Files\\Literature\\Bergel et al. - 2014 - A Domain-Specific Language for Visualizing Softwar - § 4 references coded [1.10% Coverage]

Reference 1 - 0.21% Coverage

A node represents a software element that represents a particular entity of the software system.

Reference 2 - 0.28% Coverage

An edge represents a directed relationship between two nodes, typically representing a dependency between two software entities.

Reference 3 - 0.34% Coverage

Nodes have to be properly spatially located to communicate a particular situation. A large number of layouts exists, for which some of them are driven by edges.

Reference 4 - 0.27% Coverage

Global rules may be set to either avoid code duplications between production rules or to perform color or size normalization.

Files\\Literature\\Bostock and Heer - 2009 - Protovis A Graphical Toolkit for Visualization - § 3 references coded [0.29% Coverage]

Reference 1 - 0.09% Coverage

The first task when using Protovis is to decompose the desired visualization into its constituent marks

Reference 2 - 0.16% Coverage

Panels allow repeated or nested structures, commonly used in small multiple displays [30] where a small visualization is tiled to facilitate comparison across one or more dimensions.

Reference 3 - 0.04% Coverage

graphical primitives provided by Protovis.

Files\\Literature\\Bostock et al. - 2011 - D³ Data-Driven Documents - § 1 reference coded [0.54% Coverage]

Reference 1 - 0.54% Coverage

D3’s atomic operand is the selection: a filtered set of elements queried from the current document. Operators act on selections, modifying content. Data joins bind input data to elements, enabling functional operators that depend on data, and producing enter and exit subselections for the creation and destruction of elements in correspondence with data. While operators apply instantaneously by default, animated transitions interpolate attributes and styles smoothly over time. Special operators called event handlers respond to user input and enable interaction. Numerous helper modules, such as layouts and scales, simplify common visualization tasks.

Files\\Literature\\Heer and Bostock - 2010 - Declarative Language Design for Interactive Visual - § 1 reference coded [0.09% Coverage]

Reference 1 - 0.09% Coverage

Our Java system also contributes a novel design feature: declarative specification of animated transitions.

Files\\Literature\\Ledur et al. - 2017 - A High-Level DSL for Geospatial Visualizations wit - § 1 reference coded [0.12% Coverage]

Reference 1 - 0.12% Coverage

enables users to express filter, classification, data format, and visualization specifications.

Files\\Literature\\Li et al. - 2018 - ECharts A declarative framework for rapid constru - § 1 reference coded [0.11% Coverage]

Reference 1 - 0.11% Coverage

Users are allowed to freely configure components, styles, data and interactions through a declarative option.

Files\\Literature\\Liu et al. - 2021 - Boba Authoring and Visualizing Multiverse Analyse - § 6 references coded [0.38% Coverage]

Reference 1 - 0.08% Coverage

The basic language primitives in the Boba DSL consist of source code, placeholder variables, blocks, constraints, and code graphs.

Reference 2 - 0.05% Coverage

Source Code. The most basic ingredient of an annotated script is the source code

Reference 3 - 0.05% Coverage

Placeholder Variables. Placeholder variables are useful to specify decisions points

Reference 4 - 0.07% Coverage

Code Blocks. Code blocks (Fig. 3a, pink text) divide the source code into multiple snippets of one or more lines of code

Reference 5 - 0.06% Coverage

Constraints. By default, Boba assumes all combinations between decision points are valid.

Reference 6 - 0.07% Coverage

Code Graph. Users may further specify the execution order between code blocks as a directed acyclic graph (DAG),

Files\\Literature\\Logre and Déry-Pinna - 2018 - MDE in Support of Visualization Systems Design a - § 1 reference coded [0.28% Coverage]

Reference 1 - 0.28% Coverage

A meta-model captures the structure of visualization systems, as illustrated by the EMF11 implementation depicted in Fig. 5. It defines a dashboard as a layout ofweighted columns and lines, filled with abstract visualizations. The visualization intended capabilities are specified by using concerns extracted from the taxonomy.

Files\\Literature\\Logre et al. - 2014 - Sensor Data Visualisation A Composition-Based App - § 1 reference coded [0.53% Coverage]

Reference 1 - 0.53% Coverage

Restricting the domain to its essence, a designers works according to three dimensions while designing a dashboard: (i) the data involved in the dashboard according to her monitoring needs (i.e., “What am I visualising”), (ii) the different visualisation concerns applied to these data (i.e., “How do I visualise it?”) and finally (iii) the spatial and temporal layout of the dashboard (i.e., “Where and when do I visualise it?”).

Files\\Literature\\Morgan et al. - 2017 - VizDSL Towards a Graphical Visualisation Language - § 4 references coded [0.71% Coverage]

Reference 1 - 0.12% Coverage

1) Layout: Layouts are a way of specifying the arrangement of elements in the visualisation.

Reference 2 - 0.13% Coverage

2) Interactions: Interactions refer to the way in which a
user can interact with the visualisation.

Reference 3 - 0.29% Coverage

3) Mapping of semantic corresponding visualisation entities/relationships components: to Semantic visualisation refers to the capacity for visual presentations of data entities with associated meanings and relationships

Reference 4 - 0.17% Coverage

4) Hierarchical navigation: Hierarchical navigation is the ability to move through a visualisation on different abstraction levels.

Files\\Literature\\Rojas et al. - 2020 - Cities-Board A Framework to Automate the Developm - § 7 references coded [0.74% Coverage]

Reference 1 - 0.05% Coverage

The Dashboard is the main class of the metamodel

Reference 2 - 0.08% Coverage

The Visualizer class represents objects to present city information.

Reference 3 - 0.14% Coverage

A Visualizer presents information from a data source and has a position in the screen, which is represented by the class Position.

Reference 4 - 0.10% Coverage

The SocialMedia class represents objects that show information from cities' social networks.

Reference 5 - 0.11% Coverage

The Map class represents objects that use geolocation information to show information on the city.

Reference 6 - 0.09% Coverage

The Chart class represents objects that use plots to represent cities' information.

Reference 7 - 0.17% Coverage

The DataSource class represents the data entities that dash-
board presents using visualizers, which are identified by data source name and data provider.

Files\\Literature\\Satyanarayan and Heer - 2014 - Authoring Narrative Visualizations with Ellipsis - § 1 reference coded [0.19% Coverage]

Reference 1 - 0.19% Coverage

we identify a set of requisite storytelling abstractions, including state-based scene structure, coordinated visualizations, dynamic annotations and interactive triggers.

Files\\Literature\\Satyanarayan et al. - 2017 - Vega-Lite A Grammar of Interactive Graphics - § 2 references coded [0.14% Coverage]

Reference 1 - 0.08% Coverage

Second, we contribute a high-level interaction grammar. With Vega-Lite, an interaction design is composed of selections

Reference 2 - 0.06% Coverage

For added expressivity, Vega-Lite provides a series of operators to transform a selection.

Files\\Literature\\Smeltzer and Erwig - 2018 - A domain-specific language for exploratory data vi - § 1 reference coded [0.24% Coverage]

Reference 1 - 0.24% Coverage

First, it allows the composition of visualizations into larger visualizations. This could be spatial composition or something more sophisticated such as stacking chart elements. Second, it allows visualizations to be transformed between Cartesian and polar coordinate systems.

Files\\Literature\\Smeltzer et al. - 2014 - A transformational approach to data visualization - § 1 reference coded [0.45% Coverage]

Reference 1 - 0.45% Coverage

Instead of a single, canonical way to build a given visualization, we provide a number of layered abstractions from which to choose the most appropriate for the situation. All of them, however, fundamentally produce a structure of type Vis. At the lowest level, the user can create an individual mark. A mark is a single graphical element consisting of a particular primitive shape and a set of visual parameters. Visual parameters are any components of a mark which can be bound to data, such as height, width, color, and orientation. All of these data types are then collected together in a Vis.

Files\\Literature\\Vázquez-Ingelmo et al. - 2018 - Domain engineering for generating dashboards to an - § 1 reference coded [0.28% Coverage]

Reference 1 - 0.28% Coverage

The results are presented through three perspectives: the results obtained regarding the customization of functionalities, layout and data sources.