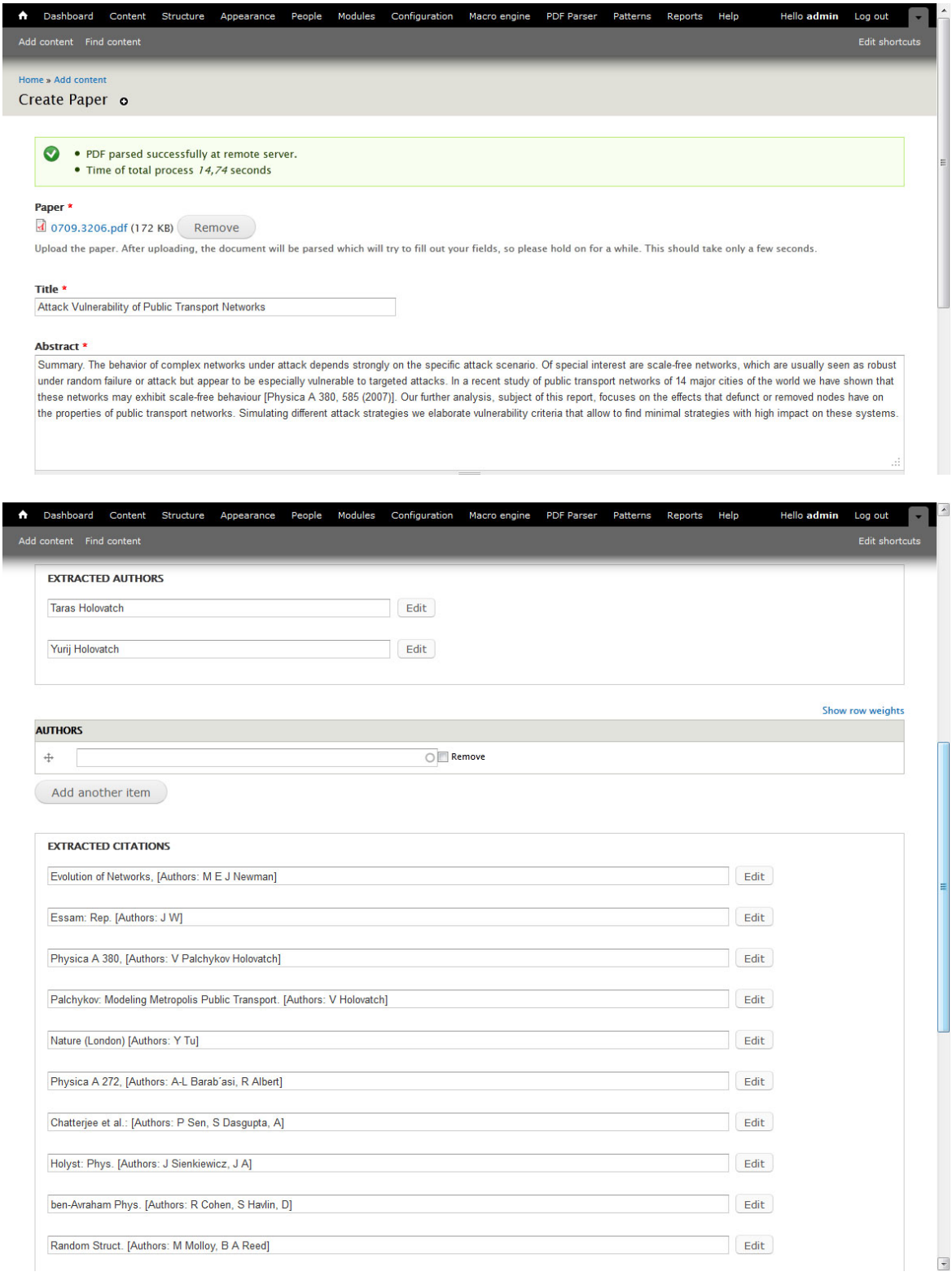

Figure 2.18: Extracted basic information (title and abstract—top) and additional information (authors and references—bottom).

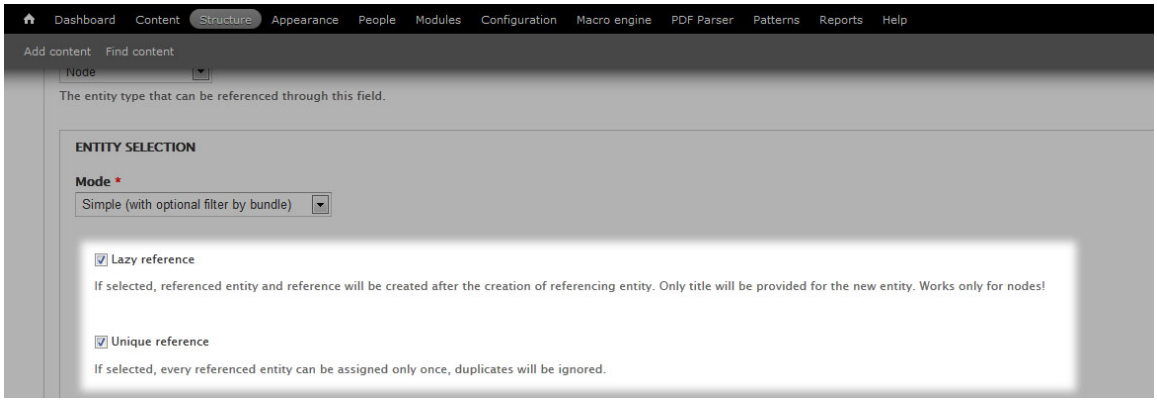

Figure 2.19: Setting of `lazy_reference`.

2.5.2 Future work

With respect to an eventual QScience module, there are some issues with the automatic removing of fields which is simply a bug that needs to be fixed. A major issue is to merge the `QScience master branch` with the QScience `reference_develop` branch. These two branches have diverged for some period because the main branch does not use the entity reference module which is the essence of the QScience module. With respect to "Lazy reference", to prepare a clean useful patch for the Drupal community, the `lazy_reference` feature must be tested and generalized as much as possible before sending it to a revision.

PDF parser is expected to be used by individual QScience instances in the future to improve their data mining ability. Its inclusion in Living Science (see Sec. 2.6) is also foreseen.

2.6 QScience Visual Search Toolbox

We introduced the QScience Visual Search toolbox in last year's report. It was described as an easy to use search and visualization tool, and the present year development continued along the same direction. Herein follows a brief presentation of the modifications and improvements that have been made.

First of all, the user interface has been completely redesigned. The most important change in this field is the new way the user interact with the web application. Instead of being window-based, the navigation system now uses a tabbed interface. The motivations for this change were esthetical, and based on user experience. In fact, with the new layout it is now easier to display the right information to the user without squeezing it into a small area. In order to achieve this result each action (Send Message, Export to CSV, LivingScience Search and Open a conference) of the Search Toolbox have been redesigned and refactored to fit best into the new architecture. The best way to describe the new user interface is with an example—see Fig. 2.20 for an illustration.

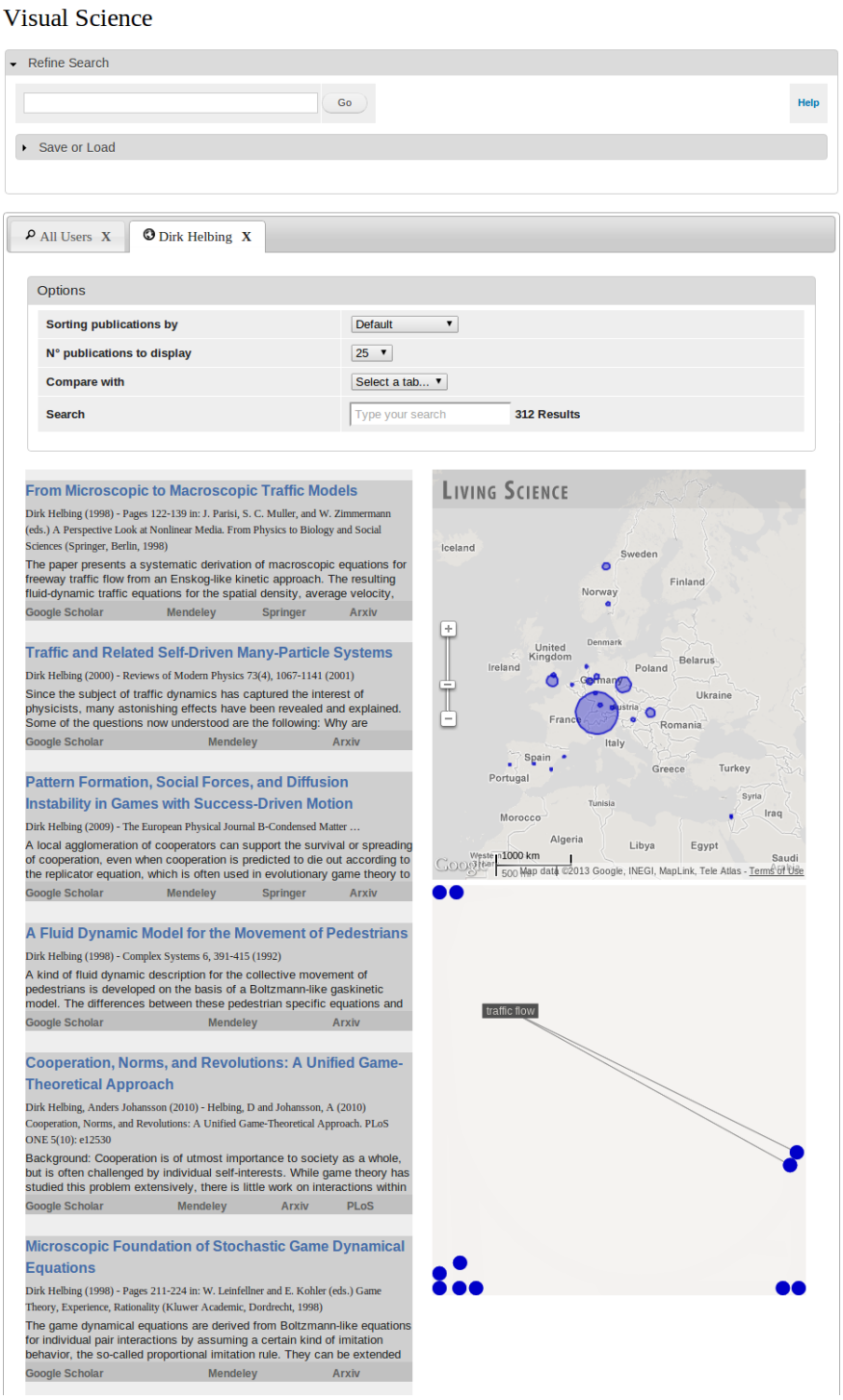

Figure 2.20: Tabbed interface showing the record of Dirk Helbing.

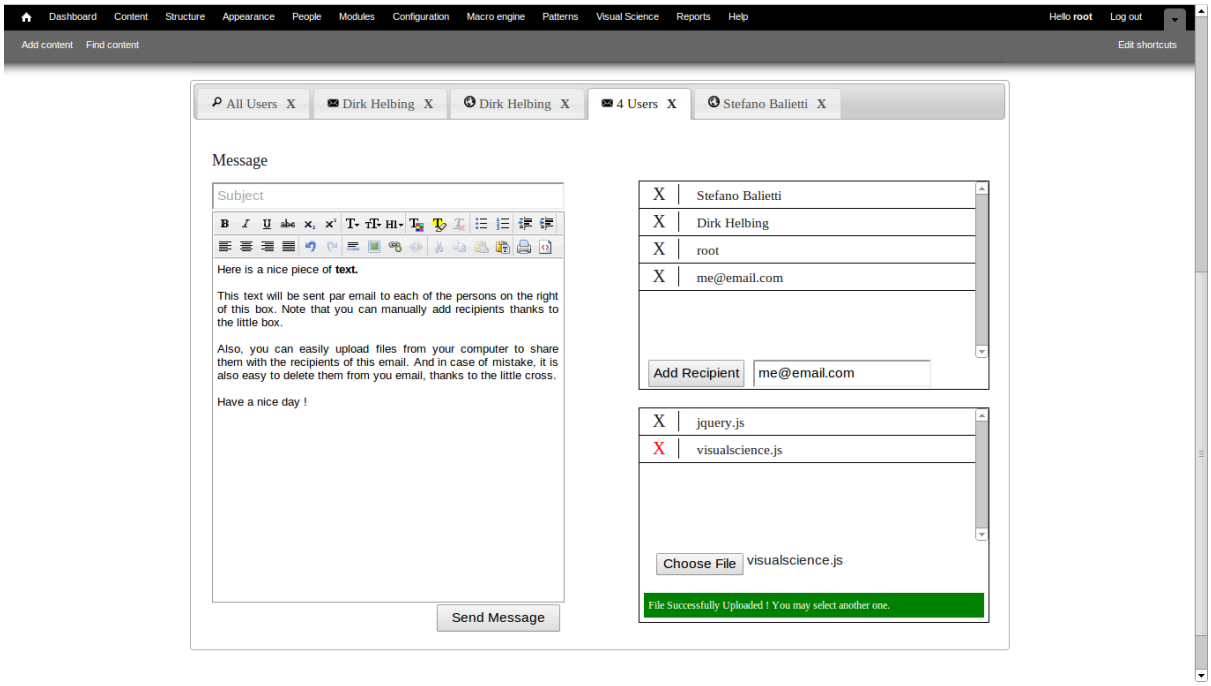

Figure 2.21: Message tab for users.

2.6.1 The VisualScience use case

Let's assume that a VisualScience user wants to collect information about a given scholar, and then to send him a message. First, he will arrive on a screen displaying a search bar. While typing the name of the professor, a few suggestions will be offered based on what he is typing (with an auto completion mechanism). Once the search is done a new tab will be opened presenting a list of possible target candidates. The user can now select the scholars of interests, and then choose one of the following actions: (i) Send a message, (ii) export the search results to a downloadable file, (iii) make a detailed "LivingScience Search", or (iv) invite them to a conference. In order to get more information about the selected person/s (see Sec. 2.6.3 below for a detailed explanation of this functionality), the user will click on the LivingScience button and a new tab will appear with the result of his search. From the same search tab, the user can click on the "Send a Message" button, and a new tab will appear offering the desired functionality (see Fig. 2.21). In this message tab, the user will be able to write his message, attach multiple files, and add other custom recipients. Once he is done, he can click on the Send button and an email will be sent to each of the entered recipients.

2.6.2 The refactoring

The refactoring of the VisualScience module was targeted at improving its general usability, however some new features have been added. For example, the attachment to the email message is now kept on the server, and inserted as a link inside the email message. The full procedures involves the following steps: while uploading the attachment, the system will create a public copy of this attachment on the server, copy

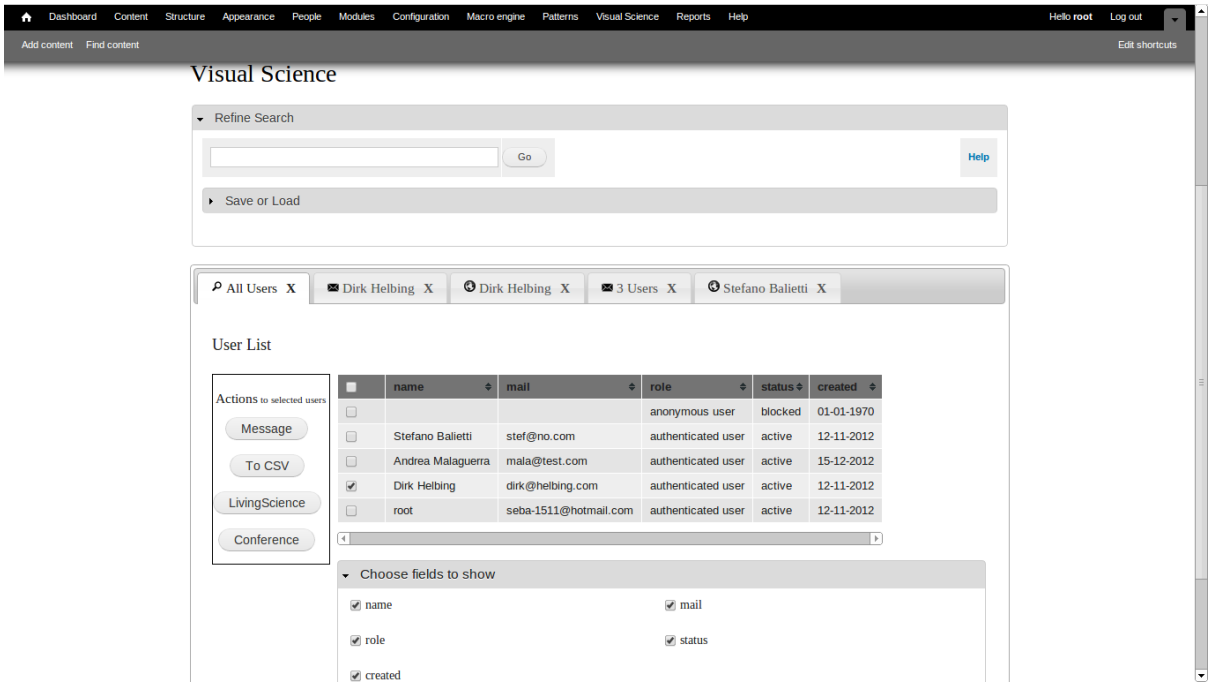

Figure 2.22: Living Science search tab for Dirk Helbing.

it into a private directory, create a new database entry for the private file, link it with a set of users that are granted the privilege to access it, and will finally delete the public copy of the attachment. All these operations are done in the background, and the procedure is completely transparent to the user.

The performance of the whole application has also been improved. The easiest way to manage the data of the tabs was to use a database on the client side. The NDDB library (`https://github.com/nodeGame/NDDB`) was the best option here. Furthermore, we needed to manage the client side rendering of the data. For this, another JavaScript library was chosen: Handlebarjs (`http://handlebarsjs.com/`). Those design choices were motivated by the usability, and the good reputation of both libraries. Finally, the whole JavaScript code of the VisualScience module has been refactored, and reorganized in several independent modules, following MVC-like pattern. To sum up, the code is now easier to maintain and much more efficient than before.

2.6.3 Living Science implementation in the Visual Search Toolbox

The Living Science search engine was already introduced in the last year's report: It is a search engine that presents detailed information about publication records on a person basis. The integration has gone further, and all the changes were made by working side-by-side with the LivingScience team.

The first change that had to be made was to implement the LivingScience result screen into the tabbed interface described above. Separated by tabs, the information is now easier to understand (see Fig. 2.22). While implementing this new layout, small improvements have been added. For example, local searching and sorting of the results from a LivingScience query is now possible, and easy to achieve through a dedi-

cated menu.

Another important improvement is represented by the "Compare With" tab. When the user wants to have a side-by-side view of two results and get a visual comparison of productivity statistics of the scholars, he can do so by clicking the "Compare With" menu in the tab options. He will get informations such as the year of activity, the top publications, co-authors of both authors, and so forth (see Fig. 2.23).

Some effort has been and is currently being dedicated to the name-disambiguation problem of the data received by Living Science. At the moment a set of simple heuristics have been implemented to work locally. However, some theoretical work currently being done at ETH Zurich and it is probably going to be implemented in the next code iteration for LivingScience or QScience.

2.6.4 Future plans

The roadmap of the future developments of the VisualScience module aims at creating an even more intuitive, and involving user experience. In fact, users will be not only passive receivers of the information, but will be able to perform inline editing of the data, and to share it immediately with the Living Science server or to QScience instances via the DrupalToDrupal module.

2.7 Living Science

Living Science is an open and up-to-date database for publication metadata for all research fields. It makes use of other publication search engines and collects information about publications, authors, journals and affiliations. While Living Science offers its services at a public website (`http://livingscience.ethz.ch`), all features are also provided as a JavaScript API to QScience instances.

2.7.1 Living Science database

Previously, for every user-initiated search we were solely relying on external search providers (Mendeley, Springer, Google Scholar, Arxiv, PLOS, Amazon) to get lists of publications for further processing. The introduction of an own local database potentially gives us faster (less than one second instead of about ten seconds) and more relevant results and enables the implementation of algorithms which require a full dataset of publications. We can add more articles from other search providers whenever new results from user-initiated searches arrive (see Fig. 2.24). As of now, the database contains about 8 million publication entries, collected from other free online databases.

For storing the publication data, we chose a graph database (Neo4j), a relational database (MySQL) and a search index (Lucence). The graph database and the relational database store the data completely redundantly. Read operations shall be very fast, so we like to have the flexibility to use the data structure which best fits to a certain query. Graphs can perform much better on local queries, e.g. generating a list of

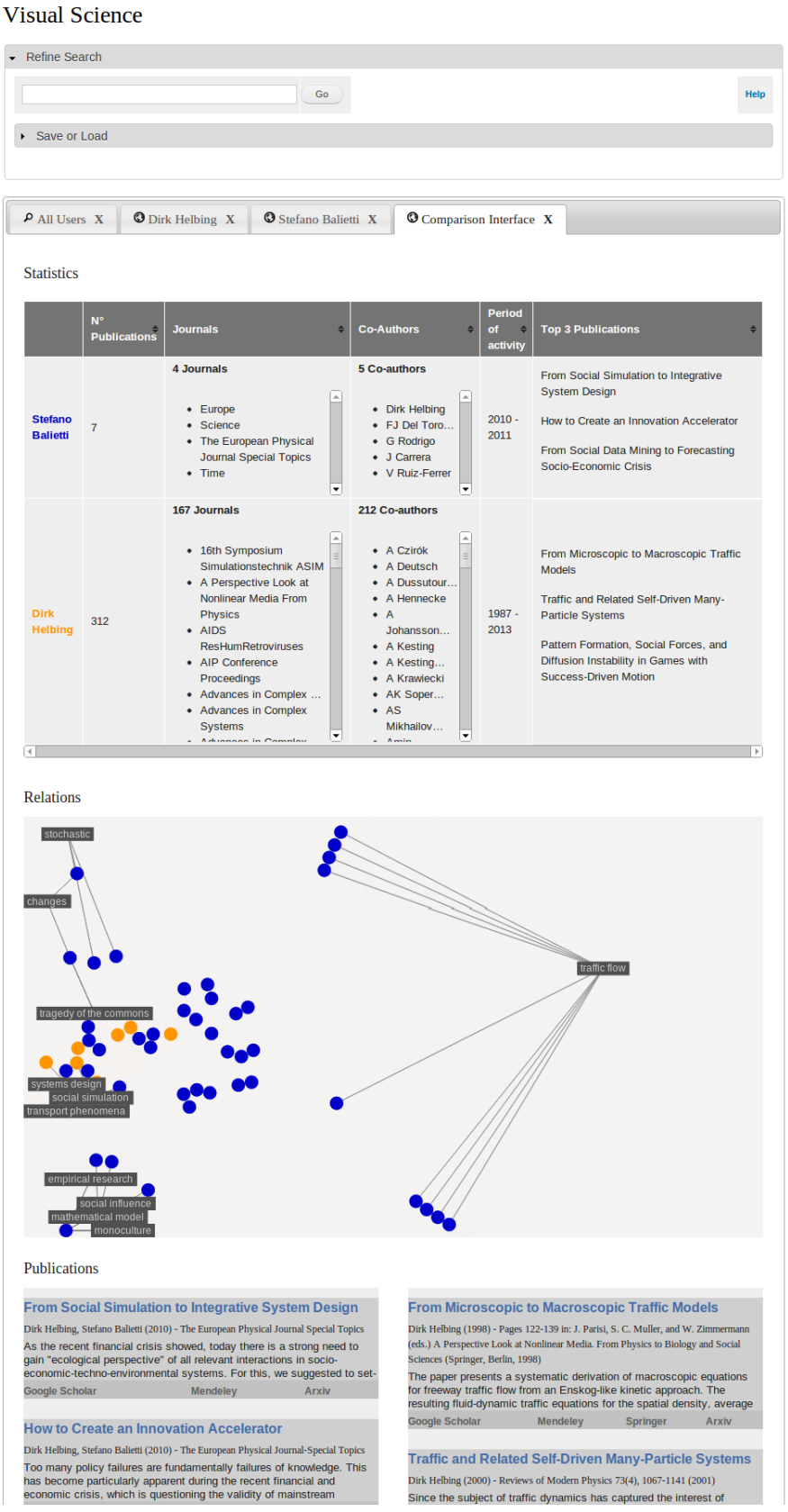

Figure 2.23: Living Science comparison of two researchers.

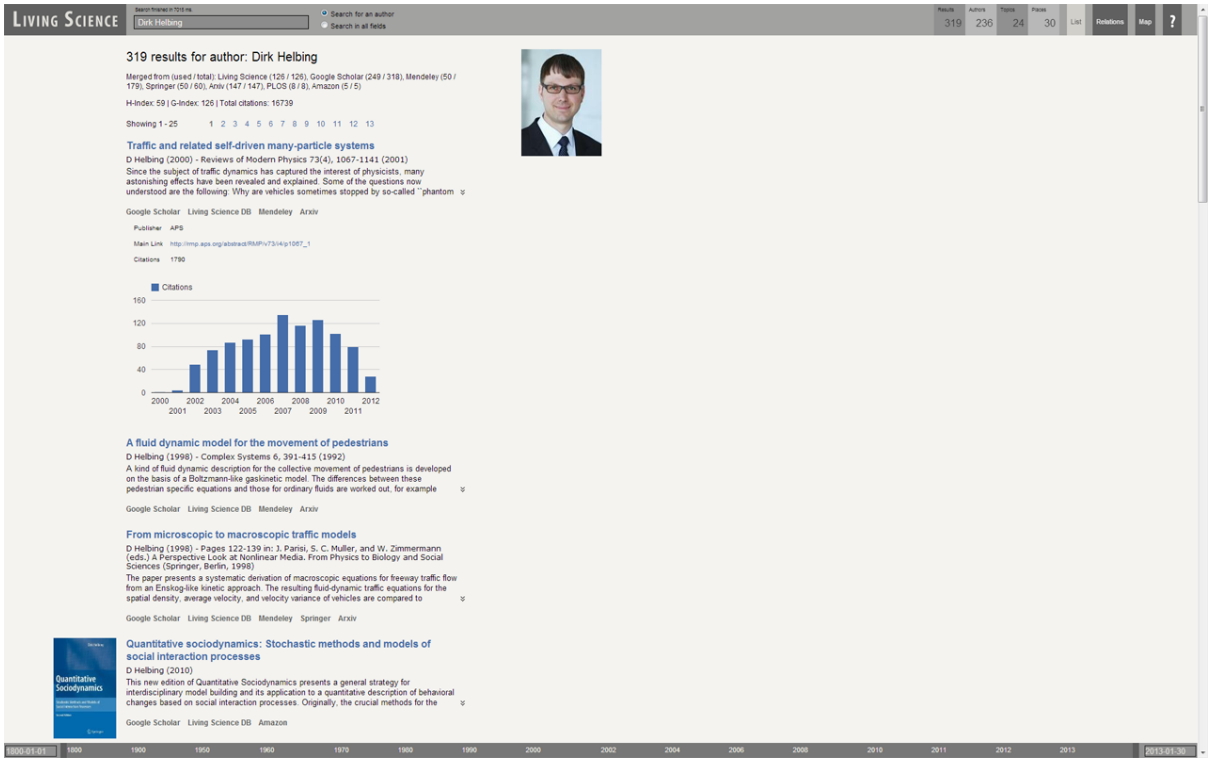

Figure 2.24: Publications search within internal Living Science database merged with results from external search providers.

collaborating authors or recommending publications that were viewed by users with similar viewing statistics.

2.7.2 API improvements and new features

The JavaScript API now allows for combining several search results. The world map and the relations graph visualize them with different color codings. Some conflicts related to the CSS and HTML DOM events between an integrating webpage and Living Science user interface parts could be resolved. When searching for an author, statistics like H-Index and G-Index are calculated based on the available information of the search result. A revised relations graph displays the similarity of the found publications by computing matching keywords found in the title and abstract (see Fig. 2.25).

2.7.3 Future work

We want to allow users to edit the Living Science database. Everybody shall be able to contribute by correcting mistakes, disambiguating author names, adding new publications etc., to get the most complete open database. For editing operations, we need to clarify means of user authentication. Up to now, QScience instances only ask for information provided by Living Science, but we would like to have a bi-directional communication. QScience shall act as a search provider, i.e. as a source of publication metadata for Living Science. The graph-based database and a potential collection of

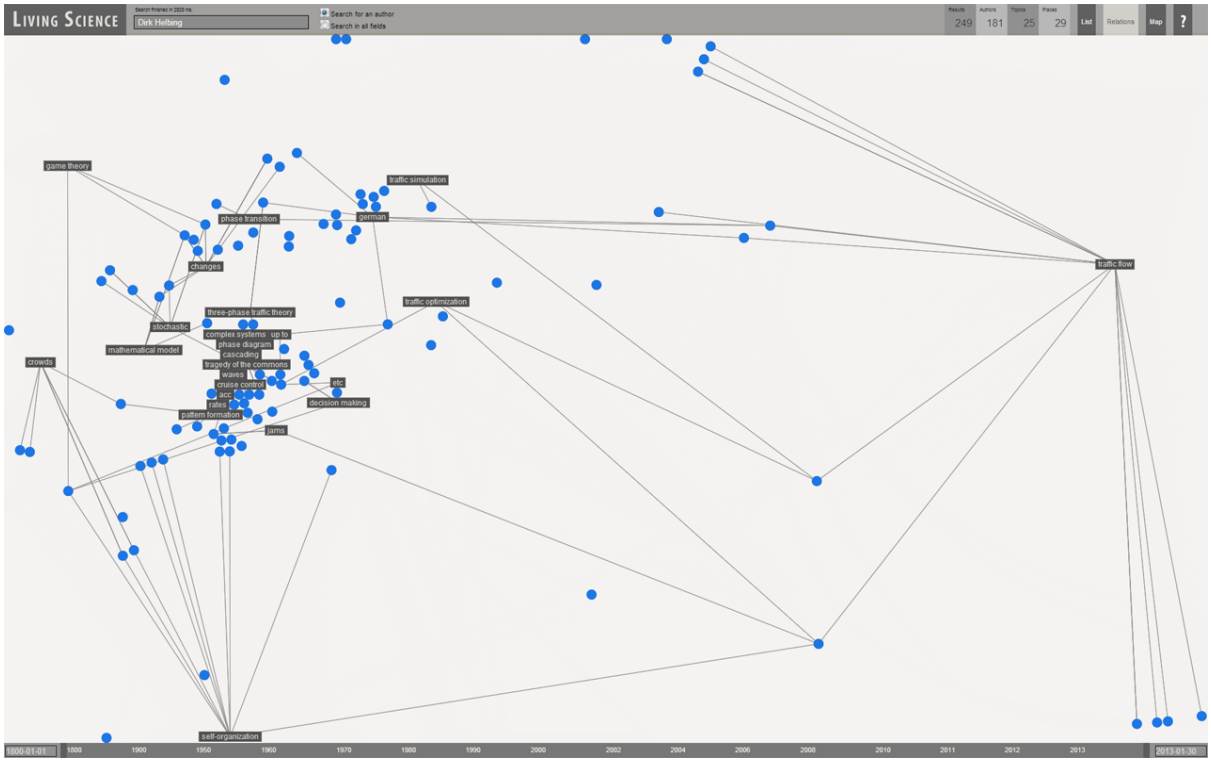

Figure 2.25: Interactive publication similarity graph constructed by keyword analysis.

usage data open possibilities for new queries like getting similar publications, measuring author performance and the detection of trends. Besides publication metadata, we are also interested in collecting and analyzing full texts. PDFs are often available from university websites and free online journals. Here, we could build on the existing QScience PDF parser.

2.8 Dissemination

A significant effort has been made in terms of dissemination of the module Patterns in the Drupal community by participating in presentations and informal meetings in several Drupal events (known as DrupalCons and DrupalCamps):

1. DrupalCon 2012 (Munich, August 2012)
    - Drupal Automated Deployment on Steroids: Patterns 7.x (`http://munich2012.drupal.org/content/drupal-automated-deployment-steroids-patterns-7x`)
    - Drupal and Academia (`http://munich2012.drupal.org/content/drupal-and-academia`)
2. DrupalCamp Spain 2012 (Madrid, October 2012)
    - Automating Drupal Development with Patterns (`http://2012.drupalcamp.es/es/sesiones/automating-drupal-development-patterns-38`)

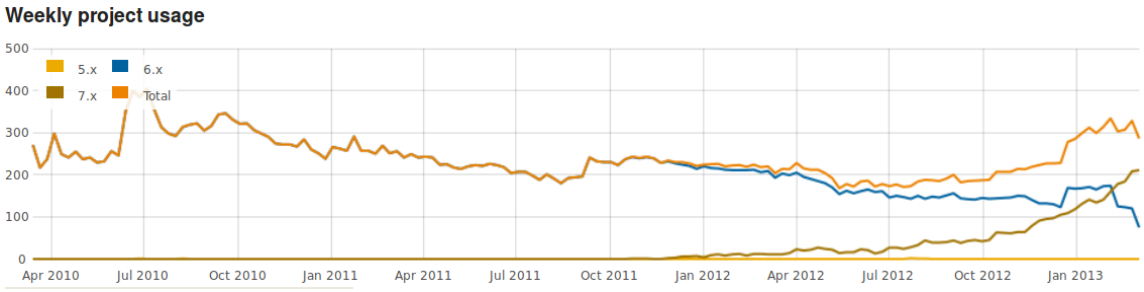

Figure 2.26: The number of total and per-version number of Patterns site installations as of 12th March 2013: Installations of 7.x versions grow rapidly.

3. DrupalCamp North West UK 2012 (Manchester, November 2012)
    - Informal meeting on Patterns 7.x

A new release of the Patterns module including code refactoring and several bug fixes was made available in November 2012 (see `http://drupal.org/node/1840318`). A new release including the changes described in various sections of this deliverable will be officially released in March 2013. The number of installations (online sites using Patterns module) of the branch maintained by QScience (7.x) has grown more than 400% over the last six months, showing a positive response from the Drupal community. The statistics (updated on a weekly basis) can be checked out at `http://drupal.org/project/usage/patterns`. We include here a snapshot from January 2013—see Fig. 2.26.

2.8.1 Future work

QLectives will be present in the DrupalCamp UK 2013 (London, March 2013) where our session on Automating Drupal Development with Patterns has been just accepted (`http://2013.drupalcamplondon.co.uk/`).

An official Drupal core initiative (see `http://groups.drupal.org/build-systems-change-management/cmi`) has been established to tackle the issue of Configuration Management as part of the core for Drupal 8 (which is expected to be released in August 2013). We established contacts with Greg Dunlap (leader of the initiative) and Alex Pott (one of the main developers), as well as with Mike Potter (main developer of the module Features `http://drupal.org/project/features`) in order to analyse and discuss the contribution of Patterns to the future of the Configuration Management in Drupal 8.

2.9 CRESS migration

During the next months, two new instances based on QScience will be created: CRESS (Centre for Research in Social Simulation at the University of Surrey, `http://cress.soc.surrey.ac.uk`) and the QLectives Project web site (`http://www.qlectives.eu/`). It will be necessary to carry out a whole cycle of development (requirements gathering,

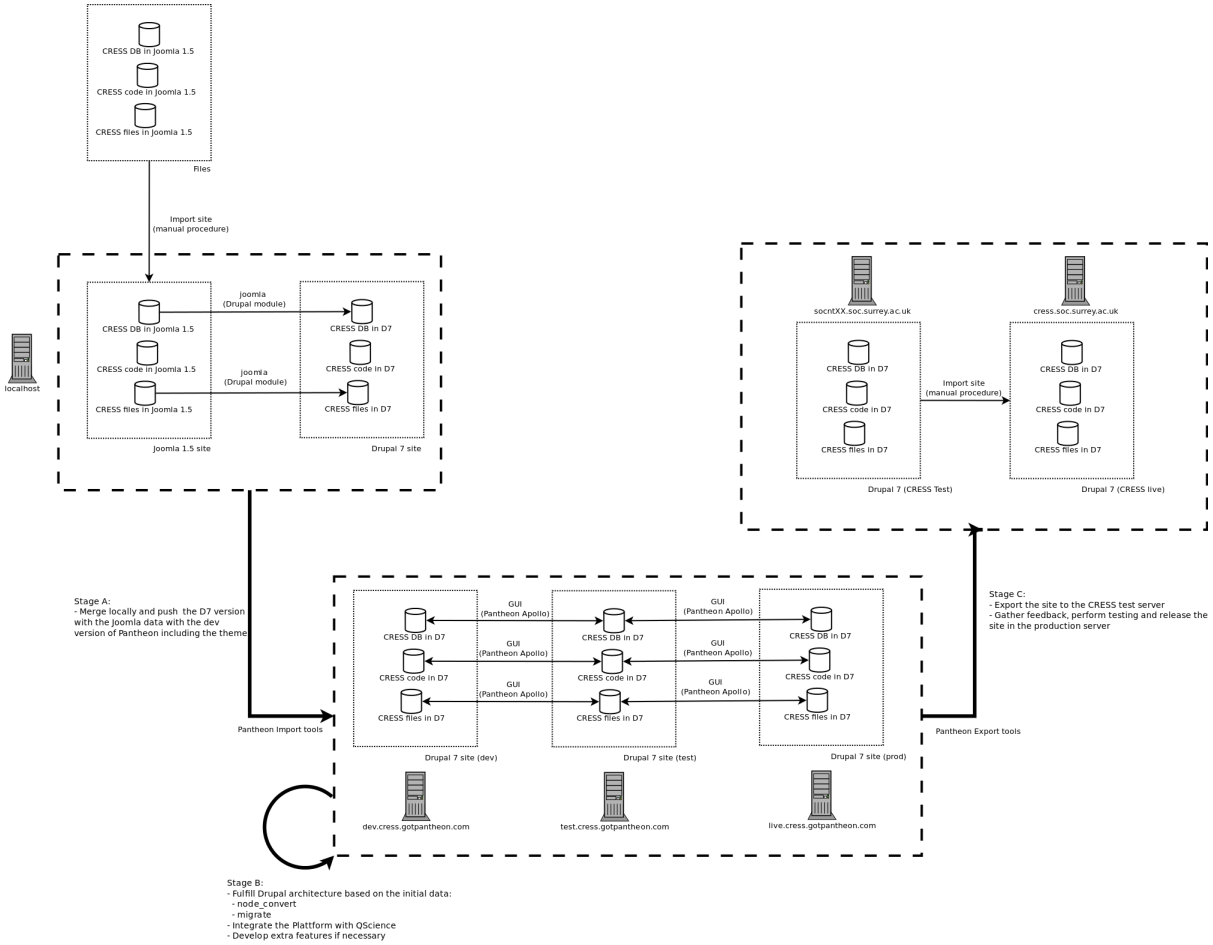

Figure 2.27: Migration plan for the CRESS web page.

analysis, design, implementation and testing) as well as performing a migration of the data from the systems currently employed to run the websites. A set of patterns will be generated and shared in the public Patterns server to allow other potential external users to utilize and adapt them.

Migration planning of the CRESS web site has already begun. The migration and development will be performed in three stages:

Stage A: Merging the current version of the theme created by Sam Sulaimanov with the local version imported from the current site running in Joomla.

Stage B: Main development phase, including adapting the current data to the new Drupal architecture, refining the theme, integrate it with QScience, add new features if needed, etc.

Stage C: Deployment to the university test and production servers.

In addition, Fig. 2.27 summarizes the most important interactions.

Chapter 3

Conclusions

QScience has achieved considerable progress in year 4 of the QLectives project. The first instance QScience is just about being launched publicly (replacement of the current Econophysics Forum) and two other are scheduled to follow. Besides, considerable effort has been invested in the development of the fundamental QScience infrastructure such as the Patterns module for Drupal 7, the central Patterns server, Drupal to Drupal communication, and various science visualisations. We have also put emphasis on dissemination which is based on public launches of QScience instances and QLectives presence at Drupal meetings where we push forward the idea of automated site management through Patterns.