Provenance based software dashboards

Lynn von Kurnatowski
Institute for Software Technology
German Aerospace Center (DLR)
Weßling, Germany
https://orcid.org/0000-0001-5144-702X

Martin Stoffers
Institute for Software Technology
German Aerospace Center (DLR)
Cologne, Germany
https://orcid.org/0000-0003-2987-4345

Carina Haupt
Institute for Software Technology
German Aerospace Center (DLR)
Berlin, Germany
https://orcid.org/0000-0001-6447-1379

I. Challenge

Over the past years, software conquered a growing number of application areas. Particularly in research, where software is already an important innovation factor and integral part. This statement is substantiated by the results of a survey that we conducted DLR[1]-wide in 2018 [1]. It shows that many researchers develop software in their daily work. It even indicates that researchers, who are already involved in software development, are interested in investing even more time in software related activities.

The increasing role of software in research, in combination with the increase of complexity in scientific fields, has led to the necessity of complex software solutions. Thus intensifying the need to ensure the quality, reliability, and trustworthiness of software systems in research. Especially in safety-critical areas, where small software errors can lead to serious failures. In order to be able to make an assurance about quality, reliability, and trustworthiness of a software product, it is necessary to have a comprehensive and detailed understanding of a software project. The large amount of heterogeneous data, which is generated before, during, and after the development in so-called software repositories can serve as data sources [2]. These repositories already contain information about the source code and the software development process. One approach to explore this data is to analyse their *provenance*.

As stated by Moreau in 2010 in "The Foundations for Provenance on the Web":

> "Provenance [...] is becoming an important concern for several research communities in computer science, since it offers the means to verify data products, to infer their quality, to analyze the processes that led to them, and to decide whether they can be trusted." [3, Page 3]

Today, *provenance* [4] is used to verify data products and to analyze processes that led to them. Our work aims to standardize, generate, and use *provenance of software artifacts* and *provenance of software development processes* by defining a general provenance data model for software development processes. Our goal is to be able to trace and determine the origin of artifacts such as issues, source code files, build results, or documentation, and to understand and get insights into the process as a whole using analysis and dashboards.

[1]German Aerospace Center (DLR)

II. Opportunity

Software development is a creative and complex process based on human decisions. Consequently, developing software as error-free as possible is a challenging task. To assess the reliability and trustworthiness of software systems, the provenance of the development process and the produced software artifacts can be recorded and analyzed.

The aforementioned increase of complexity in software solutions results in the need of more extensive development processes. Thus, resulting in very large and complex provenance graphs, which are difficult to manually understand, interpret, and verify.

Visual analysis is an opportunity to easily and quickly verify a software project, since the human brain is an amazing pattern matching machine. Thus enabling it to process large amount of visual information at once. Our idea is to create an interactive dashboard solution based on a data set of provenance describing software development processes. This allows developers to easily and quickly get an overview about a software status regarding it's quality, reliability, and trustworthiness. For this, different metrics are visualised using appropriate visualisation methods, like graph visualizations, metrics visualizations (e.g., bar charts), time-oriented visualizations (e.g., Sankey charts), task-oriented and work process-oriented visualizations (e.g., Gantt charts), or hierarchy-oriented visualizations (e.g., treemap charts).

As first step, we developed a dashboard prototype that contains individual visualizations with interactive controls (Figure 1) [5]. For our prototype we used Python and the libraries Plotly, and Dash[2]. Before visualizing the data we query it with the libraryPy2neo from a Neo4j graph database instance[3] and use Pandas to prepare the data and temporary store query results.

Since not all popular software metrics and information are of equal importance to the user, we are conducting a focus group interview to identify what requirements users have for such a visualization. As a next step a first version of a dashboard will be developed which takes the user's requirements

[2]https://dash.plotly.com/

[3]https://neo4j.com/product/neo4j-graph-database/

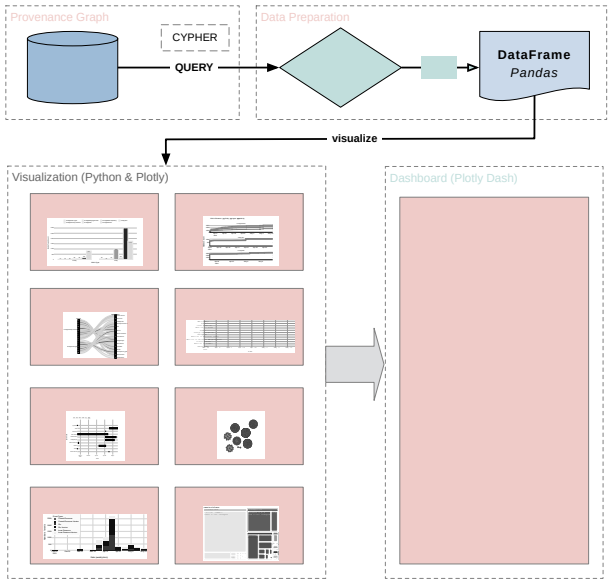

Fig. 1. Provenance visualization flow: querying the Neo4j database with CYPHER queries, storing query results and filtering and data cleansing with Pandas, visualizing with Python and PLOTLY, and displaying in the web-based dashboard with PLOTLY DASH [5].

into consideration. Finally, the resulting dashboard will be evaluated through real-world tests, to ensure that it provides an intuitive way to understand software projects.

III. TIMELINESS OR MATURITY

Due to the complexity and diversity of today's software systems, numerous development tools are used to support the the developer in his daily work. A typical tool suite consists of at least an integrated development environment (IDE), a version control system, an issue tracker, a continuous integration and continuous deployment infrastructure, and a documentation system. During the software development process, all these tools produce a large amount of different data. We use this heterogeneous data to extract the complete provenance of all process steps using W3C PROV Data Model.

The W3C specification PROV [6], which among other definitions defines the Provenance Data Model (PROV-DM) [7], allows to collect and record provenance in a well defined and machine readable formats.

Using the W3C PROV data model, we designed a general provenance model to extract *retrospective provenance* [8] about software development processes [9]. This includes provenance information of the underlying version control system as well as provenance of the respective issue trackers and release systems. The resulting provenance data then contains all activities (e.g., commits, issues changes, releases), the generated or changed entities (e.g., source code files or issues), and involved agents (e.g., developers, testers, or users) along with their relations.

Using our tool GitLab2Prov [10], [11] this information can already be collected from the web-based DevOps life-cycle tool GitLab[4]. The tool uses the GitLab API to generate the provenance information in the form of the text representation PROV-JSON, which is converted to other formats for further analysis and can be imported as a labeled property graph into the graph database Neo4j[5] using the tool prov2neo. These tools were already successfully applied to collect provenance information of inner-source and open source project [10], [12].

AVAILABILITY

GitLab2Prov is available as Open Source software under the MIT license: https://github.com/DLR-SC/gitlab2prov.

The tool prov2neo is available under the MIT license at https://github.com/DLR-SC/prov2neo.

REFERENCES

[1] L. von Kurnatowski, T. Schlauch, and C. Haupt, "Software development at the german aerospace center: Role and status in practice," in *Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops*, ser. ICSEW'20. New York, NY, USA: Association for Computing Machinery, 2020, p. 157–160. [Online]. Available: https://doi.org/10.1145/3387940.3392244

[2] L. von Kurnatowski, M. Stoffers, M. Weigel, M. Meinel, Y. Wasser, K. Rack, and H. Fiedler, "Scientific software engineering: Mining repositories to gain insights into bacardi," in *2020 IEEE Aerospace Conference*, 2020, pp. 1–10.

[3] L. Moreau, "The Foundations for Provenance on the Web," *Foundations and Trends® in Web Science*, vol. 2, no. 2-3, pp. 99–241, 2010. [Online]. Available: http://www.nowpublishers.com/article/Details/WEB-010

[4] L. Moreau, P. Groth, S. Miles, J. Vazquez-Salceda, J. Ibbotson, S. Jiang, S. Munroe, O. Rana, A. Schreiber, V. Tan, and L. Varga, "The provenance of electronic data," *Communications of the ACM*, vol. 51, no. 4, pp. 52–58, 2008.

[5] A. Schreiber, L. Kurnatowski, A. Meinecke, and C. Boer, "An interactive dashboard for visualizing the provenance of software development processes," in *2021 Working Conference on Software Visualization (VISSOFT)*. Los Alamitos, CA, USA: IEEE Computer Society, sep 2021, pp. 100–104. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/VISSOFT52517.2021.00019

[6] L. Moreau and P. T. Groth, *Provenance: An Introduction to PROV*, ser. Synthesis Lectures on the Semantic Web: Theory and Technology. Morgan & Claypool Publishers, 2013.

[7] L. Moreau and P. M. et al., "Prov-dm: The prov data model," Online, 2013, retrieved: 03.10.2021. [Online]. Available: https://www.w3.org/TR/prov-dm/

[8] T. McPhillips, S. Bowers, K. Belhajjame, and B. Ludäscher, "Retrospective provenance without a runtime provenance recorder," in *Proceedings of the 7th USENIX Conference on Theory and Practice of Provenance*, ser. TaPP'15. USA: USENIX Association, 2015.

[9] A. Schreiber and C. de Boer, "Modelling knowledge about software processes using provenance graphs and its application to git-based version control systems," in *Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops*, ser. ICSEW'20. New York, NY, USA: Association for Computing Machinery, 2020, p. 358–359. [Online]. Available: https://doi.org/10.1145/3387940.3392220

[10] A. Schreiber, C. de Boer, and L. von Kurnatowski, "Gitlab2prov—provenance of software projects hosted on gitlab," in *13th International Workshop on Theory and Practice of Provenance (TaPP 2021)*. USENIX Association, Jul. 2021. [Online]. Available: https://www.usenix.org/conference/tapp2021/presentation/schreiber

[11] C. de Boer and A. Schreiber, "Dlr-sc/gitlab2prov: Gitlab2prov 0.5," Jun. 2021. [Online]. Available: https://doi.org/10.5281/zenodo.5009043

[12] A. Schreiber, L. von Kurnatowski, and C. de Boer, "Analyzing software engineering processes with provenance-based knowledge graphs," in *2021 IEEE Aerospace Conference (50100)*, 2021, pp. 1–11.

[4]https://about.gitlab.com/features/

[5]https://neo4j.com/product/neo4j-graph-database/