Reactive Data Visualizations

Curran Kelleher, Haim Levkowitz
Computer Science Department
University of Massachusetts
Lowell, USA
https://github.com/curran/portfolio
http://www.cs.uml.edu/~haim/

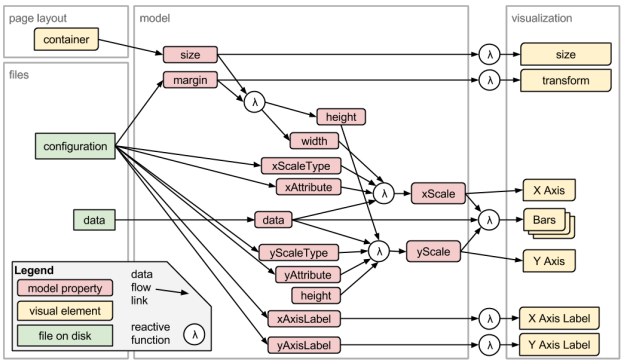


Fig. 1. The data flow graph for a reactive bar chart based on *reactive models*, our core contribution.

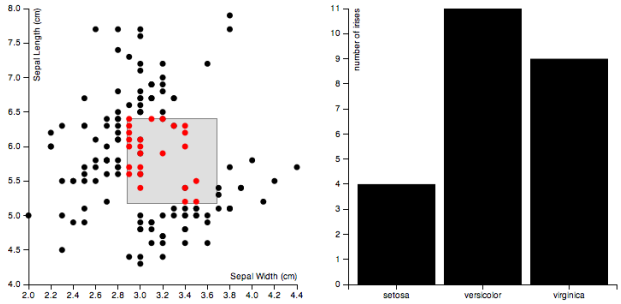


Fig. 2. An example of linked views, powered by reactive visualization components. Brushing to select records in the scatter plot causes the selected data to be aggregated and displayed in the bar chart.

***Abstract*—Managing complex data flows and update patterns is one of the most difficult challenges in interactive data visualization. For example, constructing interactive visualizations with multiple linked views can be a daunting task. Functional reactive programming provides approaches for declaratively specifying data dependency graphs and maintaining them automatically. We argue that functional reactive programming is an appropriate and effective abstraction for interactive data visualization. We demonstrates the effectiveness of our proposed approach in several visualization examples including multiple linked views. We also provide a catalog of reusable reactive visualization components.**

***Keywords*-multiple linked views; interaction technques; information visualization;**

I. Introduction

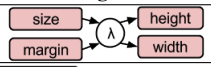
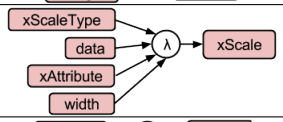
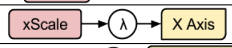
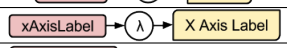
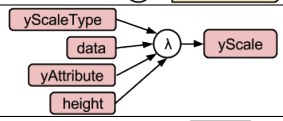
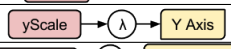
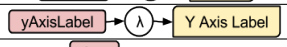
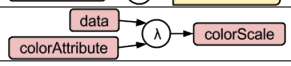
Constructing interactive visualizations is a complex task. The task becomes even more complex when multiple visualizations are presented and linked together through interactions. The issue at the core of interactive visualization and linked views is management of complex data flows and update patterns. Even with the wealth of visualization toolkits and libraries that exist today, there is a need for an abstraction that addresses these core issues. The contribution of this paper is a novel approach for developing reusable interactive visualization components using concepts from functional reactive programming.

Interactions within data visualization environments have been well studied. Becker et al. investigated brushing in scatter plots [1]. Shneiderman et al. explored dynamic queries in general and how these operations fit into a larger context of visual information seeking [2]. Ward introduced a visualization system based on multiple linked views with direct manipulation techniques including brushing and linking [3]. Interactive data visualizations can be linked together such that interactions in one visualization cause updates in another visualization. This technique is referred to as "multiple linked views" [4] and "brushing and linking" [5], [6].

We introduce a novel way to combine elements of functional reactive programming with the Model View Controller (MVC) paradigm to create what we call *reactive models*. These reactive models can serve as a foundation for reusable interactive visualization components. This approach overcomes limitations of traditional MVC frameworks, and is simpler than using a full blown functional reactive programming framework.

For complex applications such as interactive visualizations, managing propagation of changes can quickly become complex. To provide a solid foundation for dynamic visualization systems, models should be able function in the context of data dependency graphs. Developers should be able to declaratively specify data dependencies, and change propagation should be automatically managed. The $when$ operator from functional reactive programming propagates changes from one or more reactive functions (such as is found in the JavaScript libraries Bacon.js and RXJS). We apply the $when$ operator to our simple key-value models to yield what we call *reactive models*

TABLE I
REUSABLE REACTIVE VISUALIZATION COMPONENTS.

Component	Diagram	Description
margin		Computes the size of the inner visualization rectangle based on the container size (which may change when the user resizes the visualization) and the configured margin.
xScale		Computes the X scale. The domain is computed from the input data by evaluating the X attribute bounds. The range is computed from the inner visualization width.
xAxis		Renders the X Axis (the center line, tick marks and tick labels) from the X scale.
xAxisLabel		Renders the text label for the X Axis.
yScale		Computes the Y scale. The domain is computed from the input data by evaluating the Y attribute bounds. The range is computed from the inner visualization width.
yAxis		Renders the Y Axis (the center line, tick marks and tick labels) from the Y scale.
yAxisLabel		Renders the text label for the Y Axis.
colorScale		Computes the color scale. The domain is computed from the input data by evaluating the set of unique values foun in the color attribute.

II. REACTIVE VISUALIZATIONS

Interactive visualizations must respond to changes made by users such as resizing of the display, changes in the data driving the visualization, changes in configuration, and updates from other visualizations in a linked view context. We introduce reusable reactive visualization components, shown in table I, and show how they can be composed to form interactive visualizations and linked views.

A. Visualization Primitives

Consider visualizations such as the bar chart, line chart, stacked area chart, parallel coordinates and choropleth map. These visualizations share many underlying primitives such as scales, axes, margins and labels [7]. Interactive forms of these visualizations also share interaction techniques for selecting visual marks such as rectangular brushing, hovering, clicking, panning and zooming. These visualization primitives can be encapsulated as data dependency subgraphs within reactive models which we call *components*.

Table I shows a listing of components that can be combined to easily generate a foundation for a variety of interactive visualizations. These components encapsulate reactive data dependency subgraphs that implement the visualization primitives necessary for interactive visualizations. Figure 1 shows how several of these components can be assembled to create a general-purpose reactive bar chart.

Figure 2 shows an example of linked views using our approach. Here we are using a scatter plot with brushing interaction assembled using reusable components in a similar fashion to the bar chart shown in figure 1.

III. CONCLUSION

We introduce a novel way to combine elements of functional reactive programming with the Model View Controller (MVC) paradigm to create what we call *reactive models*. These reactive models allow developers to declaratively specify data dependency graphs. This kind of abstraction is well suited for developing interactive visualizations because it drastically simplifies management of complex data flows and update patterns.

Future directions for this work will focus on developing a catalog of reusable visualization components, coupling the data to currently available public data sources, developing visualization-centric user interfaces and collaboration. So far we have applied our technique to bar charts and scatter plots only, however we intend to also support the following visualizations: Table, Color Legend, Line Chart, Pie Chart, Choropleth Map, Parallel Coordinates, Heatmap, Stacked Bar Chart, Stacked Area Chart, Streamgraph, TreeMap, and Force Directed Graph Layout.

An open source implementation of reactive models is available at `github.com/curran/model`, and a catalog of example visualizations including bar chart, scatter plot, line chart and table is available at `https://github.com/curran/model-contrib`.

REFERENCES

[1] R. A. Becker and W. S. Cleveland, "Brushing scatterplots," *Technometrics*, vol. 29, no. 2, pp. 127–142, 1987.

[2] B. Shneiderman, "Dynamic queries for visual information seeking," *Software, IEEE*, vol. 11, no. 6, pp. 70–77, 1994.

[3] M. O. Ward, "Xmdvtool: Integrating multiple methods for visualizing multivariate data," in *Proceedings of the Conference on Visualization'94*. IEEE Computer Society Press, 1994, pp. 326–333.

[4] J. C. Roberts, "Exploratory visualization with multiple linked views," 2004.

[5] D. A. Keim, "Information visualization and visual data mining," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 8, no. 1, pp. 1–8, 2002.

[6] L. Anselin, I. Syabri, and O. Smirnov, "Visualizing multivariate spatial correlation with dynamically linked windows," *Urbana*, vol. 51, p. 61801, 2002.

[7] M. Bostock, V. Ogievetsky, and J. Heer, "D3: Data-driven documents," *IEEE Trans. Visualization & Comp. Graphics (Proc. InfoVis)*, 2011. [Online]. Available: http://vis.stanford.edu/papers/d3