The Ultimate Comparison Framework
(Tool Demonstration)

Oliver Kopp

IPVS, University of Stuttgart, Germany
kopp@informatik.uni-stuttgart.de

Abstract. Researches come up with scientific criteria to compare different tools and spend huge effort to run the comparison. The presentation of the comparison results is an open issue. The "Ultimate Comparison Framework" is one solution enabling a) collection of evaluation data using markdown and b) presentation of the data set in a web application.

1 Introduction

The accumulation of knowledge is an essential condition for a field to be scientific and to develop [3]. In the field of information systems research in general and business process management in particular, literature reviews are conducted to harvest the body of knowledge [7, 8]). There is still lack of sharing analysis results, especially in the field of business process management (cf. Recker and Mendling [8]). The Ultimate Comparison Framework is an approach to fill this gap: It supports publishing reviewing results in an open way. It further supports updating the research results by established collaborative software engineering techniques such as GitHub pull requests [2]. The current focus of the framework is on comparison of tools and technologies, but not limited to it.

The development of the Ultimate Comparison Framework was driven by a) sustaining search results in the context of finding the most fitting tool for a task and b) offering a framework for comparative studies crafted by students in the context of their software engineering trainings (cf. "Fachstudie" [6]).

The framework is called "Ultimate Comparison Framework", because it offers creation of multiple "Ultimate Comparisons" of different tools in different settings. The name "Ultimate Comparison" stems from the claim that the framework is easy to use (covering creation and maintenance of data as well as presenting data) and that it will be used by many researchers to present their research results. This paper is a first step into this direction by making it known to a broader community.

Users of the framework are a) researchers wanting to sustain their survey results, b) researches investigating other surveys, and c) industry users interested in introducing a new tool or framework and aiming for a scientifically-grounded comparison of existing work.

Already published "ultimate comparisons" include: Comparison of Cloud Deployment and Management Tools[1], Comparison of IoT Platforms[2], Comparison of Time-

[1] https://ultimate-comparisons.github.io/ultimate-deployment-tool-comparison/
[2] https://ultimate-comparisons.github.io/ultimate-IoT-platform-comparison/

J. Manner, S. Haarmann, S. Kolb, O. Kopp (Eds.): 12th ZEUS Workshop, ZEUS 2020, Potsdam, Germany, 20-21 February 2020, published at http://ceur-ws.org/Vol-2575

Series Databases[3], Comparison of Message Brokers[4], Comparison of Graph Libraries for JavaScript[5], Comparison of Web-based IDEs[6], Comparison of Literature Management Software[7], and Comparison of LaTeX Building Helpers[8]. The list of available comparisons is constantly updated at `https://ultimate-comparisons.github.io/`. In other words: the Ultimate Comparison Framework is not limited to a fixed set of tools, but can be used to compare tools, where comparison critera can be defined and the result can be presented as table.

The idea stems from the PaaSfinder web application [4, 5] (available at `https://paasfinder.org/`), which offers similar functionality. In contrast to PaaSfinder, the Ultimate Comparison Framework is hosted as a website and stores its data in Markdown. It is the first tool basing on plain-text (Markdown) for data storage and on GitHub for data collection and data presentation.

This paper demos the framework by showing the Ultimate Comparison of Open Source Time-Series-Databases [1]. We explain the reader-facing interface (Sect. 2) and the contributor-facing interface (Sect. 3). After sketching the implementation (Sect. 4), we provide a short summary and outlook (Sect. 5).

2 Reader-facing Interface

A person interested in comparison results, opens the web page presenting the Ultimate Comparison. He sees a table showing the compared tools as row headings and criteria as column headings. In each cell, the matched criteria are listed. For instance, if a tool is offered both under MIT and a proprietary license, the cell for "License" shows "MIT" and "proprietary". The person can sort the table by clicking on a criteria name. It is also possible to show details on each tool. An example table is presented in Fig. 1.

3 Contributor-facing Interface

Data in a comparison may get outdated or data is missing. Then, a user can update the data using the typical GitHub flow[9]. The user searches for the Markdown file containing the data of the tool (for instance the time-series database Timely), modifies it accordingly, and sends a pull request.

In case a new comparison should be set up, its name, description and comparison criteria need configured. This is enabled by a YAML configuration file. Each criterion bundles a set of possible values: Possible data types for a criterion are text, enums, and numbers. For instance, the code license can be enumerated and results of performance

[3] `https://tsdbbench.github.io/Ultimate-TSDB-Comparison/`

[4] `https://ultimate-comparisons.github.io/ultimate-message-broker-comparison/`

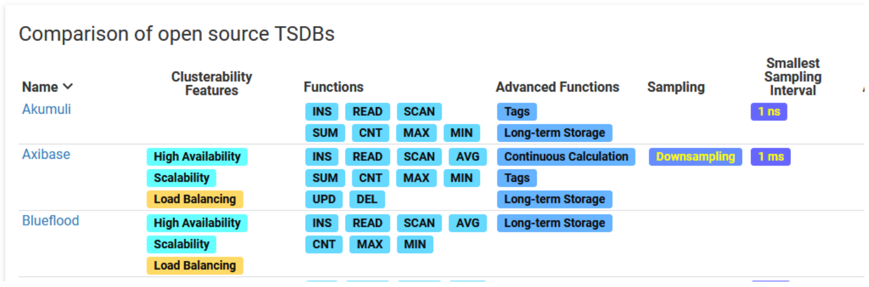
[5] `https://ultimate-comparisons.github.io/ultimate-graphframework-comparison/`

[6] `https://ultimate-comparisons.github.io/ultimate-webIDE-comparison/`

[7] `https://ultimate-comparisons.github.io/ultimate-reference-management-software-comparison/`

[8] `https://ultimate-comparisons.github.io/ultimate-latex-makers-comparison/`

[9] `https://guides.github.com/introduction/flow/`



Comparison of open source TSDBs

Name	Clusterability Features	Functions	Advanced Functions	Sampling	Smallest Sampling Interval
Akumuli		INS READ SCAN SUM CNT MAX MIN	Tags Long-term Storage		1 ns
Axibase	High Availability Scalability Load Balancing	INS READ SCAN AVG SUM CNT MAX MIN UPD DEL	Continuous Calculation Tags Long-term Storage	Downsampling	1 ms
Blueflood	High Availability Scalability Load Balancing	INS READ SCAN AVG CNT MAX MIN	Long-term Storage		

Fig. 1. Table of tools (bottom omitted)

measurements can be provided. Each criterion is input as Markdown heading. In case of text, the text has to be put below the heading. In case of an enum, all matching enums are listed as markdown list (e.g., "`- Apache-2.0`" below "`## License`" in the case of a single license). In case of numbers, the number is given below the heading.

When designing a criterion, it is important to ensure that it can be used for comparing different tools. For instance, if performance of tools depend on the environment, the chosen environment should be described. Alternatively, an Ultimate Comparison for each different environment could be setup.

To set up an ultimate comparison, basic knowledge of GitHub pages and the usage of CI/CD tools is necessary. Furthermore, writing YAML is an essential software engineering skill. To input data into an ultimate comparison, the knowledge of using GitHub as well as entering Markdown are necessary.

4 Implementation

The web interface is implemented using Angular. The data is converted from Markdown to JSON using Java on a CI server (currently TravisCI). The JSON is read by the web interface running on the client side in the browser. The web interface is hosted in the branch "gh-pages" of the respective comparison and offered to users by the GitHub Pages offering[10]. The source code of the implementation is open source and available at `https://github.com/ultimate-comparisons/ultimate-comparison-framework`.

5 Conclusion and Outlook

This paper presented the Ultimate Comparison Framework offering a collaborative comparison framework. It stores the data in Markdown files enabling the usage of arbitrary text editors to add or modify data. The data is rendered as static HTML page showing all criteria, the evaluated tools, and the fulfillment of each criterion in a table. It is an open discussion point whether tables are the best option to present comparison results. Future work has to evaluate other rendering possibilities such as charts.

[10] `https://pages.github.com/`

Facts such as write performance can also be captured in an Ultimate Comparision. The framework currently offers to fetch the last commit and decide on that fact if development is stalled. There is currently no way to trigger complete setup of each system, measure the performance, and update the result in an Ultimate Comparison. This is left as future work, for instance for TSDBBench[11].

Currently, there is no advanced analysis functionality offered. One idea is to offer clustering of the results. Currently, this can only be done by defining an enum criterion, where each enum value presents one cluster. Then, the user can sort by that criterion.

The next development ideas are a) offering a browser-based user interface to input data (instead of relying on GitHub) and b) using Wikipedia tables as data input; either as data source for displaying or as data source for synchronizing the local Markdown files. Future work includes measurement of the time required to publish and maintain results in comparison to other approaches such as scientific publications or tables in WikiPedia articles.

Acknowledgments We want to thank Stefan Kolb for the idea and inspiration of this tool Further, we want to thank Andreas Bader, Armin Hüneburg, and Christoph Kleine for discussions on the UI and driving the implementation. This work is partially funded by the Federal Ministry for Economic Affairs and Energy (BMWi) projects *Industrial Communication for Factories* (01MA17008G), *NEMAR* (03ET4018), and *SmartOrchestra* (01MD16001F).

References

1. Bader, A., et al.: Survey and Comparison of Open Source Time Series Databases. In: BTW2017 Workshops. GI e.V. (2017)
2. Gousios, G., Pinzger, M., van Deursen, A.: An exploratory study of the pull-based software development model. In: ICSE. ACM Press (2014)
3. Hunter, J.E., Schmidt, F.L., Jackson, G.B.: Meta-Analysis: Cumulating Research Findings across Studies. Educational Researcher 15(8), 20 (Oct 1986)
4. Kolb, S.: On the Portability of Applications in Platform as a Service. Ph.D. thesis, University of Bamberg, Germany (2019)
5. Kolb, S., Wirtz, G.: Towards Application Portability in Platform as a Service. In: SOSE. IEEE (2014)
6. Ludewig, J.: Erfahrungen bei der Lehre des Software Engineering. In: Software Engineering im Unterricht der Hochschulen. dpunkt.verlag (2009)
7. Paré, G., Trudel, M.C., Jaana, M., Kitsiou, S.: Synthesizing information systems knowledge: A typology of literature reviews. Information & Management 52(2), 183–199 (Mar 2015)
8. Recker, J., Mendling, J.: The State of the Art of Business Process Management Research as Published in the BPM Conference. Business & Information Systems Engineering 58(1), 55–72 (Nov 2015)

All links were last followed on February 19, 2020.

[11] `https://tsdbbench.github.io/`