# DEFINITIONS

## Formal definition of a widget:

A widget is an instantiation of a particular interaction class from the “how-manipulate” level of the Brehmer and Munzner multi-level typology. It is an implementation of an interaction within a specific visualization interface. The user manipulates the widget to manipulate the data, and the user can see the result of their manipulations as updates to the visualization/interface.

## Less formal definition:

Basically, a widget is everything that allow users to manipulate data, and to see the result, being it a GUI element, or a particular technique that requires the user to perform some actions directly on the visualization.

# INTERACTION CLASSES

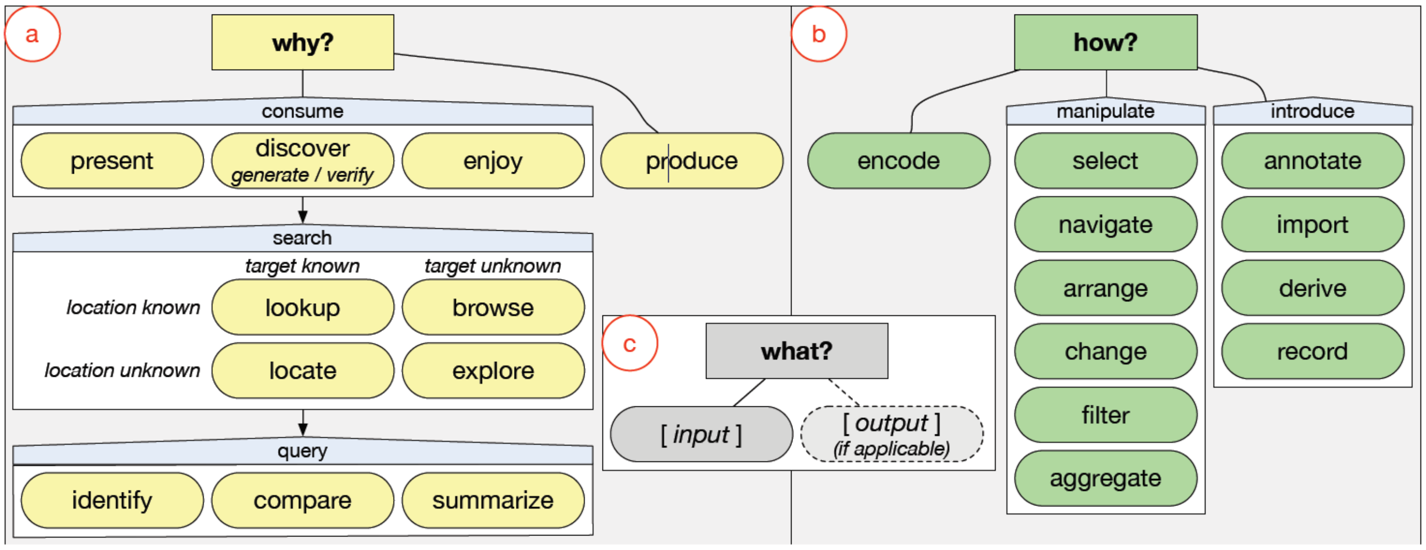


Fig. 1. Our multi-level typology of abstract visualization tasks. The typology spans *why*, *how*, and *what*; task descriptions are formed by nodes from each part: a) *why* a task is performed, from high-level (consume vs. produce) to mid-level (search) to low-level (query). b) *how* a task is executed in terms of *methods*, defined as families of related visual encoding and interaction techniques. c) *what* the task inputs and outputs are.

## MANIPULATE:

- **Select**: demarcation of one or more elements in a visualization, differentiating selected from unselected elements (e.g. clicking or lassoing elements in a single visualization and brushing methods used to highlight elements in visualization systems incorporating multiple linked views).
- **Navigate:** methods that alter a user's viewpoint (e.g. zooming, panning, and rotating). Includes other methods that trigger details on-demand views, combining navigate and select.
- **Arrange:** organizing visualization elements spatially. Some of these methods arrange representations of data, such as reordering the axes in a parallel coordinate plot or the rows and columns of a scatterplot matrix. Other methods allow users to coordinate the spatial layout of views.
- **Change:** alterations in visual encoding (e.g. altering the size and transparency of points in a scatterplot or edges in a node-link graph, altering a colour-scale or texture mapping, or transforming the scales of axes). Other methods have more pronounced effects, changing the chart type altogether, such as transitioning between grouped and stacked bar charts, or between linear and radial layouts for time-series graphs.
- **Filter:** methods that adjust the exclusion and inclusion criteria for elements in a visualization.
- **Aggregate:** methods that change the granularity of visualization elements.

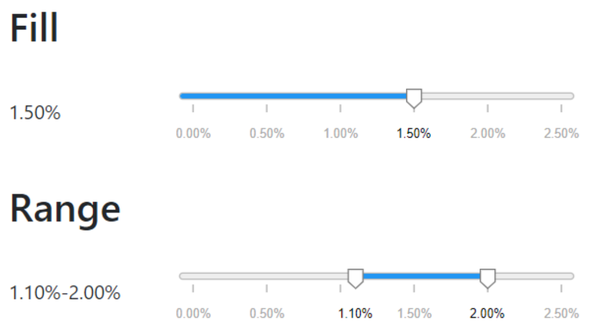
---

# TECHNIQUES

(sorted by events generation rate, highest to lowest)

---

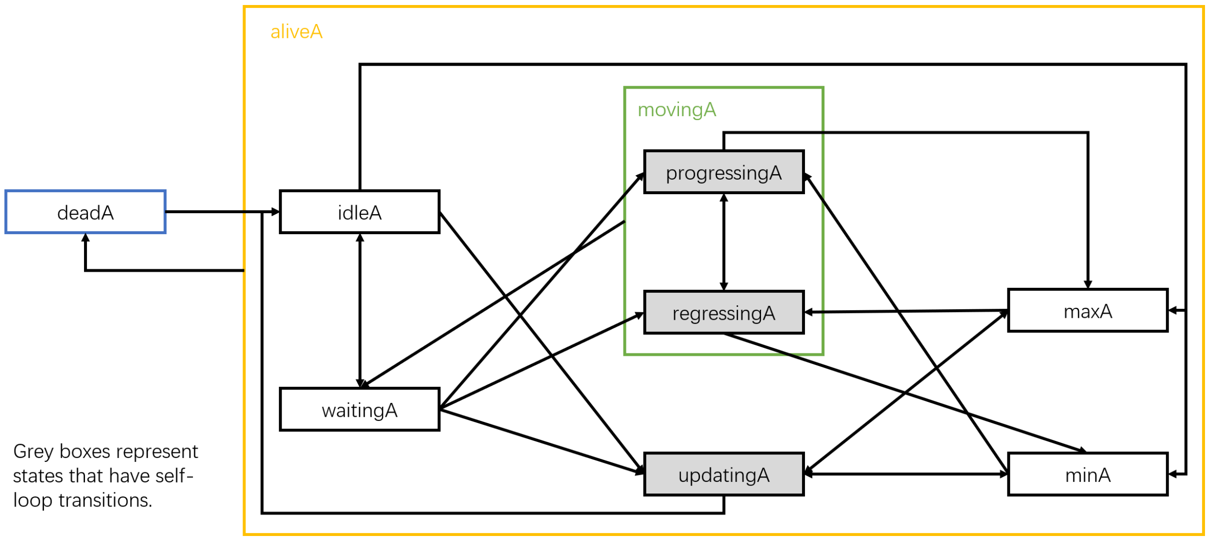
## SLIDER



- Interaction class: **Filter**.
- Usage: The slider typically represents a single data attribute. The user can drag either end of the slider to adjust the corresponding value range for the

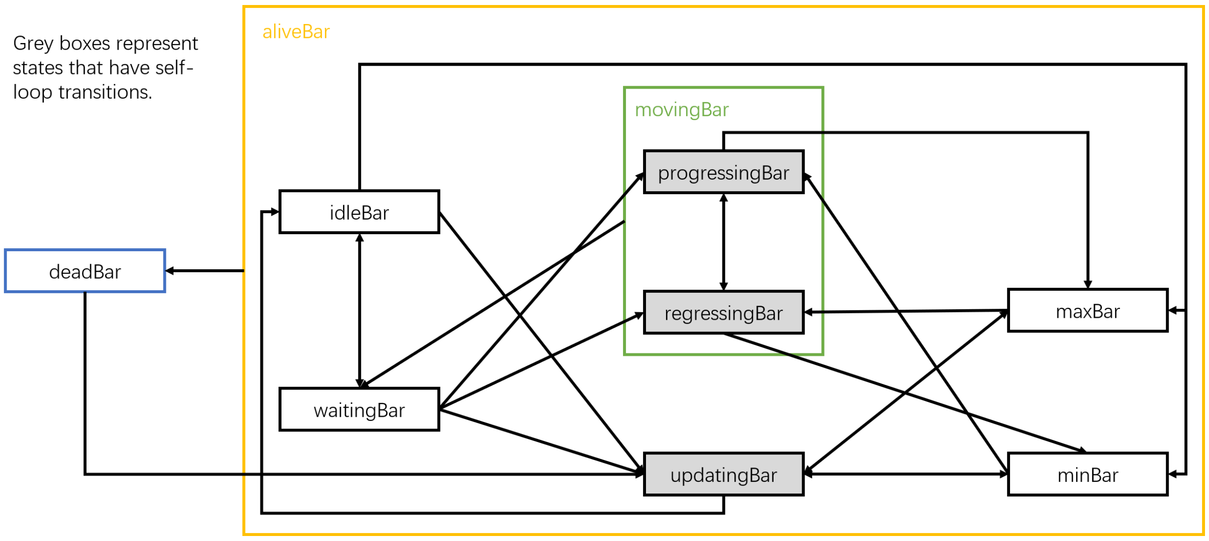
attribute, which filters out tuples that do not have a value within the target attribute range.

- Events generation rate: Really high.
- Exemplar implementation: d3 (containing every type of slider)
- Possible parameters for predictions:
  - Pixel position logs for the whiskers, to predict the drag trajectory.
  - Others...
- References (more details on **zotero->crossfilter-benchmark->interaction-widgets->sliders**):
  - *"Data visualization sliders";* Stephen G. Eick.
  - *"Dynamic queries for information exploration: an implementation and evaluation";* Christopher Ahlberg.
  - *"Dynamic queries for visual information seeking";* B. Shneiderman.
  - *"Dynamic queries: database searching by direct manipulation";* B. Shneiderman.
  - *"Empirical comparison of dynamic query sliders and brushing histograms";* Qing Li.
  - *"User preference extraction using dynamic query sliders in conjunction with UPS-EMO algorithm";* Timo Aittokoski.
  - *"Visual Information Seeking: Tight Coupling of Dynamic Query Filters with Starfield Displays";* Christopher Ahlberg.
- Statechart
  - Statechart for one handle

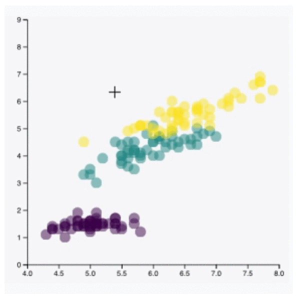


  - Statechart for the connecting bar

Grey boxes represent states that have self-loop transitions.

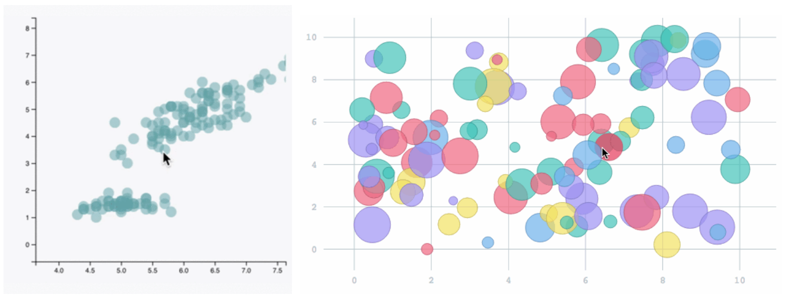


## BRUSHING

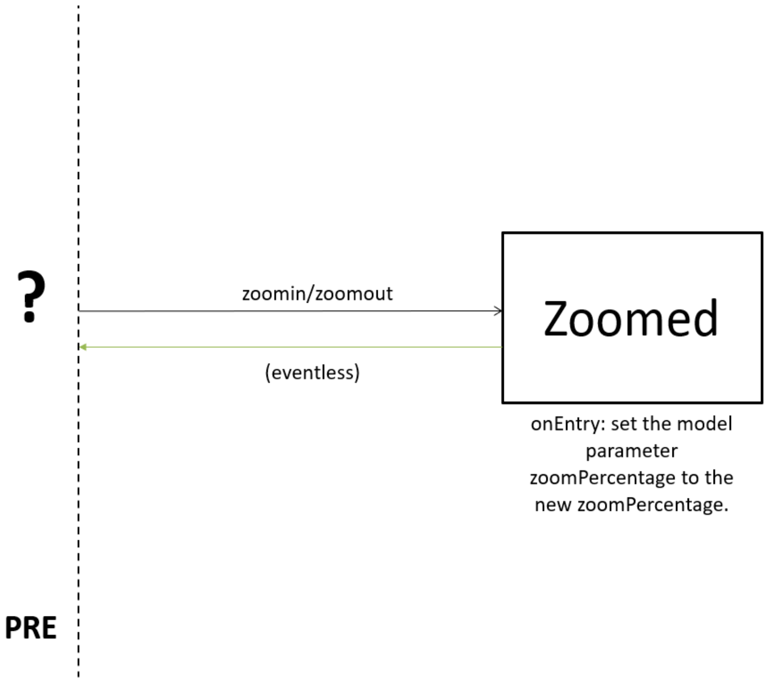


- Interaction class:
  - **Filter.**
  - **Select.**
- Usage: Users can click to start a new brush, drag around the mouse to modify the brush and finally release to end it. Often it is paired with other widgets like panning (as in the gif) or zoom.
- Events generation rate: It can depend on the implementation:
  - Really high if while brushing the visualization is updated real time.
  - A bit less high if the visualization is updated only on brush end.
- Exemplar implementation:
  - Brushing in d3.
  - Brushing in Vega.
- Possible parameters for predictions:
  - Mouse position (to predict brush start).
  - Drag direction (to predict values to add/remove from the current selection).
  - Drag speed?
- References
  - D3 official website.
  - Vega official website

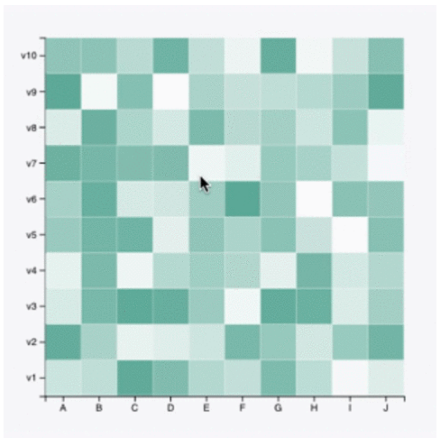
## ZOOM



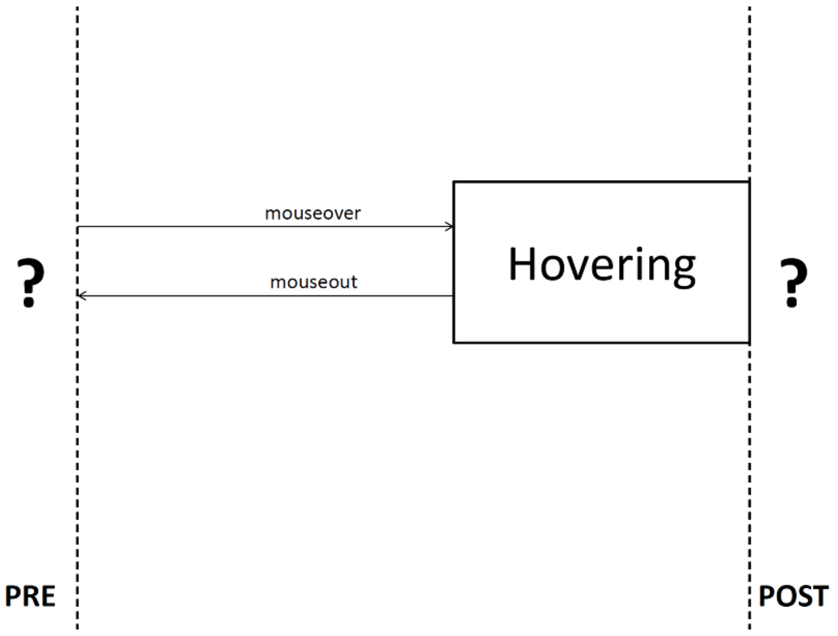
- Interaction class:     **Navigate.**
- Usage: Users can click on a point in the visualization, or use gestures on a touch screen or mouse pad, to zoom in, to get more detailed information about a section of the visualization.
- Events generation rate: It depends on how the zoom is actually implemented.
  - Really low if there are no new items to fetch.
  - A bit higher if, due to granularity changes, we need to fetch new items after a zomm has happened. If this is done real time, could it be hard?
- Exemplar implementation:
  - Zoom and panning d3 implementation.
- Possible parameters for predictions:
  - Mouse position.
  - The user already zoomed recently.
  - Mouse hovering, with time (to predict if the user is going to zoom).
- References
  - D3 official website.
  - Vega official website
- STATECHART (BuildingBlock)


?
zoomin/zoomout
(eventless)
Zoomed
onEntry: set the model parameter zoomPercentage to the new zoomPercentage.
PRE

## HOVERING

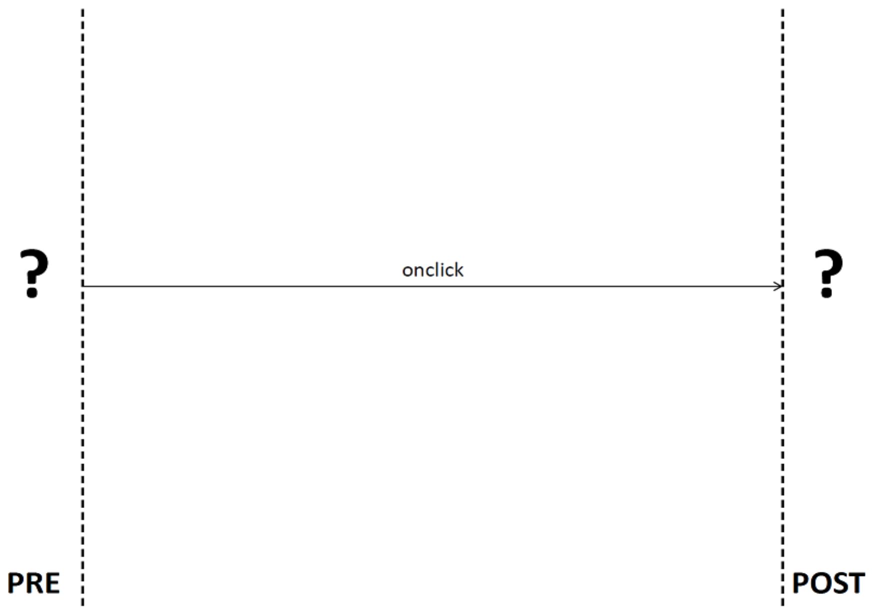


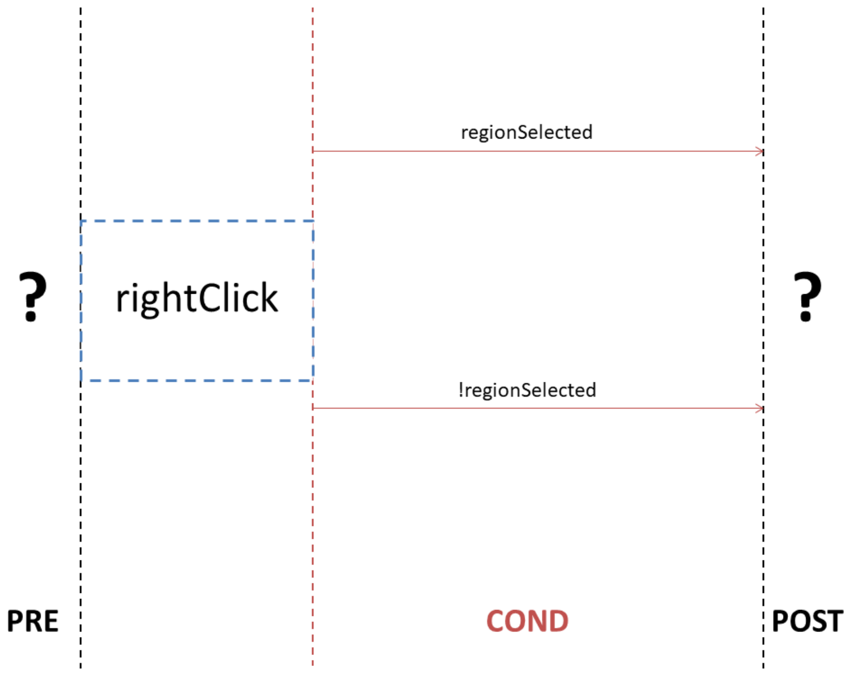
- Interaction class: **Select.**
- Usage: When the user hovers over an element of the visualization, some action is performed.
- Events generation rate: (medium to low depending on data density)
    - One selection query when the hovering happens.
    - If the hovering is continuous, there will be a sequence of queries.
- Exemplar implementation:
    - Implementing tooltip in d3.
- Possible parameters for predictions:
    - Mouse positions.
    - Hovering started on one element of the visualization.
    - Mouse speed?
- References
    - D3 official website.
    - Vega official website
- STATECHART (BuildingBlock)



## CLICK

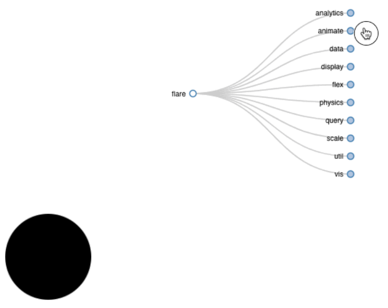
- Interaction class: **Select**.
- Usage: Users click on an element of the visualization, to perform some action.
- Events generation rate: Really low.
- Exemplar implementation:
    - Google maps based implementation (clicking is implemented on red markers).
- Possible parameters for predictions:
    - Mouse position.
    - Elements recently clicked.
    - Distance between the element clicked and the last one.
- References
    - Marco Angelini, Graziano Blasilli, Simone Lenti, Alessia Palleschi, and Giuseppe Santucci. 2020. CrossWidgets: Enhancing Complex Data Selections through Modular Multi Attribute Selectors. In Proceedings of the International Conference on Advanced Visual Interfaces (AVI '20). Association for Computing Machinery, New York, NY, USA, Article 12, 1–9. DOI:https://doi.org/10.1145/3399715.3399918
- STATECHART (BuildingBlock)


?
onclick
?
PRE
POST

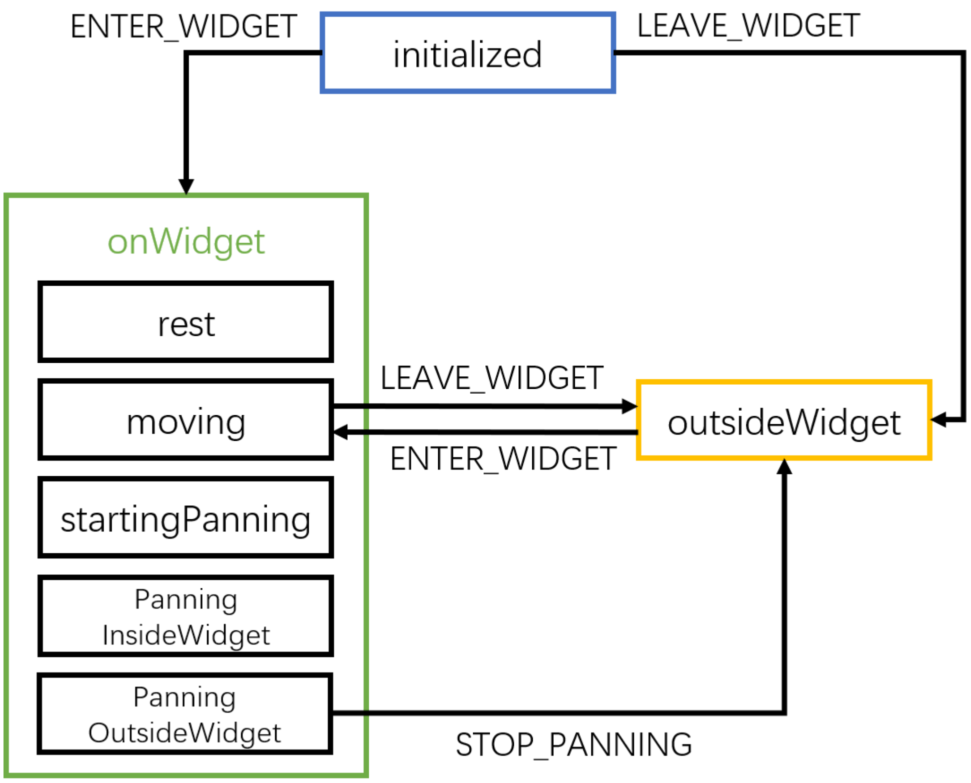

regionSelected
?
rightClick
?
!regionSelected
PRE
COND
POST

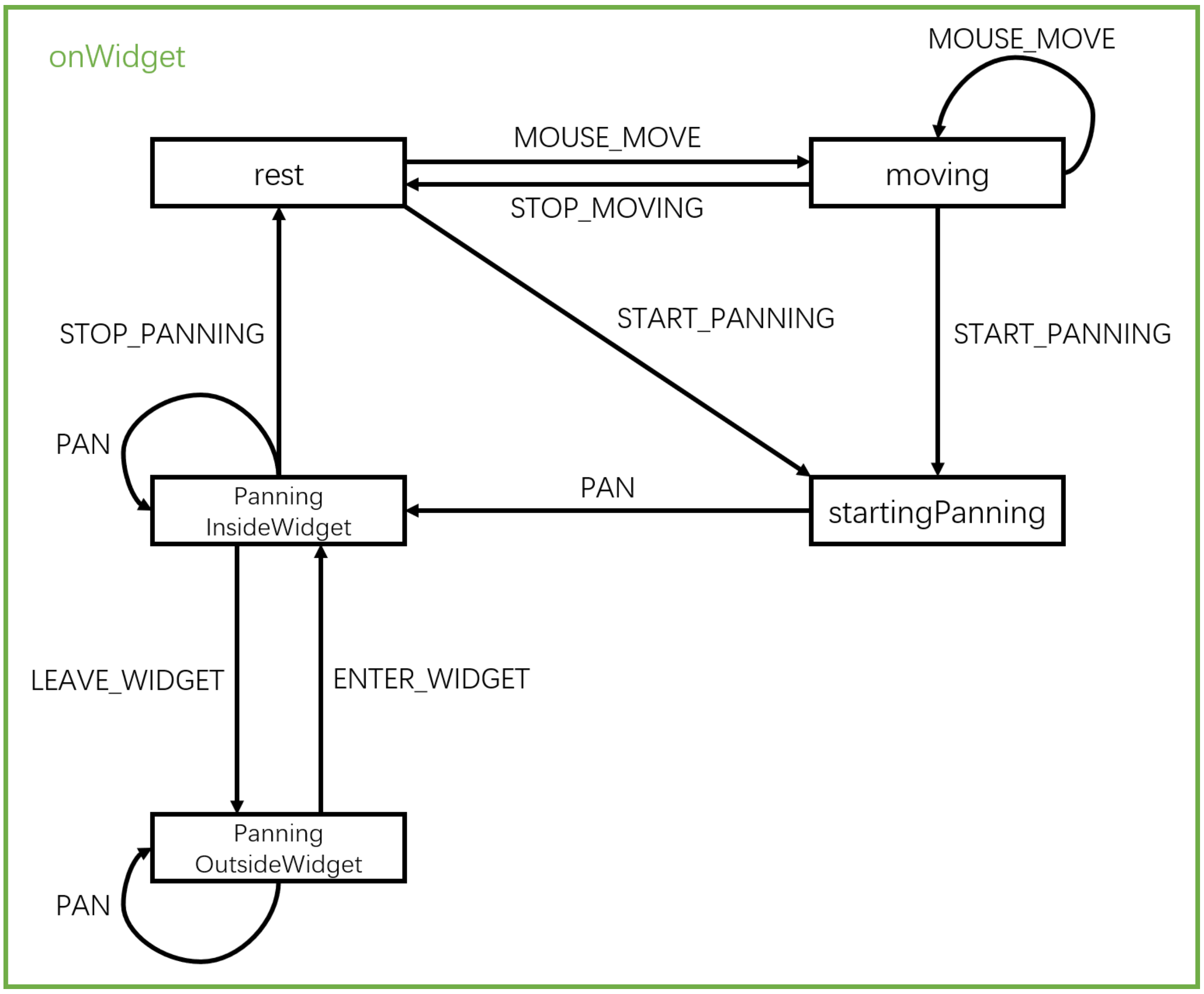
---

## PANNING


analytics
animate
data
display
flex
physics
query
scale
util
vis
flare

- Interaction class:
  - **Navigate.**
- Usage: Users click on some point in the visualization, and drag to move its elements.
- Events generation rate: Really low.
  - Dynamic selection queries when during the dragging new elements appear, only in cases like google maps?.
- Exemplar implementation:
  - Zoom and panning d3 implementation.
  - Zoomable scatter plot, with panning, in d3.
- Possible parameters for predictions:
  - Mouse position.
  - Mouse hovering.
  - Drag direction.
  - Drag speed?
- References
  - "Dynamic Prefetching of Data Tiles for Interactive Visualization" and many map-based applications.
- Statechart





---

## GUI ELEMENTS

---

### CHECKBOX/RADIO BUTTON/TOGGLE BUTTON

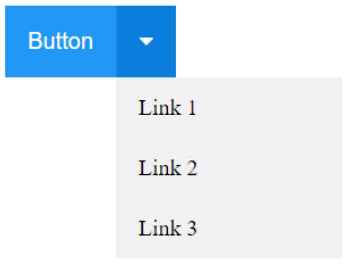
true, false, true

- Interaction class: **Filter**.
- Usage: The user clicks on the box, or on the circle, to apply a filter to some kind of data.
- Events generation rate: Behind it there is just one, generally very simple, query that relates directly to filtering for the specific value that was clicked.
- Exemplar implementation:
    - d3
    - html

- Possible parameters for predictions:
  - Distance from the mouse's position.
  - Time since the last click.
  - The user recently clicked on any other buttons in the widget (boolean).
- References
  - WIP

---

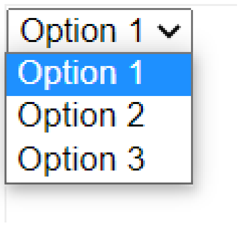
## SPLIT BUTTON



- Interaction class:
  - **Select**.
  - **Each button can have a different option -> refer to button.**
- Usage: It combines a button (typically invoking some default action) and a drop-down list with related, secondary actions. The user can press the button to invoke the action, or select another action from the drop-down list to be performed while pressing the button.
- Events generation rate: It depends on what the actions are doing on the DB.
- Exemplar implementation:
  - html
- Possible parameters for predictions:
  - Distance between the mouse and the button, the mouse and the drop-down button, the mouse and each option in the drop-down menu.
  - Time since the last click.
  - The user recently clicked on any other buttons in the widget (boolean).
- References
  - WIP

---

## LIST BOX



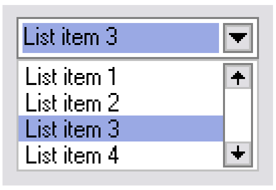
- Interaction class: **Select**.
- Usage: The user can select one or more items from a list contained within a **static**, multiple line text box. The user clicks inside the box on an item to select it, sometimes in

combination with the ⇧ Shift or Ctrl in order to make multiple selections. "Control-clicking" an item that has already been selected, unselects it.

- Events generation rate: One query, related to a selection for the specific value that is clicked.
- Exemplar implementation:
  - d3
  - implementation in d3 that is used to select one between different visualizations.
- Possible parameters for predictions:
  - Distance between the mouse and the drop-down button, the mouse and each option in the list.
  - Time since the last click.
  - The user recently clicked on any other buttons in the widget (boolean).
- References
  - WIP

---

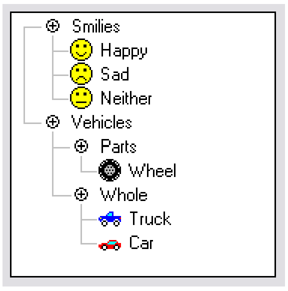
## DROP-DOWN MENU/COMBO BOX



- Interaction class: **Select**.
- Usage: The user can select one or more items from a list contained within a **dynamic**, multiple line text box. The user clicks inside the box on an item to select it, sometimes in combination with the ⇧ Shift or Ctrl in order to make multiple selections. "Control-clicking" an item that has already been selected, unselects it. The user can scroll up and down, while inside the menu, to show different options. In a combo box it is possible, for the user, to write some value in the text box, to search for it between the various options.
- Events generation rate: One query, related to a selection for the specific value that is clicked.
- Exemplar implementation:
  - d3
- Possible parameters for predictions:
  - Distance between the mouse and the drop-down button, the mouse and each option **currently visible** in the drop-down menu.
  - Time since the last click.
  - The user recently clicked on any other buttons in the widget (boolean).
- References
  - WIP

---

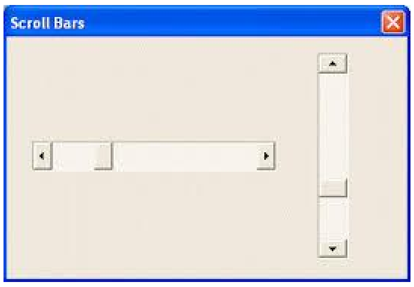
## TREE VIEW/DIAGRAM



- Interaction class:
  - **Navigate**.
  - **Select** (maybe, on the endpoints?)
  - **Aggregate?**
- Usage: The user can select one or more items from a list structured in a collapsing tree view. Clicking on one item will show all its children, or, if they are already visible, hide them.
- Events generation rate: WIP maybe one selection query when clicking on an endpoint?
- Exemplar implementation:
  - An example of an interactive d3 tree diagram.
- Possible parameters for predictions:
  - Distance between the mouse and each entry.
  - Time since the last click.
  - The user recently clicked on any other buttons in the widget (boolean).
  - Visible elements.
- References
  - WIP

---

## SCROLLBAR

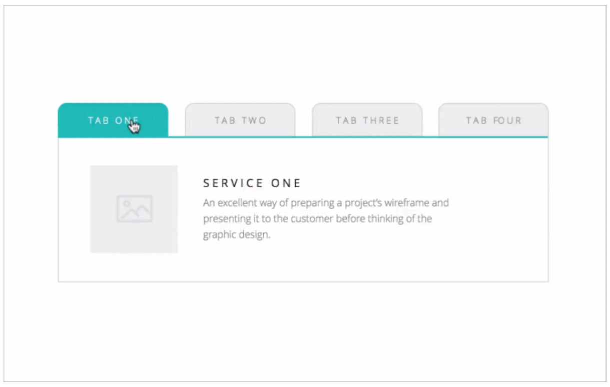


- Interaction class: **It depends on how the widget is used:**
  1. **Select** if it is used to visualize a portion of a DB, like in the example below.
  2. **Navigate** if it used to scroll through a page (classical use).
- Usage: The user can click on the bar and move it up/down or left/right to perform one of the actions above. The same operation can be done with the mouse wheel, generally only to scroll up/down.
- Events generation rate: One query, related to a selection, in case 1.
- Exemplar implementation:

    - An example of an interactive d3 visualization, that can be manipulated using a scrollbar.
- Possible parameters for predictions:
    - Distance between the mouse and the bar/s, or the up/down-left/right buttons.
    - Time since the last click.
    - The user recently clicked on any other buttons in the widget (boolean).
    - Pixel positions logs to predict drag trajectory.
- References
    - WIP

---

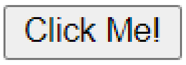
## TAB


TAB ONE
TAB TWO
TAB THREE
TAB FOUR
SERVICE ONE
An excellent way of preparing a project's wireframe and presenting it to the customer before thinking of the graphic design.

- Interaction class:
    - **Navigate**
    - **Select**
    - **Change?**
    - **Aggregate?**
- Usage: The user can click on each tab to perform a different action, like changing the visualization.
- Events generation rate: One selection query?
- Exemplar implementation:
    - html implementation
- Possible parameters for predictions:
    - Distance between the mouse and each tab.
    - Time since the last click.
    - The user recently clicked on other buttons in the widget.
- References
    - WIP

---

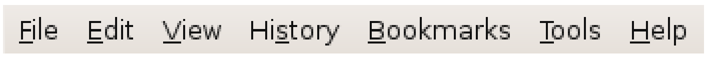
## BUTTON



- Interaction class: **It depends on the action performed after pressing it.**
- Usage: The user clicks on the button.
- Events generation rate: It depends on the action performed after pressing the button.
- Exemplar implementation:
  - Html button
- Possible parameters for predictions:
  - Distance between the mouse and the button.
  - Time since the last click.
- References
  - WIP

---

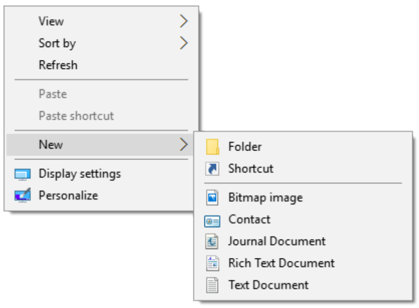
## MENU BAR



- Interaction class: **Introduce?**
- Usage: The user clicks on one of the buttons in the bar.
- Events generation rate: none.
- Exemplar implementation:
  - x
- Possible parameters for predictions:
  - Distance between the mouse and the bar?
  - Distance between the mouse and each element of the menu.
  - The user recently clicked on something in the menu?
  - Others...
- References
  - x

---

## CONTEXT MENU(POP-UP MENU)

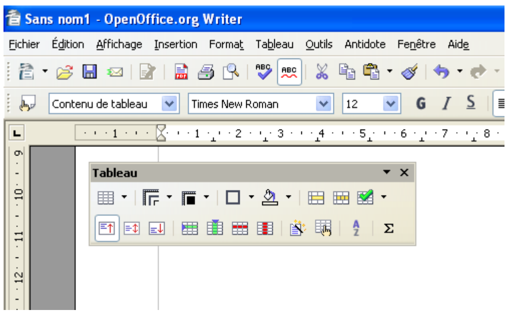


- Interaction class: **Introduce?**

- Usage: The user can select options in the menu by clicking on them. Each option can have a different task behind, or can open a new context menu.
- Events generation rate: none.
- Exemplar implementation:
  - WIP
- Possible parameters for predictions:
  - Distance between the mouse and the element that fires the pop-up.
  - Distance between the mouse and each element of the menu.
  - If the menu is visible or not?
  - Time since the menu is visible?
  - The user recently clicked on something in the menu?
  - Others...
- References
  - “However, a menu interface for changing from a scatter plot to a parallel coordinates view is an interaction technique since it allows users to manipulate a representation even though it may be less interactive or direct.”[1]

---

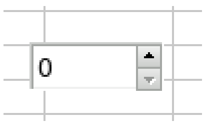
## TOOLBAR


Sans nom1 - OpenOffice.org Writer
Fichier Édition Affichage Insertion Format Tableau Outils Antidote Fenêtre Aide
Contenu de tableau
Times New Roman
12
Tableau

- Interaction class: **Introduce?**
- Usage: Like a menu but it can be manually shown/hid and dragged.
- Events generation rate: none.
- Exemplar implementation:
  - WIP
- Possible parameters for predictions:
  - Like the pop-up menu.
  - Visible or not.
- References
  - WIP

---

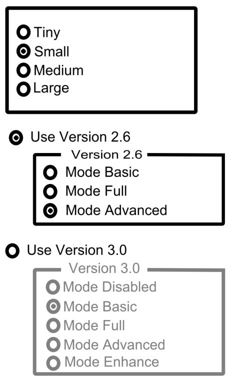
## SPINBUTTON



- Interaction class:
  - **Select** (if it is related to selecting some value to visualize)
  - **Change** (if it is changing some parameters of the visualization)
- Usage: Widget that allows a user to change the numeric value in the cell by interacting with a widget. The value of spin buttons can be changed either by typing a new number into the text box or by clicking on the arrow buttons. If the buttons are clicked and held, the numeric value will spin, incrementing or decrementing depending on the arrow being held.
- Events generation rate: One selection query if is SELECT, none otherwise. **IT CAN BE PERFORMED AGAIN AND AGAIN IN A SHORT WINDOW OF TIME.**
- Exemplar implementation:
  - Html implementation.
- Possible parameters for predictions:
  - Decreasing/increasing?
  - Distance.
  - Time since last click (for each button).
  - The user has selected the text box.
- References
  - WIP

---

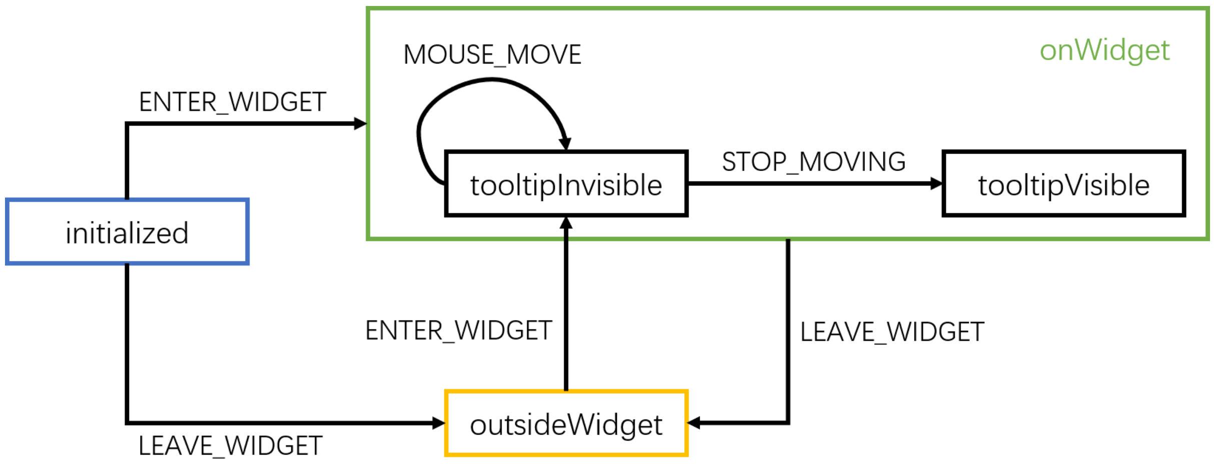
## CONTAINER WIDGETS



- Interaction class: **Arrange?**
- Usage: This is not really used by the user, as it just contains other widgets, displaying them in an organized way.
- Events generation rate: none.
- Exemplar implementation:
  - DIV in html.
  - Others.
- Possible parameters for predictions:
  - Distances.
  - Mouse inside the container?
- References
  - WIP

## TOOLTIP

- Statechart



---

# INTERESTING INFO

---

## QUOTES

- “The importance of interaction and the need for its further study seem undisputed. For example, the recent book Illuminating the Path: The Research and Development Agenda for Visual Analytics calls for further research on interaction: “Recommendation 3.3: Create a new science of interaction to support visual analytics. The grand challenge of interaction is to develop a taxonomy to describe the design space of interaction techniques that supports the science of analytic reasoning. We must characterize this design space and identify under-explored areas that are relevant to visual analytics. Then, R&D should be focused on expanding the repertoire of interaction techniques that can fill those gaps in the design space.” [1]

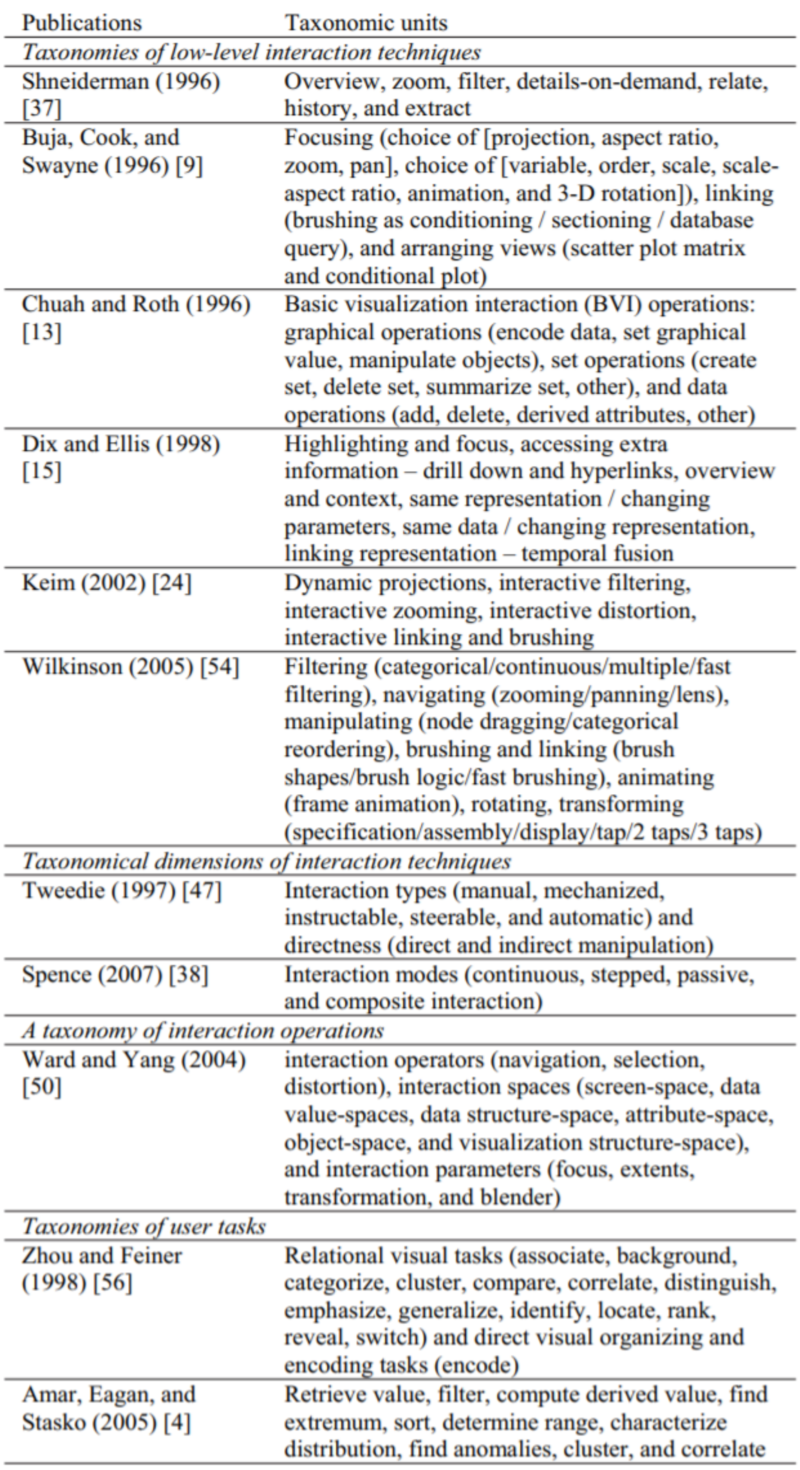
- “Finally, measuring the effectiveness of a taxonomy is difficult itself. We are drawn to a discussion of this issue by BeaudouinLafon [5] who proposes three dimensions to evaluate interaction models: 1) descriptive power, “the ability to describe a significant range of existing interface”; 2) evaluative power: “the ability to help assess multiple design alternatives”; and 3) generative power: “the ability to help designers create new designs” (p. 17). None of the taxonomies listed above appear to provide all three levels.”[1]
- “Through the categorization process, we realized that the categories are not collectively exhaustive. Some techniques are difficult to classify and do not quite fit into any one of the categories.[...] Some other interaction techniques appear to fulfill multiple user intents, which make it possible to classify them into several categories.“ [1]

Taken from:

1. J. S. Yi, Y. a. Kang, J. Stasko and J. A. Jacko, "Toward a Deeper Understanding of the Role of Interaction in Information Visualization," in IEEE Transactions on Visualization and Computer Graphics, vol. 13, no. 6, pp. 1224-1231, Nov.-Dec. 2007, doi: 10.1109/TVCG.2007.70515.

## EXISTING TAXONOMIES

Table 1. Infovis Taxonomies Relevant to Interaction Techniques



| Publications | Taxonomic units |
|---|---|
| *Taxonomies of low-level interaction techniques* | |
| Shneiderman (1996) [37] | Overview, zoom, filter, details-on-demand, relate, history, and extract |
| Buja, Cook, and Swayne (1996) [9] | Focusing (choice of [projection, aspect ratio, zoom, pan], choice of [variable, order, scale, scale-aspect ratio, animation, and 3-D rotation]), linking (brushing as conditioning / sectioning / database query), and arranging views (scatter plot matrix and conditional plot) |
| Chuah and Roth (1996) [13] | Basic visualization interaction (BVI) operations: graphical operations (encode data, set graphical value, manipulate objects), set operations (create set, delete set, summarize set, other), and data operations (add, delete, derived attributes, other) |
| Dix and Ellis (1998) [15] | Highlighting and focus, accessing extra information – drill down and hyperlinks, overview and context, same representation / changing parameters, same data / changing representation, linking representation – temporal fusion |
| Keim (2002) [24] | Dynamic projections, interactive filtering, interactive zooming, interactive distortion, interactive linking and brushing |
| Wilkinson (2005) [54] | Filtering (categorical/continuous/multiple/fast filtering), navigating (zooming/panning/lens), manipulating (node dragging/categorical reordering), brushing and linking (brush shapes/brush logic/fast brushing), animating (frame animation), rotating, transforming (specification/assembly/display/tap/2 taps/3 taps) |
| *Taxonomical dimensions of interaction techniques* | |
| Tweedie (1997) [47] | Interaction types (manual, mechanized, instructable, steerable, and automatic) and directness (direct and indirect manipulation) |
| Spence (2007) [38] | Interaction modes (continuous, stepped, passive, and composite interaction) |
| *A taxonomy of interaction operations* | |
| Ward and Yang (2004) [50] | interaction operators (navigation, selection, distortion), interaction spaces (screen-space, data value-spaces, data structure-space, attribute-space, object-space, and visualization structure-space), and interaction parameters (focus, extents, transformation, and blender) |
| *Taxonomies of user tasks* | |
| Zhou and Feiner (1998) [56] | Relational visual tasks (associate, background, categorize, cluster, compare, correlate, distinguish, emphasize, generalize, identify, locate, rank, reveal, switch) and direct visual organizing and encoding tasks (encode) |
| Amar, Eagan, and Stasko (2005) [4] | Retrieve value, filter, compute derived value, find extremum, sort, determine range, characterize distribution, find anomalies, cluster, and correlate |

(Taken from [1])