






An Analytics Tool for Exploring Scientific Software and Related Publications

Anett Hoppe¹ , Jascha Hagen², Helge Holzmann³ , Günter Kniesel⁴,
and Ralph Ewerth^{1,3} 

¹ Leibniz Information Centre for Science and Technology (TIB), Hannover, Germany
{anett.hoppe,ralph.ewerth}@tib.eu

² Leibniz Universität Hannover, Hannover, Germany
jascha.hagen@yahoo.de

³ L3S Research Center, Leibniz Universität Hannover, Hannover, Germany
holzmann@l3s.de

⁴ University of Bonn, Bonn, Germany
gk@iai.uni-bonn.de

Abstract. Scientific software is one of the key elements for reproducible research. However, classic publications and related scientific software are typically not (sufficiently) linked, and tools are missing to jointly explore these artefacts. In this paper, we report on our work on developing the analytics tool SciSoftX (<https://labs.tib.eu/info/projekt/scisoftx/>) for jointly exploring software and publications. The presented prototype, a concept for automatic code discovery, and two use cases demonstrate the feasibility and usefulness of the proposal.

Keywords: Software reproducibility · Source code exploration
Cross-modal relations

1 Introduction

The open science movement works towards the general availability of scientific insight and is considered one answer to the so-called “reproducibility crisis” [2]. Science results are often generated by a combination of software, data, and parameters, all of which contribute to the final result (and its interpretation). The complexity of all these elements is hardly describable in a single article – and often the publication does not allow the full reproduction of the achieved results. In the line of work towards consequent reproducibility of scientific results, there are three main tasks to be tackled: (a) motivate researchers to reproduce past results; (b) develop novel ways for the integrated presentation of scientific results; (c) develop tools which allow for exploration of existing scientific works.

The work at hand focuses on the two latter objectives. It presents a tool which facilitates the examination of existing research involving software by joint exploration of a scientific article and the respective source code. The prototype allows the exploration of both in one interface, and the semi-automatic creation

of semantic relations between them. The software is extended by basic visualisations. This kind of work is related to research areas, which have been active for decades: (a) automatic code analysis, and (b) automatic analysis of scientific publications. Solutions for automatic code analysis aim at generating textual documentation [7], summarising code [8], or at generating visualisations [4]. Also common is the generation of formal code models using semantic technologies [1] or logical constructs as realised in tools such as JTransformer¹. While there is much work on linking code to other (textual) resources (e.g. traceability [3]), to documentation [4], on the automatic understanding of scientific publications [5], or on linking publications with software and archiving them [6], there has been little work on *joint* analytics of scientific software and publications, yet [9].

2 SciSoftX: Scientific Software Explorer

The Scientific Software Explorer provides researchers with functionalities for the exploration of external article-software ensembles and/or annotation of own works for better comprehensibility. Its final version will provide functionalities such as (a) manual annotation of article-software relations, (b) semi-automatic discovery of relations, and (c) visualisations for relation exploration.

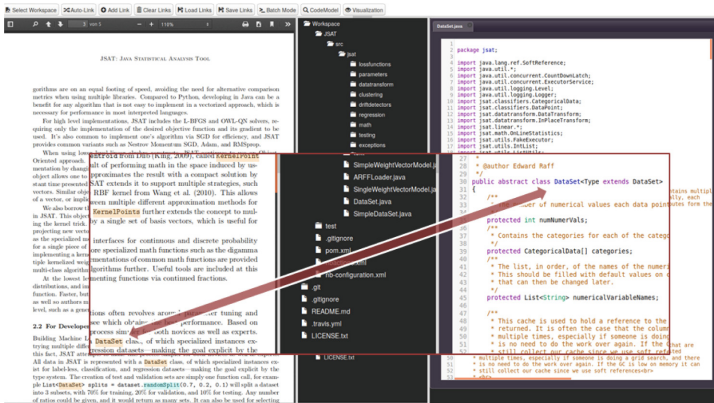


Fig. 1. Main window of the GUI: linked code references are highlighted in colour. (Colour figure online)

2.1 Functionality

SciSoftX allows the user to open and simultaneously view a software project and a publication (Fig. 1). Parsing and processing of source code is realised

¹ <http://sewiki.iai.uni-bonn.de/research/jtransformer/start>.

2.2 Use Cases

Use Case 1 – Reader-side: A researcher reads a publication that refers to a blob of software and then tries to understand the structure and rationale of the software. This time-consuming task can be supported by the automatic creation of links between textual description and actual source code, and the visualisations provided by SciSoftX. The user can click on nodes in the visualisation or on text elements that are highlighted in the publication and explore the implementation details, discover additional parameters, and understand the relevant code part step by step. Furthermore, it is possible to manually add and save useful information and metadata, which can help future users to explore the software.

Use Case 2 – Author-side: Paper authors can use SciSoftX to ensure their software is easily understood, e.g. in a reviewing process or for re-use. Therefore, they make use of the manual and automatic methods to annotate the semantic relations between their paper and the underlying software and publish the annotations. The visualisation of cross-modal relations can aid the authors (and the reviewers) to decide whether all relevant code parts and parameters are covered by the publication. In this way, the tool helps to evaluate the quality of the software description in a paper.

3 Conclusion

Reproducibility is one of the major issues of today’s scientific landscape. In this paper, we have reported on work in progress for an analytics tool that allows users to explore relations between scientific software and publications. To this date, the tool features simple mechanisms for detecting links between software and publications which serve as a proof of concept. Future work will explore (a) more powerful infrastructures for code analysis, (b) more sophisticated means for text/image analysis, e.g. mapping diagrams and formulas to source code.

References

1. Atzeni, M., Atzori, M.: Codeontology: RDF-ization of source code. In: d’Amato, C. (ed.) ISWC 2017. LNCS, vol. 10588, pp. 20–28. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68204-4_2
2. Baker, M.: 1,500 scientists lift the lid on reproducibility. *Nat. News* **533**(7604), 452 (2016)
3. Borg, M., Runeson, P., Ardö, A.: Recovering from a decade: a systematic mapping of information retrieval approaches to software traceability. *Empir. Softw. Eng.* **19**(6), 1565–1616 (2014). <https://doi.org/10.1007/s10664-013-9255-y>
4. Chen, X., Hosking, J.G., Grundy, J.: Visualizing traceability links between source code and documentation. In: IEEE Symposium on Visual Languages and Human-Centric Computing, Innsbruck, Austria, pp. 119–126 (2012). <https://doi.org/10.1109/VLHCC.2012.6344496>

5. Constantin, A.: Automatic structure and keyphrase analysis of scientific publications. Ph.D. thesis, University of Manchester, UK (2014). <http://www.manchester.ac.uk/escholar/uk-ac-man-scw:230124>
6. Holzmann, H., Sperber, W., Runnwerth, M.: Archiving software surrogates on the web for future reference. In: Fuhr, N., Kovács, L., Risse, T., Nejd, W. (eds.) TPD 2016. LNCS, vol. 9819, pp. 215–226. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-43997-6_17
7. Moser, M., Pichler, J.: Documentation generation from annotated source code of scientific software: position paper. In: Proceedings of the International Workshop on Software Engineering for Science, SE4Science@ICSE 2016, 14 May 2016–22 May 2016, Austin, Texas, USA, pp. 12–15. ACM (2016). <https://doi.org/10.1145/2897676.2897679>
8. Nazar, N., Hu, Y., Jiang, H.: Summarizing software artifacts: a literature review. *J. Comput. Sci. Technol.* **31**(5), 883–909 (2016). <https://doi.org/10.1007/s11390-016-1671-1>
9. Witte, R., Li, Q., Zhang, Y., Rilling, J.: Text mining and software engineering: an integrated source code and document analysis approach. *IET Softw.* **2**(1), 3–16 (2008). <https://doi.org/10.1049/iet-sen:20070110>