Visualization Schemas for Flexible Information Visualization

Chris North, Nathan Conklin, Varun Saini
Virginia Tech.
Proceedings of IEEE Symposium on InfoVis’02

Presented by: Hamid Haidarian Shahri

Apr. 20, 2006
Outline

- Relational Data Schema
- Motivation
- Related Work
- Snap-Together
- DataCompass, Snap Server
- Summary
Relational Data Schema

- Structural description of datasets
- Entities: attributes, tuples and relations
Motivation

- Relational data schema enables flexible database design
- No corresponding flexible ways to construct effective UI and visualization
  - visualization is based on data schema
  - database keeps changing
  - different views for same data
## Mismatch in Design Capabilities

<table>
<thead>
<tr>
<th>Design Goal</th>
<th>Relational Databases</th>
<th>Traditional Visualization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data design</td>
<td>Data design</td>
<td>Visualization design</td>
</tr>
<tr>
<td>Design Method</td>
<td>Data schema</td>
<td>Program code</td>
</tr>
<tr>
<td>Data owner</td>
<td>Data owner</td>
<td>Programmer only</td>
</tr>
<tr>
<td>Designer</td>
<td>Data owner</td>
<td>Programmer only</td>
</tr>
<tr>
<td>Design Change</td>
<td>Rapid, dynamic</td>
<td>Slow, static</td>
</tr>
<tr>
<td>Adaptability</td>
<td>Flexible</td>
<td>Brittle</td>
</tr>
</tbody>
</table>
Related Work

- Single relation visualization
  - Spotfire
  - APT
  - Sage/SageBrush
  - DEVise

- Multiple relation visualization
  - Visage
  - DataSplash/Tioga-2
  - Rivet/Polaris
  - Sieve
Visage

www-2.cs.cmu.edu/~sage/visage.html
DataSplash/Tioga-2

http://datasplash.cs.berkeley.edu/tour_quick.html
Polaris

http://graphics.stanford.edu/projects/polaris/
Now we explain a Web Browser example!!
Snap-Together User Interface

- Visualization Schemas
  - represented as a graph
  - support direct manipulation
  - similar to relational data schema
Snap-Together User Interface

**Nodes**
- Represent instantiated visualization components
- Each component has a corresponding relation (URLs, HitCounts, Referrers)
Snap-Together User Interface

- **Edges**
  - Represent coordinations between visualizations
  - Join relation (1-1, 1-M)
  - Join attribute
  - Action for coordination (select, load)
A strong analogy between relational database concepts and Snap visualization concepts enables a matching level of design capability.
Snap-Together Theory

- Snap Visualization Model
  - Multiple views/components
  - Schema primitives (select, load)
  - Data-centric coordination and joins
### Snap-Together System Architecture

<table>
<thead>
<tr>
<th>Theory</th>
<th>UI</th>
<th>Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coordinated Multi-views</td>
<td><img src="#" alt="Visualization Model" /></td>
<td>Coordination Manager</td>
</tr>
<tr>
<td>Visualization Model</td>
<td><img src="#" alt="Visualization Model" /></td>
<td>Visualization Schema</td>
</tr>
<tr>
<td>- Visualization</td>
<td><img src="#" alt="Visualization Model" /></td>
<td>Coordination Graph</td>
</tr>
<tr>
<td>- Coordination</td>
<td><img src="#" alt="Visualization Model" /></td>
<td></td>
</tr>
<tr>
<td>Relational Model</td>
<td></td>
<td>Database Manager</td>
</tr>
<tr>
<td>- Relation</td>
<td><img src="#" alt="Visualization Model" /></td>
<td>Database Schema</td>
</tr>
<tr>
<td>- Association</td>
<td><img src="#" alt="Visualization Model" /></td>
<td></td>
</tr>
<tr>
<td>Data Source</td>
<td><img src="#" alt="Visualization Model" /></td>
<td>Relational Database</td>
</tr>
</tbody>
</table>

- Colored squares represent entities or components within the system architecture.
- Arrows indicate relationships and dependencies between these components.
- The Coordinated Multi-views, Visualization Model, Relational Model, and Data Source are listed in the theory section.
- The UI section contains visual representations of the models.
- The Architecture section includes components like Coordination Manager, Visualization Schema, Coordination Graph, Database Manager, and Database Schema.
Snap-Together Demo 2

- Other applications

Video
For novice users or very complex database schemas

- Step-by-step construction
  - Yellow: relations already displayed by visualization
  - Red: from the data schema

- Interchangeable with visualization schema

- Bottom-up approach (vs. Top down approach in V. schemas)
Snap Visualization Server

- Event-based coordination
  - Send & receive events
  - Translate events on selection/navigation

- Extensible architecture (component implementation language)
Summary: Snap’s Three Perspectives

- **Theory**: multi-view visualization, coordinating between data design and visualization design

- **UI**: diagrammatic UI to enable rapid customization of visualization without programming

- **System Architecture**: web-based component architecture to support run-time integration of diverse data sources and visualization tools, and dissemination of custom visualizations as web pages
Discussion

- Strong Points

- Potential Problems
Remarks

- **Merits:**
  - Visualization schema notion: flexible and easy user interface, no programming
  - DataCompass to guide users
  - Extensible architecture for coordinating visualization components (snap server)

- **Shortcoming:**
  - No standards for the development of visualization components, i.e. API’s or hooks in the component
  - Limited support for coordinated data navigation, various events (pan, zoom, …)
Thanks!