GeoSoft 2, WS1516
Publishing interactive scientific papers

Marius Appel, Daniel Nüst, Edzer Pebesma

February 2, 2016

1 Introduction

Besides discovering new things, scientists spend a lot of time communicating their findings and opinions. Much of this communication is done by writing and publishing scientific papers. The process of publishing papers has not changed much in the last 30 years. Besides being digital, it does not take advantage of today's technical solutions to include interactivity into the paper reading experience. In this project a large scientific publisher wants to revolutionise the way scientific papers are reviewed and published.

1.1 State-of-the-art

Today, scientific publication dominantly concerns the creation of static documents that provide no more interaction than the printed paper articles as they were 100 years ago. Although many papers can be read as HTML documents, the figures in the paper typically do not afford any interactivity with the underlying data, despite the large amount of technical possibilities, and potential increase in understandability as well as chance to collaborate with others. Part of the reason lies in the fact that scientists are conservative and do not like to provide their data for readers to interact with, part of it is a conservative attitude of the publishers who have fixed workflows that are difficult to modify (see e.g. Elsevier's editorial system, which is used for around different 2000 journals).

The workflow now is:

1. author submits article to journal by uploading e.g. LaTeX or ODF sources, bibliography, and figure originals
2. EES compiles this into a pdf, and after technical check passes it on to journal editor
3. editor reads the journal, and if approved passes it on to reviewers
4. reviewers evaluate the paper, and advice editor

5. editor decides: reject, revise, or accept, and corresponds to the author
6. in case of accept: paper gets published (PDF, HTML, paper) in the journal

1.2 Project goals

The goal of this project is to create a system that allows scientists to prepare and submit papers that include interactive figures, and allows publisher to distribute these papers on the internet.

We make the following assumptions:

1. scientists are not programmers, they know little about web programming or JavaScript,
2. scientists are clever enough to prepare documents using LaTeX,
3. scientists are interested in new publication forms, and
4. publishers have an established system for the review process that has a public API that can tell if a paper, identified by its id, is publicly available or not
5. the interactive paper will be read in a web browser.

Based on this platform, a publisher can provide a new level of quality to readers of its journal and consolidate its market position by spearheading the development of the next generation of scientific papers. To demonstrate the usability of the platform, the customer provides a collection of scientific papers with specific datasets and features. The system must support these documents yet be designed in an abstract fashion that allows to transfer the solution to other scientific domains or to include other means of visualisation or data retrieval.

The new system must take into consideration several target groups:

- scientists who want to publish a paper and are users of the platform
- scientists and the interested public who reads the publication online
- publishers who operate the platform
- editors and reviewers, who are part of the publication and peer review process

Further requirements are listed in the remainder of this paper.

2 Requirements

2.1 Functional requirements

The main functionality of the platform is submitting and reading interactive scientific articles in a web-based platform.

2.1.1 Basic functionality

/LF0010/
The web-based system must provide a landing page displaying a list of available publications.

/LF0020/
Available publications must be displayed on the landing page in order of publication, newest first.

/LF0030/
A publication and all related resources must be available under a globally unique URL ("publication package download").

/LF0040/
The platform must support the submission and interactive illustration of example publications provided by the customer.

2.1.2 Article submission

/LF1010/
The customer does not want to store sensitive user information including passwords. To prevent anonymous fake submissions, users must authenticate using external authentication mechanisms (i.e. OAuth 2.0) before uploading publications.

/LF1020/
The platform must accept publications written in LaTeX.

/LF1030/
Data must be included within the paper or self-contained delivery package and the following formats must be supported:

- GeoJSON files for vector data
- GeoTIFF files for raster data
- RData files for time series (as zoo / xts objects) and spatial data (as sp objects)

As specified in section 2.1.3, the platform must provide functionality for interactive exploration of such datasets. The design how to link publications and their data remains up to the contractor.

2.1.3 Interactive reading

/LF2010/
Publications must be completely readable in a standard web-browser. This includes static text, figures, and tables as well as interactive figures that are generated from uploaded data.

/LF2020/
The system must be able to visualize time series and spatial data (see /LF1030/ for required formats).

/LF2030/
Readers must be able to interactively zoom and pan datasets (time series, spatial vector and raster data).

/LF2040/
Readers must be able to change the following visualization parameters:

- Basemaps for spatial data
- Axes scale for time series
- Color schemes
- Selection of displayed layers or series, if multiple are given

2.2 Non-functional requirements

2.2.1 Transferability

The user interface must support all popular web browsers, including browsers on mobile devices, with a similar user experience, while at the same time adapting the layout and features to the respective device. The contractor must report on the tested platforms.

2.2.2 Maintainability

The system must be published as open source under an OSI-approved license according to the license requirements and taking into account the licenses of other used software. The contractor must ensure license compatibility.

An established build and configuration system as well as dependency management must be applied for the used Programming language, e.g. Maven or Gradle for Java, or Grunt/Bower for JavaScript. According documentation on building, configuring and installing the system must be provided in Markdown-formatted documentation files using the appropriate markup for lists, links, etc.

2.2.3 User friendliness

The system must support intuitive use to the extend that targeted groups can use its main (non-bonus) features without further documentation. Common practices for web-based user interfaces and interaction paradigms should be applied.

2.2.4 Performance

After selecting a publication from a list, the next page must be displayed within 1 second to allow users to stay focused on their current train of thought. Complex page contents may be loaded asynchronously. The interactive visualisations

must react within 0.1 seconds to give a user the impression of direct manipulation.[1]

2.2.5 Boundary conditions

The system developed by the contractor must be given an appealing product name with high recall value. The product name is used throughout the source code and documentation of the system.

The language of the user interface is English.

One **bonus feature** from the list below, or own ideas for features, must be completed. Own bonus feature ideas must be submitted in due time and formally accepted by the customer. The chosen feature must be mentioned clearly in the project documentation and underlies the same non-functional requirements as regular features.

2.3 Extended features

Each extended feature may only be implemented by at most two contractors. *First-come, first choice.*

/EF10/
: Provide a way to assign and manage persistent identifiers (DOIs) to data sets which are included in the papers. Users should be able to explore DOIs interactively without leaving the application and at least one piece of external information for a DOI must be included in the system based on a third party resource.

/EF20/
: Users have the ability to interactively change the projection (coordinate reference system) of maps at runtime. This allows users to select the appropriate display for regional datasets, e.g. data covering the arctic. This feature shall be documented and tested with appropriate test datasets.

/EF30/
: Make the interactivity reproducible: Users must have the ability to communicate a particular state of the interaction (e.g. zoom level, focus, selection) to others. The sharing feature must support email as well as social networks and correspond to common interaction paradigms.

/EF40/
: Make the analysis changeable: A User must have the ability to change selected input parameters so that she can experiment with the underlying algorithm without downloading the data itself. The changes trigger a change of the visualizations. If combined with /EF30/, the analysis parameters must be part of the sharing feature.

[1] Source: http://www.nngroup.com/articles/powers-of-10-time-scales-in-ux/

/EF50/
Support full text search for the publications, including literal matching (i.e. *"match-this"*) and at least one search term combination (e.g. 'and', 'or', 'not'). The search results must be sortable by author name, author affiliation and publication date, and used search terms must be highlighted in the results. The complete text of the publication, including metadata, names, references, abstract and so forth, must be included in the search.

/EF60/
Interactive references: use at least two external resources to make the list of references explorable without leaving the application. External resources can be journal pages, social networks, media sites, science networks, etc. A user must be able to save or share a specific reference in combination with the currently viewed interactive paper for later reference.

/EF70/
Interactive summary statistics: implement the computation of simple summary statistics (mean, standard deviation, variance, quantiles, min, max, histogram) based on interactive selections of data subsets by the readers.

2.4 Bonus features

A contractor may choose to implement an additional extended feature as a bonus feature, "Sternchen-Aufgabe", which will be taken into consideration appropriately when the work is evaluated. The chosen feature must be mentioned clearly in the project documentation as **/BFxx/**.

3 Example papers

1. plotKML: Scientific Visualization of Spatio-Temporal Data (code and data)
2. Meaningful spatial prediction and aggregation
3. Vaclavik, T., Lautenbach, S., Kuemmerle, T., Seppelt, R. (2013): Mapping global land system archetypes . Global Environmental Change 23(6): 1637-1647. Online available: October 9, 2013,
   DOI: 10.1016/j.gloenvcha.2013.09.004.
   (data for plots and maps, i.e. after analysis, will be provided as RData files)

Further papers may be provided by the customer throughout the execution of the project.

4 Delivery contents

The delivery consists of the following subproducts:

1. commented source code on GitHub in one repository (delivery in more than one repository must be formally accepted by the customer)
2. installation documentation
3. user documentation (can be integrated into the product)
4. feature documentation (using screenshots, referencing requirements from this document)
5. operational installation on a server (provided by the customer) including installation protocol

5 Acceptance criteria

The acceptance criteria encompass the fulfillment of all functional and non-functional requirements as well as the complete provision of all items listed in section 4.

6 Seminar Procedure

6.1 Learning objectives

1. make project management experiences, including group work, task management, coordination
2. learn to work for a customer, in a process similar to typical business procedures
3. learn to work with established software projects and re-use libraries where appropriate
4. develop an open source project
5. deepen knowledge and skills in software development (e.g. collaboration platform GitHub)

6.2 Timetable

The seminar consists of five phases spread over the whole winter semester and the prior semester break.

Phase	Content	Dates
Initial training	get to know a relevant topic and prepare it for the fellow seminar members in a handout and presentation (15 mins presentation + 5 mins discussion lead by presenters); customers present requirements	semester break & first two meetings
students are split up in several groups and work on the same project		
planning	project groups create concepts and plans for the implementation of the Lastenheft	two weeks/meetings
implementation	project groups implement their solution, revising documentation and concepts along the way; special phases: project peer review (around christmas/new years), pre-release (three weeks before final presentation)	about 10 weeks
presentation	project groups present their solution to to the customers and peers	one meeting beginning of February

6.3 Efforts and grading

9 ECTS correspond to 270 working hours. Minus 16 hours for preparing the initial presentation, that leaves 254 hours over 15 weeks, which is an average workload of **16.9 hours per week** for the students.

What	Work load (hrs)	Grade contribution
initial training (talk preparation and attendance)	24	10%
planning (Pfilchtenheft, Zeitplan)	34	15%
implementation (incl. technical documentation, peer evaluation)	200	65%
presentation (incl. personal documentation)	12	10%

Grading is based on code contributions on the software development platform GitHub. Therefore, we require each student to contribute under their own GitHub account. A fork & pull development model is highly encouraged because it allows to list the relevant pull requests in a final personal report.

7 Topics for the first series of seminars

1. What is scientific reproducibility?
2. Publishing platforms 2.0
3. Web technologies
4. JavaScript web application frameworks
5. Web interfaces for time series
6. Web interfaces for maps
7. Geo web APIs
8. cross-platform development and touch interfaces in the browser
9. Modern websites
10. Collaborative software development
11. Databases and cloud services