EuroVis Workshop on Visual Analytics (2021)
J. Bernard and K. Vrotsou (Editors)

Customizable Coordination of Independent Visual Analytics Tools

Lars Nonnemann[†1], Marius Hogräfer[†2], Heidrun Schumann[1], Bodo Urban[3], Hans-Jörg Schulz[2]

[1] University of Rostock, Institute for Visual and Analytic Computing, Germany
[2] Aarhus University, Department of Computer Science, Denmark
[3] Competence Center for Visual Assistance Technologies, Fraunhofer Institute for Computer Graphics Research Rostock, Germany

Abstract

While it is common to use multiple independent analysis tools in combination, it is still cumbersome to carry out a cross-tool visual analysis. Some dedicated frameworks addressing this issue exist, yet in order to use them, a Visual Analytics tool must support their API or architecture. In this paper, we do not rely on a single predetermined exchange mechanism for the whole ensemble of VA tools. Instead, we propose using any available channel for exchanging data between two subsequently used VA tools. This effectively allows to mix and match different data exchange strategies within one cross-tool analysis, which considerably reduces the overhead of adding a new VA tool to a given tool ensemble. We demonstrate our approach with a first implementation called AnyProc and its application to a use case of three VA tools in a Health IT data analysis scenario.

CCS Concepts

• ***Human-centered computing*** → *Visualization systems and tools;* • ***Information systems*** → *Data exchange;*

1. Introduction

Software integration is a longstanding challenge in computer science [GTT03, LC04]. For Visual Analytics (VA), this challenge has been addressed with approaches ranging from analysis libraries [VR17] to extensible frameworks [BSS*19]. Yet all these approaches have not eliminated the need to switch between different, highly specialized VA tools in an interactive visual data analysis. The reason is that no framework can be top of the class at all possible VA tasks. Hence even in the age of powerful VA software like KNIME [BCD*09] and Tableau [STH02], we still rely on separately running OpenRefine [Ham13] for any serious data cleaning task or Gephi [BHJ09] for advanced network visualization.

Switching between VA tools breaks the analytic flow by having to worry about technical constraints of data export/import between tools. Something simple like going back to a previous tool to readjust a parameter and observe its effects in the subsequent tool becomes a rather painful experience – in particular if one needs to do this back-and-forth between tools multiple times to get the parameter right. In some cases, VA tools that do not support the export/import of data or visualizations at all, let alone in a standardized format. So, while switching between independent VA tools is necessary for complex application scenarios, it comes at a cost.

Our approach for VA tool coordination acknowledges the need for multiple specialized VA tools to perform VA at its best. Yet we aim to reduce the cost of using these independent VA tools in conjunction by facilitating and automating the switches between them. To that end, we build on our recent conceptual models for VA tool coordination [SRN*20] and data exchange among VA tools [NSS*20] to realize such a coordination and to demonstrate that it is feasible and useful. Our approach borrows from the paradigm of data-flow oriented visualization software. Just that instead of composing individual modules of the same system into one data flow, it does so for individual VA tools. After a brief summary of related work in Sec. 2, we outline our approach in Sec. 3 and demonstrate it for a use case with three independent tools in Sec. 4.

2. Related Work

Various approaches exist to support working with multiple individual tools. These approaches can be broken down along the three concerns they mainly address: data exchange between tools, UI integration between tools, and analytic process support across tools.

Getting data from one tool to another in a seamless way can be done either via a central hub such as a relational database [NS00, PP01], or via decentralized web services [RM11, LWH14]. Dedicated software design pattern like the *proxy tuple* can aid different tools to access a unified data model [FHBW11]. Yet for integrating a tool in a software ecosystem based on any of these approaches, it is commonly required to use the same approach.

UI integration between different tools plays a major role when using multiple individual tools concurrently. Such integration can be rather lightweight by adding visual links or information scents to highlight the same data across tools [WPL*10, FLCT14], or it

[†] The first two authors contributed equally to this work.

can be more involved and actually blend different tools into a combined UI [SCPR06,TMC04]. Though in particular the blending approaches have the downside that users can no longer rely on their established mental map of the individual tool UIs.

Support for the analytic process across multiple tools is given in two possible ways: by guiding the user along a given analytic workflow from tool to tool [SSL*12], or by establishing analytic provenance across tools [WBV14,AFH*19]. The challenge shared among these mechanisms is to capture internal states and transitions from within the individual tools.

The approach presented in this paper addresses mainly the data exchange as a foundation for the two other concerns. Yet, it differs from the existing approaches in three regards:

(1) It is *opportunistic* by using any available data channel between two VA tools. This allows us to mix tools from different software ecosystems – e.g., native applications using the file system for data exchange and web apps using a server-based data exchange.

(2) It is *minimalistic* by exchanging data only between VA tools used subsequently or concurrently during the analysis. So instead of broadcasting data updates to all tools, our approach simply propagates data changes along the analytic workflow as it is carried out.

(3) It sees VA tools as *atomic* and focuses on the switch points when the user changes from one VA tool to another. We do not aim to open the black boxes of the VA tools [MPG*14] by observing internal states, but merely to capture their output and feed it into the next black box, as the user switches between them. Nevertheless, that may still require minimal code changes – for example in tools that lack any data input/output functionality to be otherwise used.

3. A Framework for Coordinating Analytical Tool Chains

VA tools are often used in a chained manner using specialized tools for each analysis step from cleaning, preprocessing, and visual-interactive analysis to fine-tuning the visual results. Our previously established approach for VA tool coordination [SRN*20] breaks tool chains down in three layers:

- The *Usage Flow* capturing which tools are intended to be used in which order, so as to know between which tools coordination is necessary and only to realize it between them.
- The *Data Flow* capturing the exchange of data between VA tools, so as to specify individually for each pair of VA tools to be coordinated how to exchange data between them.
- The *Control Flow* capturing when the actual transition between tools happens, so as to have a clear indicator when to switch and thus when to exchange data between VA tools.

To provide automated support for some of the tedious tasks involved in such a switch – first and foremost for exchanging data and parameters across tools – all three layers must be specified. In the following, we explain how our framework *AnyProc* (short for Analytical Process Constructor) realizes this specification. AnyProc is made available under a permissive open source license at `https://github.com/nonnemann/anyproc_public` and more details about its implementation and use are described in its accompanying wiki at `https://github.com/nonnemann/anyproc_public/wiki/Home`.

The Usage Flow is highly domain-dependent and influenced by user preferences and the analysis task at hand. This makes it difficult to pre-define, except in instances where an agreed upon best practice is to be carried out [SSL*12]. Hence, AnyProc exhibits the usage flow directly to the user through a graphical editor in which to assemble the tool chain. This editor uses the metaphor of data-flow oriented systems by allowing users to include VA tools from a repository of available tools and to link them in the order of their intended use. It explicitly provides the possibility to include one VA tool multiple times in a tool chain for its repeated use, as well as to place multiple tools at the same time for their concurrent use.

The Data Flow determines the technical realization of which data is passed in which way between subsequently and concurrently used VA tools. To that end, AnyProc allows further configuration of any pairwise data exchange in terms of which *channel* to use (e.g., a server, the file system, the system-wide clipboard) and in how this data exchange is to be performed (e.g., are changes only propagated forward from one tool to the next, or are they synced backward along the tool chain to show up in previously used tools). This pairwise configuration follows our recently proposed characterization of data exchange [NSS*20].

The Control Flow is the concrete instantiation of the usage and data flows when performing a concrete analysis. Through the invocation of VA tools, it describes how often and how long they are used, and it thus initiates the underlying syncing of data between VA tools. To this end, AnyProc provides a small persistent screen widget, called *executor*. It gives ready access to the tool chain by featuring buttons to start the next VA tool as well as to re-start the previous VA tool. When using these buttons, the defined data flow will be carried out – whether this means to simply send a few keystrokes to both VA tools for copying and pasting the data across, or to contact a server for a more elaborate data exchange.

4. Applying our Framework to a Cross-Tool VA Scenario from Health IT

This scenario uses AnyProc for the visual analysis of several tables from the critical care dataset MIMIC-III [JPS*16] using three independent VA tools. From this dataset, we use the ca. 46,000 patient records, their ca. 58,000 hospital admissions, as well as their registered fluid intake events (ca. 17.5 million entries) and fluid output events (ca. 3.6 million entries). The scenario itself is not particularly complex, but requires already quite a bit of back-and-forth between different tools. In that sense, it is nevertheless able to show the fundamental challenges of a cross-tool analysis and how AnyProc can support the user in dealing with them.

4.1. Specifying the Usage Flow

We plan to use the following three independent VA tools – in that order, but allowing for a back and forth between them if needed:

1. KNIME [BCD*09] for preprocessing the data using unsupervised machine learning methods like clustering and PCA;
2. VisFlow [YS17] for the exploratory, interactive visual analysis of the results obtained using KNIME;
3. ColorBrewer [HB03] for fine-tuning the color scale in the plots generated by VisFlow.

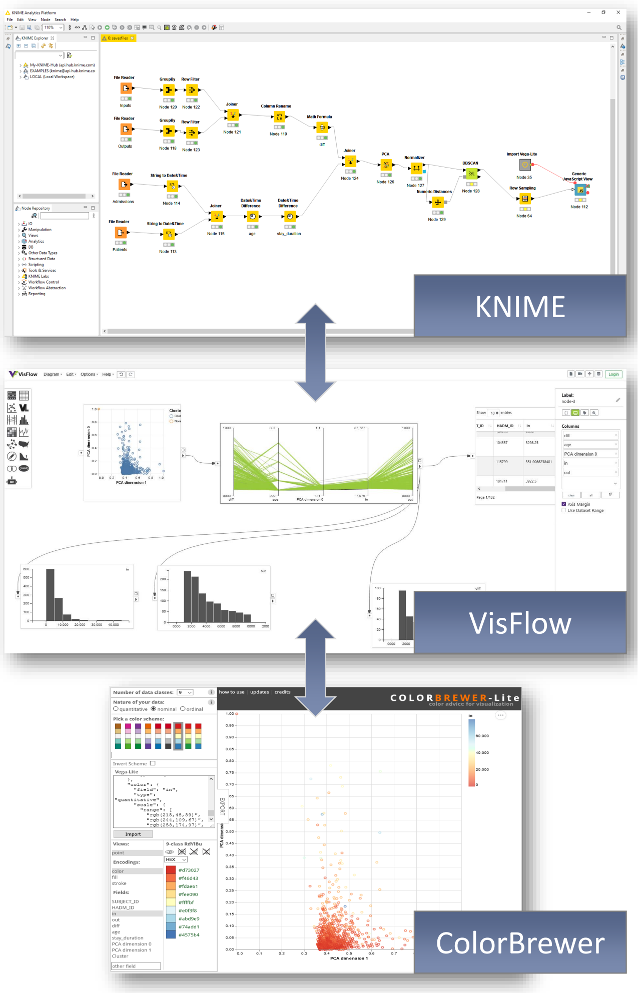

Figure 1: *Initial tool chain with KNIME, Visflow, and ColorBrewer.*

This setup is rather straightforward, as each of these three VA tools excels at a different stage of the analysis process: KNIME is powerful in its computational analysis capabilities, but it does not lend itself to the visual-interactive exploration of the computed results. Whereas, VisFlow has its strength in the flexible configuration and exploration of data visualizations, but lacks the advanced computational capabilities of KNIME. Finally, ColorBrewer is used to remedy VisFlow's missing support for selecting color scales. Figure 1 shows this usage flow among the three VA tools.

4.2. Specifying the Data Flow

In this second step, we now define how these three tools pass data among each other. At least in part, this requires some code changes as, for example, ColorBrewer in its original form does not allow any data to be passed in or out at all. Hence, we need a solution that allows us to add this functionality to ColorBrewer, while at the same time demanding less coding effort than integrating ColorBrewer's functionality into VisFlow. To this end, we rely on the open source library ReVize [HS19] and its accompanying ReVize server. The ReVize library establishes a standardized I/O interface for web-based applications through the use of Vega-Lite [SMWH17] as an all-encompassing exchange format that can be used to transfer data, its principal visual encoding as a chart, that chart's parameters (axis labels, color scales, etc.), and even some of the chart's interaction logic. For our analysis process, we use variants of KNIME, VisFlow, and ColorBrewer that are already ReVize-enabled that can be found at `https://vis-au.github.io/toolchaining/`. In that sense, ReVize provides us with a well-defined means to funnel data in and out of VA tools.

On top of ReVize comes the ReVize server, which is a hybrid HTTP/Websocket server that provides a lightweight channel through which changes of a Vega-Lite data/view specification can be sent. This channel exists in addition to other available channels, like the aforementioned data exchange via the clipboard. Which channel to use to connect two subsequent tools can be specified independently for each connection between tools in the data-flow graph in AnyProc – i.e., between KNIME and VisFlow, as well as between VisFlow and ColorBrewer.

We thus configure the connections accordingly, which results in the following data exchange characteristics for both connections [NSS*20]: Each data transfer includes the *full dataset* together with additional *metadata about its visual representation* using the *structured data format* Vega-Lite. Using the ReVize server as a *central infrastructure*, the data exchange is used for *inputs and outputs* from tools, as well as to broadcast interactive *modifications* from *one to many* other VA tools connected to that server. This is done *bidirectionally* (backward and forward along the tool chain) in a *synchronous* fashion where changes are *pushed* to the server when switching from one VA tool to another. At all times, the connected VA tools have *full access* to the most recent Vega-Lite specification, which is kept *persistent* on the server.

4.3. Specifying the Control Flow

Switching to the AnyProc executor, KNIME as the first VA tool in the tool chain is started. Since we configured it to connect via ReVize to the next tool in the tool chain, KNIME is opened already with a minimal analytic pipeline in place that shows the four data sources, as well as with a final Vega-Lite node that will output the necessary specification in the end. We first add data transformations (e.g., date conversions), computations (e.g., determining the patients' ages), cleaning (e.g., removing missing entries), and aggregations (e.g., summing up fluid intakes and outputs per admission), before joining all data of interest into a single table. We then perform a PCA on this multi-dimensional data to map them into two dimensions. On top of this mapping, we perform a DBSCAN clustering to find groups of patients and then we finally plot the resulting data. This is as far as KNIME takes us and we switch to VisFlow for interactively exploring the plot.

To do so, we click the [Next] button in the AnyProc executor, which opens up VisFlow with the plot generated in KNIME. In the background, hidden from the user, KNIME has continuously pushed any updates of the plot to the ReVize server – but only now with the click of the button, the ReVize server is instructed to pass these updates on to other tools. We now use VisFlow to add Parallel

Coordinates that provide more detail on the cluster characteristics. We then use brushing& linking and interactive tooltips to explore the clustered data. We soon realize that one of the clusters contains patients of over 300 years. This observation leads us to go back to KNIME to filter out these erroneous data.

A click on the [Previous] button in the AnyProc executor brings up KNIME again. We revise our analysis pipeline to filter out patients older than 130 years. After re-running the pipeline, the cluster of the 300 year old patients is gone. Via the [Next] button, we switch again to VisFlow, relying on AnyProc to update the contents in VisFlow according to the changes we made in KNIME.

In VisFlow, we explore the data further. We find some curious cases of patients who were given medication, but whose fluid intake is nevertheless registered as 0 ml or even -1 ml. On top of that, we notice that the absolute fluid levels are not very useful for our analysis as they vary drastically with the duration of the hospital stay. Instead, we would like to see, for example, a patient's average fluid levels per day or the difference between a patient's overall fluid intake and output. So, we switch back to KNIME using the [Previous] button, add the necessary filters and computations to the pipeline, and adjust the PCA to use the newly computed values.

Switching to VisFlow via the [Next] button updates the charts in VisFlow and we can proceed with our analysis. When investigating the newly computed difference between fluid intake and output, we would like to use a diverging color scale to better discern net loss from net gain. Unfortunately, VisFlow's capabilities for customizing color scales are very limited. To nevertheless be able to make the desired changes to the charts' colors, we switch to ColorBrewer using the [Next] button.

This opens-up ColorBrewer showing the visualization from VisFlow. As it was the case for KNIME, any update in VisFlow was already constantly pushed to the ReVize server in the background – yet the button click triggers the server to distribute the latest update to the connected tools. Now we can select an appropriate color scale while previewing its effects on the visualization. When done, we go back to VisFlow using the [Previous] button and there the changes made in ColorBrewer show up. Using VisFlow, we find more implausible data, such as patients who output an average of 17 liters of fluid per day or patients who output over 160 liters of fluid more than they take in. We could remove them by adding more filters to the preprocessing in additional roundtrips between KNIME and VisFlow. Yet, this no longer seems like a viable solution and we instead opt for adding the data cleaning tool OpenRefine to the beginning of our tool chain, which is sketched in Sec. 4.4.

Up to this point, we have made seven switches between tools. Without AnyProc, most of these switches would have involved, for example, navigating the file system to export and import the data, as well as navigating the start menu to find the next VA tool to be used. This is of course assuming that the tools rely on the same file format. With AnyProc, these costs of switching between tools are not gone, but they are paid once, up-front when setting up the usage flow and data flow, so that the technicalities involved no longer interrupt the analysis. The more tool switches are necessary in an analysis, the larger the benefit of using AnyProc. And as illustrated by this scenario, already a rather simple setup of two or three tools can make a considerable number of tool switches necessary.

4.4. Customizing the Usage Flow and the Data Flow

Bringing a new tool like OpenRefine into the analysis, is a crucial feature to be able to accommodate the exploratory nature of many analyses and to dive into the unexpected findings they yield. To that end, we first extend the usage flow by opening the AnyProc editor again and dragging a new VA tool – in our case OpenRefine – into the tool chain. Once placed at the beginning of the tool chain, we also need to move the data sources from KNIME to OpenRefine. Finally, we need to specify how it connects to KNIME, i.e., the data flow of which data channel to use to pass on its output. The connection between the two tools can then be edited to use, for example, the file system to pass data, as it is shown in Figure 2.

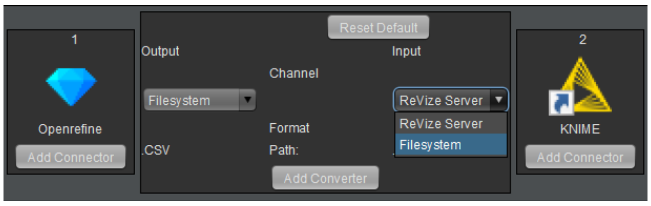

Figure 2: *Specifying the connection between OpenRefine and KNIME to use the file system for the data exchange. Data channels can be adjusted through drop-down menus and data converters can be added using the [Add Converter] button to solve any mismatches.*

5. Conclusions

With AnyProc, we provide a framework for the configuration and execution of VA tool chains that supports the automatic data exchange in the background. At this point, our framework can be considered a first step in the direction of full-fledged VA tool coordination. Many issues are still open for future research that such a framework needs to address. From the perspective of data exchange, this includes the handling of different data formats capturing different aspects of the data, its analysis and visualization, potentially even in ambiguous ways. In particular the latter is a problem for data exchange between VA tools, as we have shown previously that even standardized data formats like NetCDF do not prevent different tools from interpreting the same data very differently [SNHS17]. From the user perspective, it is imperative to provide visual feedback on how far along the tool chain the analysis is and what exactly the previous and the next VA tools are.

All that being said, the work presented here provides already a good first impression of what VA tool coordination can do for us: we neither have to make do with the limitations of individual tools, nor do we have to carry all of the burden of integrating them with each other. A pragmatic middle ground is possible that eases the combined use of individual tools without fully integrating them.

Acknowledgments

We gratefully acknowledge funding of this research by the German Research Foundation (DFG) through project *UnIVA*, as well as by the Innovation Fund Denmark (IFD) through the Grand Solution project *Hospital@Night*. We also thank Mario Aehnelt for his input and suggestions on this work, as well as Michael Zielinski for contributing to the implementation of the Data Flow specification.

References

[AFH*19] Archambault D., Fekete J.-D., Herschel M., Jankun-Kelly T. J., Kerren A., Laramee R. S., Lunzer A., Stitz H., Tory M.: VAPS: Visual analytics provenance standard for cross-tool integration of provenance handling. In *Dagstuhl Report on Provenance and Logging for Sense Making* (2019), pp. 42–46. doi:10.4230/DagRep.8.11.35. 2

[BCD*09] Berthold M. R., Cebron N., Dill F., Gabriel T. R., Kötter T., Meinl T., Ohl P., Thiel K., Wiswedel B.: KNIME – the Konstanz information miner: Version 2.0 and beyond. *SIGKDD Explorations Newsletter 11*, 1 (2009), 26–31. doi:10.1145/1656274.1656280. 1, 2

[BHJ09] Bastian M., Heymann S., Jacomy M.: Gephi: An open source software for exploring and manipulating networks. In *Proceedings of the International AAAI Conference on Web and Social Media (ICWSM)* (2009), AAAI, pp. 361–362. 1

[BSS*19] Behrisch M., Streeb D., Stoffel F., Seebacher D., Matejek B., Weber S. H., Mittelstädt S., Pfister H., Keim D.: Commercial Visual Analytics Systems–Advances in the Big Data Analytics Field. *IEEE Transactions on Visualization and Computer Graphics 25*, 10 (2019), 3011–3031. doi:10.1109/TVCG.2018.2859973. 1

[FHBW11] Fekete J.-D., Hémery P.-L., Baudel T., Wood J.: Obvious: A meta-toolkit to encapsulate information visualization toolkits – one toolkit to bind them all. In *Proceedings of the IEEE Conference on Visual Analytics Science and Technology (VAST)* (2011), IEEE, pp. 91–100. doi:10.1109/VAST.2011.6102446. 1

[FLCT14] Fourney A., Lafreniere B., Chilana P., Terry M.: InterTwine: Creating interapplication information scent to support coordinated use of software. In *Proceedings of the 27th annual ACM symposium on User interface software and technology (UIST)* (2014), ACM, pp. 429–438. doi:10.1145/2642918.2647420. 1

[GTT03] Gorton I., Thurman D., Thomson J.: Next generation application integration: challenges and new approaches. In *Proceedings of the 27th Annual International Computer Software and Applications Conference (COMPAC)* (2003), pp. 576–581. doi:10.1109/CMPSAC.2003.1245398. 1

[Ham13] Ham K.: OpenRefine (version 2.5). http://openrefine.org. free, open-source tool for cleaning and transforming data. *Journal of the Medical Library Association 101*, 3 (2013), 233–234. doi:10.3163/1536-5050.101.3.020. 1

[HB03] Harrower M., Brewer C. A.: ColorBrewer.org: An online tool for selecting colour schemes for maps. *Cartographic Journal 40*, 1 (2003), 27–37. doi:10.1179/000870403235002042. 2

[HS19] Hográfer M., Schulz H.-J.: ReVize: A Library for Visualization Toolchaining with Vega-Lite. In *Proceedings of Smart Tools and Apps for Graphics (STAG)* (2019), Eurographics Association, pp. 129–139. doi:10.2312/stag.20191375. 3

[JPS*16] Johnson A. E., Pollard T. J., Shen L., wei H. Lehman L., Feng M., Ghassemi M., Moody B., Szolovits P., Celi L. A., Mark R. G.: MIMIC-III, a freely accessible critical care database. *Scientific Data 3*, 1 (2016), 1–9. doi:10.1038/sdata.2016.35. 2

[LC04] Land R., Crnković I.: Existing approaches to software integration – and a challenge for the future. In *Proceedings of the 4th Conference on Software Engineering Research and Practice in Sweden* (2004). 1

[LWH14] Liu C., Wang J., Han Y.: Mashroom+: An interactive data mashup approach with uncertainty handling. *Journal of Grid Computing 12*, 2 (2014), 221–244. doi:10.1007/s10723-013-9280-5. 1

[MPG*14] Mühlbacher T., Piringer H., Gratzl S., Sedlmair M., Streit M.: Opening the black box: Strategies for increased user involvement in existing algorithm implementations. *IEEE Transactions on Visualization and Computer Graphics 20*, 12 (2014), 1643–1652. doi:10.1109/TVCG.2014.2346578. 2

[NS00] North C., Shneiderman B.: Snap-together Visualization: A User Interface for Coordinating Visualizations via Relational Schemata. In *Proceedings of the working conference on Advanced Visual Interfaces (AVI)* (2000), ACM, pp. 128–135. doi:10.1145/345513.345282. 1

[NSS*20] Nonnemann L., Schulz H.-J., Schumann H., Urban B., Aehnelt M.: A characterization of data exchange between visual analytics tools. In *Proceedings of the International Conference on Information Visualisation (IV)* (2020), IEEE, pp. 340–349. doi:10.1109/IV51561.2020.00066. 1, 2, 3

[PP01] Pattison T., Phillips M.: View coordination architecture for information visualisation. In *Proceedings of the Asia-Pacific Symposium on Information Visualisation (APVIS)* (2001), pp. 165–169. 1

[RM11] Rogowitz B. E., Matasci N.: Metadata Mapper: a web service for mapping data between independent visual analysis components, guided by perceptual rules. In *Human Vision and Electronic Imaging XVI*, Rogowitz B. E., Pappas T. N., (Eds.), vol. 7865. SPIE, 2011, pp. 165–177. doi:10.1117/12.881734. 1

[SCPR06] Stuerzlinger W., Chapuis O., Phillips D., Roussel N.: User interface façades: Towards fully adaptable user interfaces. In *Proceedings of the 19th Annual ACM Symposium on User Interface Software and Technology (UIST)* (2006), ACM, pp. 309–318. doi:10.1145/1166253.1166301. 2

[SMWH17] Satyanarayan A., Moritz D., Wongsuphasawat K., Heer J.: Vega-Lite: A grammar of interactive graphics. *IEEE Transactions on Visualization and Computer Graphics 23*, 1 (2017), 341–350. doi:10.1109/TVCG.2016.2599030. 3

[SNHS17] Schulz H.-J., Nocke T., Heitzler M., Schumann H.: A systematic view on data descriptors for the visual analysis of tabular data. *Information Visualization 16*, 3 (2017), 232–256. doi:10.1177/1473871616667767. 4

[SRN*20] Schulz H.-J., Röhlig M., Nonnemann L., Hográfer M., Aehnelt M., Urban B., Schumann H.: A layered approach to lightweight toolchaining in visual analytics. In *Computer Vision, Imaging and Computer Graphics Theory and Applications* (2020), Cláudio A. P., Bouatouch K., Chessa M., (Eds.), Communications in Computer and Information Science, Springer, pp. 313–337. 1, 2

[SSL*12] Streit M., Schulz H.-J., Lex A., Schmalstieg D., Schumann H.: Model-driven design for the visual analysis of heterogeneous data. *IEEE Transactions on Visualization and Computer Graphics 18*, 6 (2012), 998–1010. doi:10.1109/TVCG.2011.108. 2

[STH02] Stolte C., Tang D., Hanrahan P.: Polaris: A system for query, analysis, and visualization of multidimensional relational databases. *IEEE Transactions on Visualization and Computer Graphics 8*, 1 (2002), 52–65. doi:10.1109/2945.981851. 1

[TMC04] Tan D. S., Meyers B., Czerwinski M.: WinCuts: Manipulating arbitrary window regions for more effective use of screen space. In *CHI'04 Extended Abstracts on Human Factors in Computing Systems* (2004), ACM, pp. 1525–1528. doi:10.1145/985921.986106. 2

[VR17] Ventocilla E., Riveiro M.: Visual analytics solutions as 'off-the-shelf' libraries. In *Proceeding of the 21st International Conference on Information Visualisation (IV)* (2017), IEEE, pp. 281–287. doi:10.1109/iV.2017.77. 1

[WBV14] Waldner M., Bruckner S., Viola I.: Graphical histories of information foraging. In *Proceedings of the 8th Nordic Conference on Human-Computer Interaction (NordiCHI)* (2014), ACM, pp. 295–304. doi:10.1145/2639189.2641202. 2

[WPL*10] Waldner M., Puff W., Lex A., Streit M., Schmalstieg D.: Visual links across applications. In *Proceedings of Graphics Interface (GI)* (2010), pp. 129–136. 1

[YS17] Yu B., Silva C. T.: VisFlow – web-based visualization framework for tabular data with a subset flow model. *IEEE Transactions on Visualization and Computer Graphics 23*, 1 (2017), 251–260. doi:10.1109/TVCG.2016.2598497. 2