Eurographics Conference on Visualization (EuroVis) (2013)
M. Hlawitschka and T. Weinkauf (Editors)

A Survey of Visualization Construction User Interfaces

Lars Grammel[1], Chris Bennett[1], Melanie Tory[1], and Margaret-Anne Storey[1]

[1]Department of Computer Science, University of Victoria, Canada

Abstract

We have systematically surveyed the publications on visualization construction user interfaces that have been published in 12 major Visualization and HCI venues. We found six different visualization construction approaches (visual builder, visualization spreadsheet, textual programming, visual dataflow programming, template editor, and shelf configuration). The approaches differ in their flexibility, whether they support presentation or exploration tasks, and the spatial, temporal, and conceptual distance between the user interface (UI) and the visualization. Our results provide guidance to designers of visualization construction UIs.

Categories and Subject Descriptors (according to ACM CCS): H.5.2 [Computer Graphics]: User Interfaces—Graphical User Interfaces (GUI)

1. Introduction

As the demand for rapid visual data exploration as well as for engaging communication using custom visualizations grows, there is an increasing need to design visualization tools that allow users to choose visual structures and mappings (*visualization construction*). We define visualization construction as the central step from data tables to visual structures in the visualization reference model by Card et al. [CMS99] (Figure 1). This model outlines an iterative process in which raw data is transformed into rendered views. At each stage, the user may be involved, choosing or creating suitable transformations and mappings. The visualization construction task can be very challenging [GTS10, HCL05]. Our own past work showed that novices encounter barriers at every stage of the visualization construction process, including data selection, visual mapping, and interpretation [GTS10]. So, what UI approaches can best support visualization construction?

In this paper, we aim to understand what types of visualization construction UIs exist, and what commonalities, differences, and trade-offs they have. Our results identify interesting research directions to explore and should help designers to create visualization construction tools that match their use cases. To achieve our objective, we have systematically reviewed the literature on visualization construction UIs for both specifying visual structures and creating visual mappings (Section 2). This led to the identification of six distinct visualization construction approaches (Section 3). We describe how the approaches relate to data presentation and data analysis and examine the distances between construction UIs and visualizations in Section 4. We conclude with the main use cases of the different approaches in Section 5.

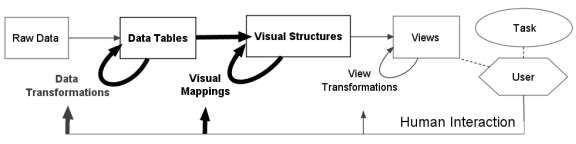

Figure 1: *Visualization Reference Model [CMS99]*

2. Literature Survey Method

We reviewed the literature in a systematic way with documented selection criteria and process to support the reproducibility and extensibility of our findings. For brevity, we only outline the selection criteria and process here. A detailed description and a full categorization of the publications is available in the supplementary file. We selected **full research papers** (6 pages and more) **that appeared in 12 major Visualization and HCI venues** (Vis, InfoVis, VAST, PacificVis, EuroVis, CHI, UIST, IUI, AVI, TVCG, TOCHI, IV) **between 1990 and 2012**. Our scope was limited to standard desktop computing platforms with mouse/keyboard-input. We focused primarily on tools that create single 2D visualizations composed of discrete high-level graphic elements, although some tools also produced more complex visualizations.

DOI: 10.2312/PE.EuroVisShort.EuroVisShort2013.019-023

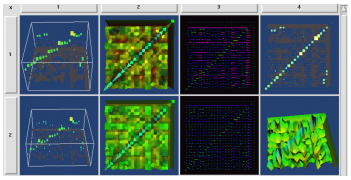

Figure 2: *Visualization Spreadsheet Example [CKBR97]*

Initially, the first author went through all the proceedings and journal issues, and pre-selected papers based on title, abstract and UI screenshots to identify the publications that matched our criteria. 282 full research papers were pre-selected and further filtered based on their full content. Each selected paper was then read by the first and one other author. The content was reviewed in detail and a final decision was made based on whether or not the paper matched the selection criteria. Our final selection consisted of those 64 publications that contain full construction approaches. The visualization construction approaches described in the publication were then identified and added to the classification. The classification was created in an iterative and exploratory way as we reviewed more and more publications.

3. Findings

A *visualization construction approach* is a cohesive UI type that supports the creation of complete visualizations. Visualization construction approaches are composed of lower-level techniques, e.g. UI elements for specific types of color mappings. We identified six major visualization construction approaches in our literature review.

3.1. Visualization Spreadsheet

Visualization spreadsheets display a matrix of visualizations (Figure 2). They facilitate the rapid comparison and adjustment of different visual mapping settings. There are two variations of visualization spreadsheets that are different in how the visualizations are modified. In the first variant, a few specific values of two configuration settings (e.g. visual mappings) are shown as rows and columns, and the cells contain visualizations of their combinations (while leaving other configuration settings fixed) [JKM00, JKM01]. When the user selects a cell, row or column from the spreadsheet, a value for that setting is selected and options for a new setting can be displayed instead. The second variant of visualization spreadsheets [CKBR97] allows the user to select cells, rows, and columns, and to apply operations to them (such as loading data, manipulating the content of cells, or setting the visual mappings). Visualization spreadsheets often use other techniques such as textual programming languages for defining operators and scripts to augment their functionality.

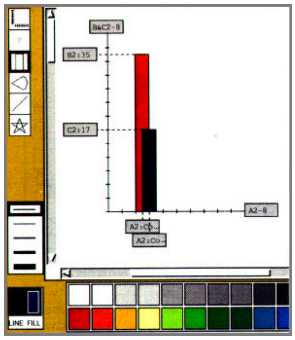

Figure 3: *Visual Builder Example [MGG94]*

3.2. Visual Builder

A visual builder interface (Figure 3) for visualization construction consists of a palette containing visual element prototypes and an assembly area. The UI concept is similar to graphics editor software such as Adobe Photoshop. It facilitates the construction of custom visualizations by enabling the user to put different visual elements together and to map data to them. The user selects visual elements from the palette, e.g. rectangles, and adds them to the assembly area. The palette can contain atomic visual elements [MGG94], composites [RKMG94], or control elements such as linked axes [CvW11]. Visual elements in the assembly area can typically be moved and resized using direct manipulation. Constructing the layout can be supported with guides, grids and constraints [CSS97]. The assembly area can show either a model of the visualization [CSS97, PvW08, RKMG94], a preview of what the visualization would look like [MGG94], or the actual visualization [YMSJ05, CvW11]. Additional dialogs and property boxes are often used to support the detailed configuration of visual elements and visual mappings, e.g. [PvW08, YMSJ05].

3.3. Textual Programming

Any regular programming language that provides access to the graphics system and to data storage can be used to create visualizations. Concepts and algorithms for creating visualizations can be encapsulated in libraries [FHL10, HCL05] and domain-specific languages (DSLs) [CSS97, Ada06, SDW09]. These libraries and DSLs can provide support for some specific visualizations, e.g. treemaps and maps [SDW09], or for many different types of visualizations [BH09, BOH11]. The flexibility of programming languages and paradigms has led to a variety of ways to create visualizations using textual programming. The languages differ in their accessibility and include languages that could be used by non-programmers (e.g. [WH11]). It is beyond the scope of this research to describe all the different trade-offs of these programming notations, e.g. using the cognitive dimensions framework [GP96]. With regards to visualization

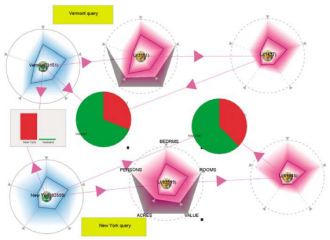

Figure 4: *Visual Dataflow Example [EST07]*

construction, the surveyed environments differ in the extent to which they are embedded into larger visualization systems. For example, the user can be supported by providing easy access to example programs for modification (scaffolding) and by checking for potential high-level visualization problems in the modified programs [EF94]. There are also differences between the libraries and DSLs in how tightly the definition of visual structure is coupled with the definition of visual mappings, in the available degree of visualization structure specification, and in the way data can be selected when defining visual mappings.

3.4. Visual Dataflow Programming

The dataflow programming approach for visualization construction is based on the idea that operators change the data along a pipeline until it is entered into visualizations. In visual dataflow environments, data sources, operators, and visualization models are typically represented as nodes which get connected through edges to form a data flow (Figure 4). The operators transform the data from the data sources before passing the data into the visualizations. Visual dataflow environments have been very prominent in scientific visualization (e.g. modular visualization environments), but they have been used for information visualization as well [FFIT00, KC96, RC03]. Williams et al. presented a classification of the elements of visual dataflow systems for visualization, including a discussion of design decisions and trade-offs [WRH92]. With regard to visualization construction, the main differences are whether previews of the visualization are shown as part of the dataflow [EST07, RC03] or not [FFIT00, KC96], and to what extent operators have a visualization representation as in [EST07]. Since the visual dataflow itself is often not important for analyzing the visualization, a mode that hides the visual dataflow is available in some tools, e.g. [RC03]. While UIs in this category usually represent the dataflows as node-link diagrams to the user, other representations such as spreadsheets [NB00] or lists of operators [IMI*10] can be used. There are typically a vast number of potential dataflows that can be assembled and thus it has become important to automatically suggest pipeline parts or full pipelines given partial pipelines [IMI*10, Koo08, SVK*07, TVW00].

3.5. Template Editor

In this approach, the user selects the data to visualize and then picks a visual structure in which to represent it. The main distinguishing criteria of this approach are the separation between the initial visualization selection steps and the refinement of the selected visualization. The selection of the visual structure can be part of a wizard, e.g. as in many popular spreadsheet applications such as Microsoft Excel, but it can also be done by selecting a menu item or a toolbar button. The extent to which the created visualizations can be configured without having to go back to selecting a new visual structure varies between not allowing for any tuning [AA98], allowing for changing some mappings and configuring some parameters [McK09, VWvH*07], and allowing for the reconfiguration of the visual mappings [CWT*08]. If the approach is integrated into a different approach such as shelf configuration configuration, a flexible reconfiguration of the visualization is possible without having to select the visual structure again [MHS07].

3.6. Shelf Configuration

The user configures the visualization by specifying the visual mappings to a fixed set of visual properties and by configuring additional options, both of which are exposed in a UI with a fixed layout. For example, the UI for the Polaris/Tableau table algebra exposes axis, retinal property, grouping, sorting and layer shelves for configuring the visual attribute mappings and dropdowns for selecting the mark type [SH00, STH02]. The key difference to the visual builder approach is that the visual structure composition is not exposed to the user. The user is instead restricted to the part of the visualization design space that is standardized and exposes the available visual properties in the UI.

4. Discussion

In this section, we describe how the approaches relate to data presentation and data analysis, examine the distances between construction UIs and visualizations, and discuss the limitations of this literature survey.

4.1. Data Presentation vs. Data Exploration

The two main use cases of visualization are data presentation, i.e. creating visualization to communicate insights, and data exploration, i.e. creating visualizations to understand the data and to find insights. We found two distinct ways of specifying visual mappings in the literature review. In *data-driven visual mapping*, the user selects a data attribute or element and assigns it to a visual attribute or element. In *visualization-driven visual mapping*, the user starts with a visual element or property and assigns a data element to it. While these two ways of specifying visual mappings appear to be very similar, we believe that the order in which they require decisions to be made (data-driven: first data attribute,

then visual property; visualization-driven: first visual property, then data attribute) needs to fit the user's mental processing for her/his task. A mismatch might impact the user's workflow. For example, it might be that a data-driven mapping construction works well for rapid visual data analysis where the user's focus is on understanding data, but not for visual communication tasks where the user's focus lies on creating a visual design to present already determined insights to others.

While the same visualization type can be used for both purposes, each purpose has different construction requirements. Data exploration depends on rapid data-centric visualization construction, whereas data presentation requires freedom in designing and configuring visual form:

Data Presentation is supported best by the *visual builder* and the *template editor* approaches. The former is better suited for the creation of custom visualization, whereas the latter is useful for presenting data in a common visual structure. If the focus is on the visual mappings only, the *visualization spreadsheet* approach is useful as well. When more flexibility in interaction and graphic design is required, *textual programming* is an option if the effort is warranted by the benefits.

Data Exploration of structured data in a standardized format is best supported by the the *Shelf Configuration*. When more data exploration flexibility in the analysis is required, *visual dataflow programming* is a good alternative. For non-standard data sets and analysis problems, one can apply *textual programming*.

4.2. Distance between UI and Visualization

It is important for users to understand how changes they make to the visualization specification affect the visualization produced. We have found that three kinds of distances influence how easy it is to gain this understanding: **temporal distance** between manipulating the specification and seeing the changes in the visualization, **spatial distance** between the construction UI and the visualization, and **conceptual distance** between the concepts exposed in the construction UI and the concepts that the visualization is made up from. For all three kinds of distances, reducing the gap between the visualization construction UI and the actual visualization should be beneficial, because it helps the user to relate his or her actions to their effect on the visualization. The relevance of these distances is supported by some well established principles in HCI input design. For example, direct manipulation and dynamic queries [AWS92] are generally recommended (to reduce spatial distance and temporal distance respectively), and there is empirical evidence that the perceptual structure of a task should match the control structure of the input device [JSMM94] (to reduce conceptual distance).

4.3. Limitations

There are several limitations to the approaches presented in this survey. There is some overlap between the approaches, and there are cases were there is no clear dividing line. The approaches themselves are based on a systematic literature review and discussions between two researchers with a computer science background on how to classify the individual papers. However, others might arrive at a slightly different categorization into approaches. Finally, the set of approaches is not exhaustive - for example recombining lower-level elements of the approaches might yield new approaches.

5. Conclusion

Specifying visual structures and mappings is an important aspect of constructing visualizations. Through our systematic literature review, we identified six distinct visualization construction approaches. Each of the approaches has different strengths that fit particular use cases well:

Visualization Spreadsheets allow the user to incrementally apply and compare different visual mappings. This facilitates the *exploration of different visual mappings*.

Visual Builders provide great flexibility in the visual structures that the user can create. This enables the *creation of custom visualizations for presentation purposes*.

Textual Programming gives the programmer full freedom to design any kind of visualizations and interactions and to handle data in any format. This enables the *creation of custom interactive visualizations and the exploration of data in non-standard ways*. However, it comes at a cost of high temporal and spatial, and often conceptual, distance.

Visual Dataflow Programming lets users rapidly assemble data transformation and visualization pipelines. It can be used to *transforming non-standard data using several operations and rendering it in standard visualizations*.

Template Editor enables the quick construction of standard visualizations. Its use case is *presenting structured data in standard visualizations*.

Shelf Configuration has a low conceptual and temporal distance. It is aimed at *rapid data exploration of data sets in standard formats*.

The approaches exhibit different challenges to the user and fit at different points in the spectrum between data analysis and presentation. Based on our findings and the discussion, we believe that the empirical comparison and the recombination of different approaches and their elements are fruitful research directions to explored.

Acknowledgements

We thank the VisID group and the reviewers for their feedback. The first author was supported by an IBM CAS PhD fellowship.

References

[AA98] ANDRIENKO G. L., ANDRIENKO N. V.: Intelligent visualization and dynamic manipulation: two complementary instruments to support data exploration with gis. In *AVI '98* (1998), ACM, pp. 66–75. 3

[Ada06] ADAR E.: Guess: a language and interface for graph exploration. In *CHI '06* (2006), ACM, pp. 791–800. 2

[AWS92] AHLBERG C., WILLIAMSON C., SHNEIDERMAN B.: Dynamic queries for information exploration: an implementation and evaluation. In *CHI '92* (1992), ACM, pp. 619–626. 4

[BH09] BOSTOCK M., HEER J.: Protovis: A graphical toolkit for visualization. *IEEE TVCG 15* (November 2009), 1121–1128. 2

[BOH11] BOSTOCK M., OGIEVETSKY V., HEER J.: D3 data-driven documents. *IEEE TVCG 17*, 12 (2011), 2301–2309. 2

[CKBR97] CHI E. H.-H., KONSTAN J., BARRY P., RIEDL J.: A spreadsheet approach to information visualization. In *UIST '97* (1997), ACM, pp. 79–80. 2

[CMS99] CARD S. K., MACKINLAY J. D., SHNEIDERMAN B. (Eds.): *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufmann, USA, 1999. 1

[CSS97] CASTELLS P., SZEKELY P., SALCHER E.: Declarative models of presentation. In *IUI '97* (1997), ACM, pp. 137–144. 2

[CvW11] CLAESSEN J. H. T., VAN WIJK J. J.: Flexible linked axes for multivariate data visualization. *IEEE TVCG 17*, 12 (Dec. 2011), 2310–2316. 2

[CWT*08] CHAN B., WU L., TALBOT J., CAMMARANO M., HANRAHAN P.: Vispedia: Interactive visual exploration of wikipedia data via search-based integration. *IEEE TVCG 14* (November 2008), 1213–1220. 3

[EF94] EISENBERG M., FISCHER G.: Programmable design environments: integrating end-user programming with domain-oriented assistance. In *CHI '94* (1994), ACM, pp. 431–437. 3

[EST07] ELMQVIST N., STASKO J., TSIGAS P.: Datameadow: A visual canvas for analysis of large-scale multivariate data. In *VAST '07* (2007), IEEE, pp. 187–194. 3

[FFIT00] FUJISHIRO I., FURUHATA R., ICHIKAWA Y., TAKESHIMA Y.: Gadget/iv: A taxonomic approach to semi-automatic design of information visualization applications using modular visualization environment. In *INFOVIS '00* (2000), IEEE. 3

[FHL10] FORBES A., HÖLLERER T., LEGRADY G.: behaviorism: a framework for dynamic data visualization. *IEEE TVCG 16*, 6 (2010), 1164 –1171. 2

[GP96] GREEN T., PETRE M.: Usability analysis of visual programming environments: A cognitive dimensions framework. *Journal of Visual Languages and Computing 7* (1996), 131–174. 2

[GTS10] GRAMMEL L., TORY M., STOREY M.-A.: How information visualization novices construct visualizations. *IEEE TVCG 16* (2010), 943–952. 1

[HCL05] HEER J., CARD S. K., LANDAY J. A.: prefuse: a toolkit for interactive information visualization. In *CHI '05* (2005), ACM, pp. 421–430. 1, 2

[IMI*10] INGRAM S., MUNZNER T., IRVINE V., TORY M., BERGNER S., MÖLLER T.: Dimstiller: Workflows for dimensional analysis and reduction. In *VAST 2010* (Oct. 2010), pp. 3 –10. 3

[JKM00] JANKUN-KELLY T. J., MA K.-L.: A spreadsheet interface for visualization exploration. In *VIS '00* (2000), IEEE Press, pp. 69–76. 2

[JKM01] JANKUN-KELLY T. J., MA K.-L.: Visualization exploration and encapsulation via a spreadsheet-like interface. *IEEE TVCG 7* (July 2001), 275–287. 2

[JSMM94] JACOB R. J. K., SIBERT L. E., MCFARLANE D. C., MULLEN JR. M. P.: Integrality and separability of input devices. *ACM Trans. Comput.-Hum. Interact. 1* (March 1994), 3–26. 4

[KC96] KAZMAN R., CARRIERE J.: Rapid prototyping of information visualizations using vanish. In *INFOVIS '96* (1996), IEEE. 3

[Koo08] KOOP D.: Viscomplete: Automating suggestions for visualization pipelines. *IEEE TVCG 14* (November 2008), 1691–1698. 3

[McK09] MCKEON M.: Harnessing the information ecosystem with wiki-based visualization dashboards. *IEEE TVCG 15* (2009), 1081–1088. 3

[MGG94] MYERS B. A., GOLDSTEIN J., GOLDBERG M. A.: Creating charts by demonstration. In *CHI '94* (USA, 1994), ACM, pp. 106–111. 2

[MHS07] MACKINLAY J., HANRAHAN P., STOLTE C.: Show me: Automatic presentation for visual analysis. *IEEE TVCG 13* (November 2007), 1137–1144. 3

[NB00] NUNEZ F., BLAKE E.: Vissh: A data visualisation spreadsheet. *Data Visualization 2000* (2000), 209–218. 3

[PvW08] PRETORIUS A., VAN WIJK J.: Multiple views on system traces. In *PacificVIS'08* (2008), IEEE, pp. 95–102. 2

[RC03] ROSS G., CHALMERS M.: A virtual workspace for hybrid multidimensional scaling algorithms. In *INFOVIS 2003* (2003), pp. 91 –96. 3

[RKMG94] ROTH S. F., KOLOJEJCHICK J., MATTIS J., GOLDSTEIN J.: Interactive graphic design using automatic presentation knowledge. In *CHI '94* (1994), ACM, pp. 112–117. 2

[SDW09] SLINGSBY A., DYKES J., WOOD J.: Configuring hierarchical layouts to address research questions. *IEEE TVCG 15* (November 2009), 977–984. 2

[SH00] STOLTE C., HANRAHAN P.: Polaris: A system for query, analysis and visualization of multi-dimensional relational databases. In *INFOVIS '00* (2000), IEEE, pp. 5–. 3

[STH02] STOLTE C., TANG D., HANRAHAN P.: Polaris: a system for query, analysis, and visualization of multidimensional relational databases. *IEEE TVCG 8*, 1 (January 2002), 52 –65. 3

[SVK*07] SCHEIDEGGER C., VO H., KOOP D., FREIRE J., SILVA C.: Querying and creating visualizations by analogy. *IEEE TVCG 13* (November 2007), 1560–1567. 3

[TVW00] TELEA A., VAN WIJK J.: Smartlink: An agent for supporting dataflow application construction. *Proceedings of IEEE VisSym* (2000), 189–198. 3

[VWvH*07] VIEGAS F. B., WATTENBERG M., VAN HAM F., KRISS J., MCKEON M.: Manyeyes: a site for visualization at internet scale. *IEEE TVCG 13* (November 2007), 1121–1128. 3

[WH11] WICKHAM H., HOFMANN H.: Product plots. *IEEE TVCG 17*, 12 (Dec. 2011), 2223–2230. 2

[WRH92] WILLIAMS C., RASURE J., HANSEN C.: The state of the art of visual languages for visualization. In *VIS '92* (1992), IEEE Press, pp. 202–209. 3

[YMSJ05] YI J. S., MELTON R., STASKO J., JACKO J. A.: Dust & magnet: multivariate information visualization using a magnet metaphor. *Information Visualization 4* (October 2005), 239–256. 2