Snap-Together Visualization: Evaluating Coordination Usage and Construction

Chris North and **Ben Shneiderman**
Human-Computer Interaction Lab &
Department of Computer Science
University of Maryland, College Park, MD 20742 USA
north@cs.umd.edu, ben@cs.umd.edu
http://www.cs.umd.edu/hcil

ABSTRACT

Multiple coordinated visualizations enable users to rapidly explore complex information. However, users often need unforeseen combinations of coordinated visualizations. Snap-Together Visualization is a conceptual model, based on the relational model, and system to enable users to quickly coordinate otherwise-independent visualization tools. Users construct customized browsing environments with coordinations for selecting, navigating, and loading data, without programming.

Evaluation revealed benefits, cognitive issues, and usability concerns with coordination concepts and the Snap system. Two user studies explore the value of coordination usage and the learnability of coordination construction. The overview and detail-view coordination improved user performance by 30-80%, depending on task. Data savvy users were very capable and thrilled to rapidly construct powerful coordinated visualizations.

Keywords

User interface, information visualization, multiple views, coordination, user study and usability.

INTRODUCTION

In exploring information, two or more coordinated visualizations are often required to adequately display and browse the data. For example, Microsoft's Windows Explorer employs three visualizations to browse hierarchical file systems: an outliner view of the folders, a tabular view of the files in the selected folder, and a web view of the details of the selected file including a miniature quick-view. In Spotfire [AW95], a commercial scatterplot visualization tool, selecting a record in the plot displays its attribute values in a web browser.

While these combinations of coordinated views are very helpful for some tasks, what about other combinations. What if, in Windows Explorer, users want to view their folders as a scatterplot instead of an outliner? Then they could quickly spot large old folders, and select them to see contents in the tabular view. If browsing a census database, why can't users click on a state in a Spotfire visualization to display its counties in, say, a Treemap [Shn92] visualization? (See Figures 1,2,3 for examples.)

These alternate combinations typically require custom development. In our lab, researchers stumble over this problem often, and must constantly re-implement coordinations between new unforeseen combinations of views. Unfortunately, this is a poor solution to the problem. Even with good component-based design, these hard-coded combinations are inflexible and difficult to construct.

A lightweight mechanism is needed to allow end-users to easily "snap" individual visualizations together into custom combinations. These combinations can exploit simple relationships in the data to support browsing. This must not be a toolkit that requires programming, but a user interface.

Specifically, users should be able to choose and coordinate visualizations so that: selecting or navigating to a data item in one view causes another view to select or navigate to that item or load and display data related to that item. The "load" capability is particularly potent. For example, users can browse through multiple hierarchical levels in a database using different visualizations tuned for each level, as in the states and counties example above.

Snap-Together Visualization (Snap) is a conceptual model and implemented system developed to meet these needs. To determine if Snap accomplishes this goal, it is critical to evaluate the system to determine:

1. *Value of coordination usage*: What is the payoff of the resulting "snapped" combinations when browsing?
2. *Learnability of coordination construction*: Can users learn coordination concepts and snap visualizations together to construct appropriate combinations?

This paper presents an overview of the Snap-Together Visualization approach and then reports on studies of its value and learnability.

HCIL Technical Report No. 99-26 (October 1999)

http://www.cs.umd.edu/hcil

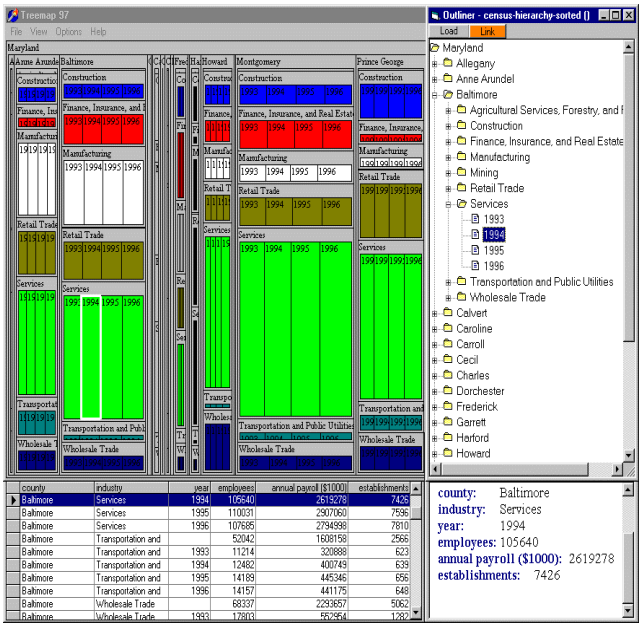


Figure 1: A snapped user interface for browsing census data, using a Treemap, outliner, table, and text view.

Related Work

As information visualization advances, systems for multiple coordinated visualizations are becoming more common. These systems allow users to explore data in a flexible way, providing flexibility in loading different data sets, flexibility in displaying many different visualizations of the data, and some provide flexibility in the coordinations between the displayed visualizations. Very little work has been done to evaluate coordination approaches.

Three impressive systems that are most related to Snap are Visage [RLS96], DEVise [LRB97], and LinkWinds [JBO94]. Each has a unique and interesting focus.

With Visage's information-centric approach, users explore data by dragging-and-dropping data items between many visualizations. Brushing highlights items across all views, and is generalized to multiple tables. Its VQE component synchronizes dynamic query filtering between views. Its SAGE component generates custom visualizations.

DEVise takes an attribute-based approach that enables coordinations that are more complex. Users can synchronize panning and zooming of many plots with common axes. It also supports the notion of piping data between views, so that the data in two views can be combined into a 3rd.

LinkWinds combines the attribute-based and data-piping approaches. Users connect a variety of visualizations of scientific sampled data as a series of filters. Then users can select attribute ranges in one view to filter contents of views downstream in the pipeline.

Also, there are many data plot brushing [BC87] systems, both research and commercial (e.g. Datadesk, SAS). Most of these systems focus on the single table model.

Snap focuses on (a) interconnecting the visualization tools created by other researchers and developers in the field to (b) construct coordinated browsers for rapid navigation of relationships (as the Windows Explorer example illustrates). It expands on Visage's approach in that individual data items are the focus of interaction, and on DEVise's and LinkWinds' approach of enabling users to link views. Snap's simple model finds a middle ground between these that enables more than just brushing, but avoids the complexities of the attribute-based approach.

Multiple coordinated visualization approaches have become an important and diverse topic. For a comprehensive review of many systems, as well as implementation details of Snap, see [Nor00].

SNAP TOGETHER VISUALIZATION

The Snap-Together Visualization model is based on the relational database model:

Relational Concept		Snap Concept
Relation	=	Visualization
Tuple	=	Item in a visualization
Primary key	=	Item ID
Actions:		
Select tuple	=	Select or navigate to item
Query	=	Load
Join	=	Coordination

Snap's model of a visualization is: a visual depiction of a relation in which individual tuples are selectable or actionable. These actions are simple unary operations on an item, and are classified in 2 categories: select actions (e.g. highlight or outline the item; invoked by clicking on or mousing over the item), and navigate-to actions (e.g. scroll or zoom to the item). Each visualization publishes its own supported actions to Snap.

The unary load action on a visualization queries and displays tuples that are related to a given tuple. This is accomplished with queries that take the tuple's primary key as a parameter. For example:

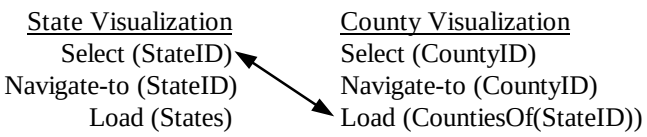
Load CountiesOf(<Maryland>) =

SELECT * FROM Counties
WHERE StateID = <Maryland>.

A Snap coordination tightly couples actions between two visualizations by using the join relationship between their relations. For example, selecting or navigating to an item in one view could cause another view to select or navigate to that item (one to one) or load and display data items related to that item (one to many). Hence, a graph of coordinations between visualizations corresponds to the graph of joins between the relations in the database schema diagram. This was inspired in part by RMM [ISB95], a system for constructing web site navigational

structure from underlying relational databases. In RMM, database relationships correspond to hyperlinks, whereas, in Snap they correspond to coordinations.

The Snap user interface employs a 2-step process. Users first load tables or queries into visualizations, then they snap the visualizations together to establish coordinations. To snap, users specify which action and item pairs to tightly couple between visualizations. For example:



In addition, coordinations that are more complex can be accomplished by inserting an arbitrarily complex mapping relation between the items acted on between views.

A critical design goal was to minimize impact on visualization tool implementation. Snap uses a simple model and communication protocol that independent tools built by other developers can hook into easily. Existing tools can easily be made "snap-able", since they can maintain their own data structures and only need to support actions they already support. For example, Spotfire, a commercial software package, was integrated using its API and a 10-line VB wrapper to translate communication. We propose this protocol as a standard, like cut-and-paste APIs in window systems that is trivially added to a visualization tool to make it immediately snap-able with other tools.

Common Coordinations in Snap

Some common coordinations and their Snap specification:

- *Brushing*: (e.g. Figure 1, Treemap to outliner)

Coupled actions: select to select
Data relationship: one to one (e.g. same table)
Usage: Selecting an item in one view highlights the corresponding item in another view. Typically used to identify like items when a set of items is displayed in different views for different contexts. In the many-to-many case, can be used to explore data interrelationships.

- *Overview and Detail View*: (e.g. Figure 2)

Coupled actions: select to scroll
Data relationship: one to one
Usage: Selecting an item in the overview scrolls (or more generally navigates) the detail view to that item. The item is represented visually smaller in the overview than in the detail view. Allows direct access to details, and provides context for details.

- *Multi-Level Browsing*: (e.g. Figure 3, list to table)

Coupled actions: select to load
Data relationship: one to many (e.g. states to counties)
Usage: Selecting an item in one view loads related items into another view. Typically used for browsing across levels of scale, across one-to-many joins, or from aggregate to its contents.

STUDY 1: VALUE OF COORDINATION USAGE

The goal of this study is to measure the added value of coordinated views over independent or single views in terms of user task times and subjective satisfaction for browsing large information spaces. The visual feedback across views could be distracting or disorienting for users. However, if there is a benefit, what is its magnitude?

While there are many possibilities, this study examines the Overview and Detail View coordination. This coordination has two enhancements over the traditional single-view detail-only display:

1. *Overview*. A display enhancement that depicts the full breadth of the data in a compact form, as in a table of contents.
2. *Coordination*. An interaction enhancement that allows users to select an item in the overview to scroll the detail view to that item. Likewise, directly scrolling the detail view highlights the current item in the overview.

Chimera and Shneiderman's [CS94] result seems to indicate that the overview and detail view should perform better than detail-only. However, if so, what is the important factor that causes improved performance? Is it (a) the information displayed in the overview, or (b) the coordination between the overview and detail view?

Independent Variables

User interface. A simple textual interface, constructed with Snap-Together Visualization, uses the overview and detail-view coordination for browsing population statistics of the U.S. states from the Census Bureau's 1990 census. Three treatments: (see Figure 2)

1. *Detail-Only*: A single scrolling textual view of the states in alphabetical order and their data.
2. *No-Coordination*: The same view as Detail-Only, with the addition of an overview tiled on the left. The overview displays an alphabetical list of the names of the states. The views are not coordinated.
3. *Coordination*: The same views as No-Coordination, with the addition of coordination between them. In Snap, this links overview:select(stateID) to detailview:scroll(stateID).

Task. A variety of browsing tasks, using a question and answer approach. Nine treatments: (listed easy to difficult)

1. *Coverage-Yes*: "Does the information include statistics about the state of Ohio?" where Ohio is included in the data.
2. *Coverage-No*: Same as Coverage-Yes, but where the state is not included in the data.
3. *Overview patterns*: "How many states in the list begin with the letter M?"
4. *Visual lookup*: "What is the population of the 6th state from the bottom of the list?"
5. *Nominal lookup*: "What is the population of Georgia?"
6. *Compare-2*: "Which of the following states has higher Median Family Income: California or Washington?"

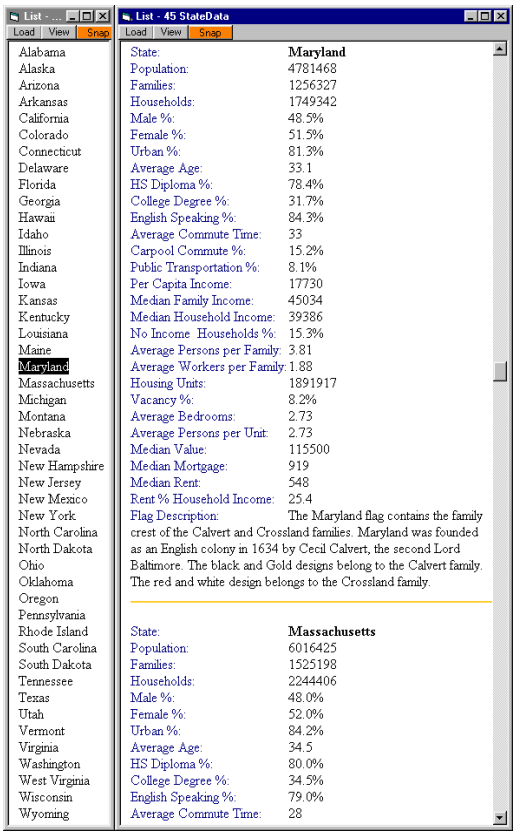


Figure 2: The user interface for study 1, using a pair of textual views with overview and detail-view coordination.

7. *Compare-5*: "Which of the following 5 states has higher Median Household Income: Florida, Texas, Louisiana, Alaska, or Oregon?"
8. *Search* for target value: "Which state has Average Commute Time of 31?"
9. Full *scan*: "Which state has the highest College Degree %?"

Dependent Variables:

User performance time: Time to correctly complete each task, not including reading the task question.

User subjective satisfaction: Subjects rated their satisfaction with each interface on a scale of 1 to 9 on four categories (with scales): comprehensibility (confusing to clear), ease of use (difficult to easy), speed of use (slow to fast), overall satisfaction (terrible to wonderful).

Procedure

The 18 subjects were students and staff from campus, and were paid $10 to participate. A within-subjects design was used. Each subject used all three interfaces to perform all nine tasks. To avoid repetition, three different but similar sets of task questions were used. To counterbalance for potential orders effects, all 6 possible permutations of interface order were each assigned 3 times. The three task sets were not permuted.

For each interface, subjects were first trained in its use and performed several practice tasks before beginning the timed trials. After finishing all three interface treatments, subjects then completed the subjective satisfaction questionnaire.

Results

Analysis of the data reveals a strong and interesting result. Chart 1 shows the mean user performance times for each task and interface. A 3x9 ANOVA reveals that the user interface effect, task effect, and interaction effect are all statistically significant at $p<.001$. Nine one-way ANOVAs reveal that interface is significant for all nine tasks at $p<.001$.

Finally, individual t-tests between each pair of interfaces within each task determine performance advantages. For tasks 1, 2, and 3, Coordination and No-Coordination are both significantly faster than Detail-Only at $p<.001$, but not proven different from each other. Whereas, in tasks 5 through nine, Coordination is significantly faster than both No-Coordination and Detail-Only at $p<.001$, and the latter are not proven different from each other. However, while task 4 (Visual lookup) could be included in the second group of tasks, it may classify as an in-between case. For this task, Coordination is significantly faster than the other two interfaces at $p<.005$, but No-Coordination is marginally significant over Detail-Only at the $p<.07$ level.

First, Coordination results in major improvement in task performance time over Detail-Only for all tasks. On average, Coordination achieves an 80% speedup over Detail-Only for easy tasks and 50% for difficult tasks. The least improvement, about 33%, is in task 6 (compare-2). This task had the lowest interaction to thinking time ratio.

The No-Coordination interface is likely the source of the interaction effect between task and interface. It results in a nearly binary pattern. For tasks 1-3, No-Coordination performs faster than Detail-Only, and its averages are on par with Coordination. In these tasks, subjects only needed the information in the overview to accomplish the task. Whereas, in tasks 5-9 the Coordination interface is faster than No-Coordination, and the averages for No-Coordination are on par with Detail-Only. In these tasks, subjects needed to access the details of the data. Observing subjects' behavior as they performed these tasks revealed that when using No-Coordination they tended to ignore the overview. The lack of significant difference between No-Coordination and Detail-Only in these cases does not imply that they are necessarily the same. We conjecture that they are the same due to our observation of the users. In any case, what is important is that Coordination is significantly faster than No-Coordination in these cases. Hence, in tasks where access to details is important, undoubtedly a majority in common applications, coordination is an absolute key.

Task 4 (Visual lookup) might classify as an in-between case. With No-Coordination, many subjects determined

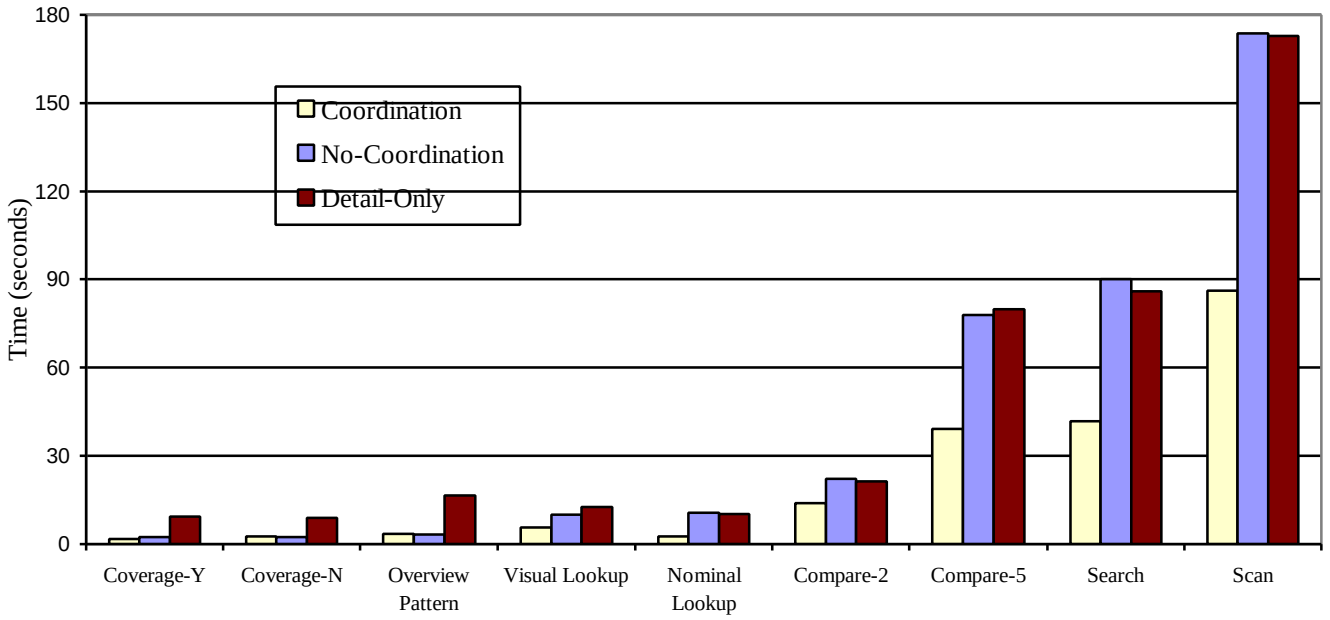


Chart 1: User study 1, average user performance time for tasks. The coordinated interface ha significantly faster performance in most cases.

the name of the target state from the overview, then scrolled to it in the detail view. With Detail-Only, they scrolled to the bottom, then scrolled back up while counting, and sometimes lost track. Apparently, this is a case where just having the contextual information of the overview was somewhat advantageous. Even so, Coordination was still a major improvement over both.

In fact, a key result is that Coordination performance times for lookup tasks (4 and 5) are in the same extremely fast range as overview tasks 1-3. Whereas, No-Coordination times drop to Detail-Only level performance. When looking up details, perhaps the most common task, Coordination especially excels.

Subjective Satisfaction

With the satisfaction data (Chart 2), a 3x4 within-subjects ANOVA indicates that user interface and satisfaction category are significant at $p<.001$, and interaction effect is significant at $p<.005$ level. One-way ANOVAs for each category indicate that Comprehensibility is significant at $p<.02$ level. Ease of use, Speed of use, and Overall Satisfaction are all significant at $p<.001$ level.

Analyzing each pair of interface treatments within each category reveals that all pairs are significant at $p<.005$ except: Detail-Only and No-Coordination in Ease of Use are significant at $p<.05$ and the same pair in Comprehensibility are not proven different.

Coordination is a clear winner, gaining nearly twice the rankings of Detail-Only and No-Coordination in Ease, Speed, and Overall. On average, subjects ranked No-Coordination 1-2 points higher than Detail-Only, except in Comprehensibility they ranked about the same. While completing the survey, subjects often stated that No-Coordination was only useful for the overview tasks.

Answers

Returning to the research questions: Which factor is more critical, the overview information or the coordination? The answer is nearly binary. If only the overview information is needed, then naturally coordination is not necessary. However, for the important cases where access to details is needed, [illegible] is the magnitude [illegible] difficult tasks, the [illegible] half.

When first presente[illegible] many subjects im[illegible] overview expecting [illegible] they had not yet se[illegible] not only are users [illegible] displays, but they a[illegible] visibly distraught w[illegible] they hoped. Even [illegible] presented with the [illegible] satisfaction data i[illegible] indicate that, as th[illegible] prefer environments [illegible] speed their tasks.

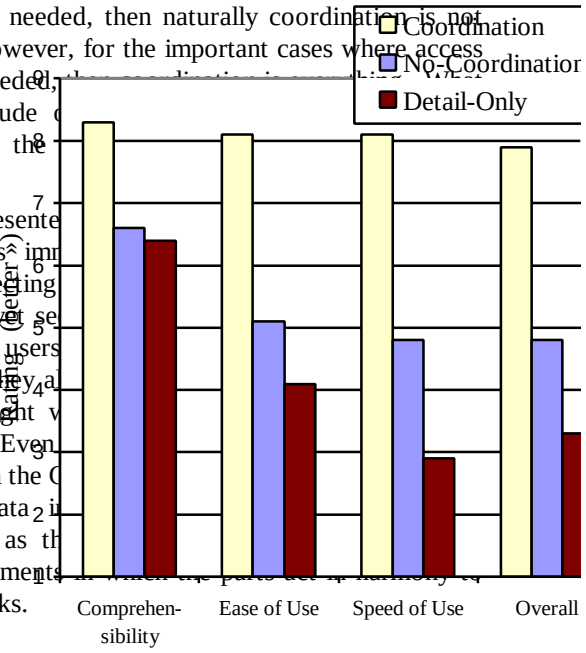


Chart 2: User study 1, average user subjective satisfaction. The coordinated interfaced rates significa higher in all four categories.

STUDY 2: LEARNABILITY OF COORDINATION CONSTRUCTION

The goal of the second study is to determine if users can learn to construct coordinated-view browsers and how difficult it is for users to construct them, in terms of success rate and time to completion, and to identify cognitive trouble-spots in the construction process. Can users grasp the concept of coordinating two independent visualizations together to form a unified browsing tool? We want to learn what cognitive issues are involved, how much training is required, how users' backgrounds effect performance, and whether relatively novice users will be able to construct powerful exploration tools in a short time. This study also reveals potential Snap user interface improvements.

The Snap-Together Visualization environment is used to examine these issues. Currently, Snap employs a 2-step approach to building coordinated-view browsers. First, users drop database tables or queries into visualizations. Second, users snap the visualizations together to coordinate actions between them. Snap currently uses Microsoft Access to enable users to edit queries.

Procedure

Because of the nature of the goals of this study, a more informal approach is taken. We worked with six subjects on a one-on-one basis. Four of the subjects were employees of the U.S. Bureau of the Census, three of which were data analysts or statisticians, and one a programmer. The other two subjects were computer science students on campus.

First, background information was obtained from each subject concerning their occupation and experience with: census data, computers, databases, Microsoft Access, visualization tools, and programming.

Then, each subject was trained on Snap-Together Visualization. The training program consisted of:

1. A quick demonstration of Snap by the administrator to give the subject an overview and motivation.
2. Review of various background concepts including:
    - Relational database concepts including: tables, records, fields, primary keys, foreign keys.
    - Database query concepts including: projection, selection, sort and join.
    - Snap-Together Visualization model concepts.
3. Detailed instruction on the use of Snap and Microsoft Access. The subjects walked through the construction of a few variations of coordinated interfaces for browsing census data. This demonstrated how to construct common types of coordinations.

Then, when confident to continue, each subject began the testing phase. Subjects were given a database of census data for the U.S. states and counties, and Snap (including a set of Visualization tools) and Microsoft Access. Testing consisted of three exercises in which subjects were asked to construct a coordinated user interface according to a provided specification:

Exercise 1: The first specification consisted of a printed screenshot of the desired user interface. It was identical to the Coordination interface in the first user study (Figure 2). This trial was designed to be fairly easy, similar to those constructed in the training, and to build confidence.

Exercise 2: The second specification was also a screenshot (Figure 3), but more difficult. It involved a one-to-many join relationship, so that selecting a state would display data for that state's counties.

Exercise 3: The final specification consisted of a textual description of the browsing task that the constructed interface should support: "Please create a user interface that will support users in efficiently performing the following task: To be able to quickly discover which states have high population and high Per Capita Income, and examine their counties with the most employees." This trial was designed to test if subjects could think abstractly about coordination, think task-oriented, think in terms of user-interface design, and to allow for creativity and variation.

Finally, subjects were given the opportunity to play, describe problems with the Snap user interface, and offer suggestions for improvement.

We measured:

- Subjects' background information.
- Learning time.
- Success (y/n or how close to success).
- Time to completion.

We also observed:

- Cognitive trouble spots (in training and test trials).
- Snap user interface problems.

Results

From the background survey, none of the subjects except the Census programmer had experience with Microsoft Access or SQL, and little exposure to relational database concepts. The Census analysts had significant experience with census data, but generally used flat files or spreadsheets. Each had experience with only basic visualization tools (e.g. Excel charts).

All the subjects completed the training phase in 30-45 minutes. They all were able to complete all three exercises, with occasional help in wading through Access's visual query editor. They accomplished exercise 1 in 2-5 minutes, and exercise 2 in 8-12 minutes. They spent 10-15 minutes on exercise 3 until they were satisfied with their solution.

As to the subjects' general reaction to Snap-Together Visualization, we were impressed by their level of excitement. The subjects were quick to learn the concepts

and usage, and were very capable to construct their own coordinated interfaces. Several stated that they had a gratifying sense of satisfaction and power in being able to both (a) so quickly snap powerful exploration environments together, and (b) with just a single click effect exploration across several visualizations and see the many parts operate as a whole. They reported that it made exploration seem effortless, especially in comparison to the standard tools they are used to.

There was an interesting difference between the reaction of the programmers' and data analysts. The programmers were excited about the component based programming approach, and the ability to rapidly construct new interfaces. Whereas, the data analysts were excited to be able to explore the data so thoroughly and efficiently. They did not see it as construction, but as exploration.

In fact, to our surprise, the data analysts performed better than the programmers did. They learned the database concepts quicker, completed the exercises quicker, and constructed creative interesting new interfaces. Perhaps they were more motivated by the use of examples involving Census data. Even during the training, they were already trying variations of snaps and exploring the data. Two pointed out various anomalies in the data. After finishing the exercises, these subjects each stayed for an additional hour to play. All four Census subjects expressed desire to use Snap-Together tools in their work. In fact, a collaborative effort has been undertaken.

An important result was the creativity and variation evident in the subjects' solutions to exercise three. Subjects were able to design user interfaces that made cognitive sense to their own perspective on the data. They used a mixture of visualizations including tables, scatterplots, and lists. For example, while the expected design was 2 scatterplots with a select-to-load (one-to-many) snap, one of the data analyst subjects augmented this design with a pair of lists for the state and county names. The subject stated that this would help to see which state and county was currently selected in the scatterplots, and also allow for accessing states by name which would be difficult with the scatterplot alone. Another subject who preferred to see the values placed the counties in a table sorted by number of employees. One had even constructed a browser (similar to Figure 1) using the Treemap visualization, which is generally considered a more advanced tool difficult for novices. In addition to variation in user interfaces, subjects made use of the transitive property of snaps to snap visualizations in different pairings.

Overall, subjects did not have problems grasping the cognitive concept of coordinating views. They were able to generate designs by duplication and by abstract task description. The problems they did have were simply in manipulating the Snap and Access user interfaces.

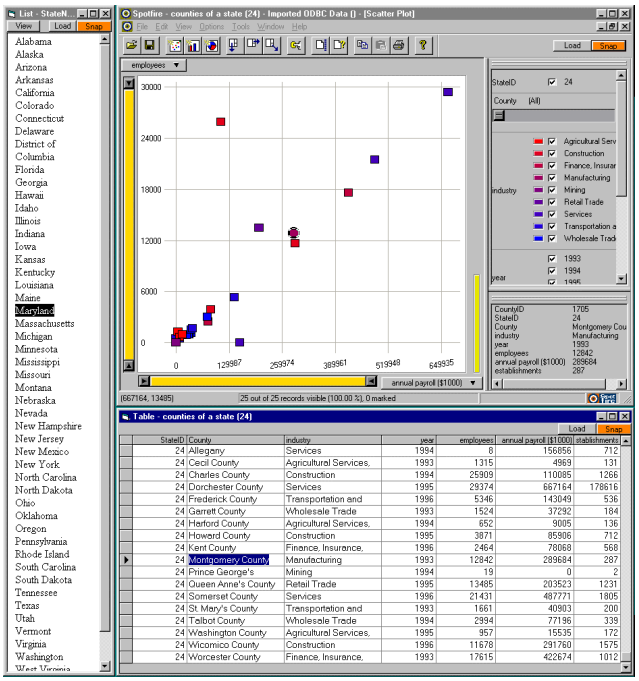


Figure 3: The user interface specification for study 2, exercise 2. It uses a text list, Spotfire scatterplot, and table view to browse census data for states and counties.

Querying was by far the most difficult part of the construction process for the subjects. Learning to use Access and its query editor is a challenge in such a short time.

User Interface Issues

Understanding the basic underlying model of Snap was critical. However, the current Snap user interface and the form fill-in style of the Snap Specification dialog do not reflect this model well. This study identified four major trouble spots in the interface:

1. The terminology of the snap-able actions "select" and "load" caused some confusion. It was not clear enough that these represented user interface actions. Apparently some subjects were confusing "select" with the database query sense of selection.
2. For simplicity, Snap uses the Access query editor. However, this made constructing a select-to-load (one-to-many) snap very laborious, and subjects sometimes got lost in the 3 step process: writing the parameterized query, opening the query in a visualization, and specifying the snap.
3. When constructing interfaces of three or more visualizations, subjects sometimes forgot which visualizations were snapped and how they were snapped. They had to recheck each pair.
4. When subjects weren't quite sure how to snap visualizations, they would often "just try stuff" and see how it behaves. A snap debugging mode is needed to help

them see how the tight-couplings propagate between the visualizations.

Redesigning the Snap interface around an overview diagram would solve these problems. A node and link diagram could represent the visualizations as nodes and snaps as links between them. This overview could become the primary user interface for constructing, editing, examining, and debugging snaps. Such a visual representation with direct-manipulation interaction would closely reflect the conceptual model of Snap. Hence, this would likely reduce users' training time as well. We are already working on this.

In addition, while the ability to specify queries with Access enables more complex scenarios, it is a burden for the common simple snaps. Snap could generate these simple queries automatically and, with the addition of a field selection interface, would obviate the need for Access in common cases.

Window management is also a serious problem. Subjects spent considerable amounts of time rearranging visualization windows on the screen into nicely tiled layouts. Others have proposed solutions to this general problem (see [KS97] for a review).

Combined Analysis

In conjunction with results from the first study, this data may indicate the breakpoint at which time savings during browsing surpass coordination construction time. In exercise 1, subjects constructed the same user interface as was used in the first study for browsing tasks. The time cost of constructing the coordinated interface was about 2-5 minutes, while it saved about 0.6-1.5 minutes over the standard Detail-Only interface for the more difficult tasks (where 5 or more items were accessed). Hence, after just a few tasks, users are already reaping savings with snapping their own interface. Of course, it is difficult to factor in learning time and effects of sharing snapped interfaces. Nevertheless, this simple analysis is revealing. Customized information visualization is within the grasp of novice users.

CONCLUSIONS and FUTURE WORK

Snap-Together Visualization enables end-users to construct their own customized coordinated visualization environments to explore their data. Two studies examined the usability of these concepts in terms of their value and learnability.

The overview and detail view coordination offered a 30-80% speedup for many user tasks. Data savvy users were enthusiastic as they learned coordination concepts and constructed coordinated interfaces of their own. The implications are powerful. These users are ready for and strongly desire significantly more advanced tools than standard detail-only, uncoordinated, or hard-wired systems. While these cognitive issues were examined within the Snap platform, we believe that these results will apply to similar coordinations in other systems.

This is only the tip of the iceberg. Design of each coordination is critical, and others need to be evaluated. Of particular interest are the multi-level browsing and brushing coordinations. In addition, these studies have already identified major improvements to the Snap user interface for construction based on cognitive issues. The new designs will need refinement and testing. These initial successes point to great potential in coordination, and continued research is needed to explore coordination overviews, coordination guidelines, and strategies for aggregation, history keeping, and more.

ACKNOWLEDGMENTS

This research was partially supported by funding from the U.S. Bureau of the Census. Thanks to Kent Norman for advice on the first study.

REFERENCES

[AW95] Ahlberg, C., Wistrand, E., "IVEE: An Information Visualization and Exploration Environment", *Proc. IEEE Information Visualization '95*, pp. 66-73, (1995).

[BC87] Becker, R., Cleveland, W., "Brushing scatterplots", *Technometrics*, 29(2), pp. 127-142, (1987).

[CS94] Chimera, R., Shneiderman B., "An exploratory evaluation of three interfaces for browsing large hierarchical tables of contents", *ACM Transactions on Information Systems*, 12(4), pp. 383-406, (Oct. 94).

[ISB95] Isakowitz, T., Stohr, E., Balasubramanian, P., "RMM: a methodology for structured hypermedia design", *Communications of the ACM*, 38(8), pp. 34-44, (August 1995).

[JBO94] Jacobson, A., Berkin, A., Orton, M., "LinkWinds: interactive scientific data analysis and visualization", *Communications of the ACM*, 37(4), pp. 43-52, (April 1994).

[KS97] Kandogan, E., Shneiderman, B., "Elastic Windows: evaluation of multi-window operations", *Proc. ACM CHI'97*, pp. 250-257, (March 1997).

[LRB97] Livny, M., Ramakrishnan, R., Beyer, K., Chen, G., Donjerkovic, D., Lawande, S., Myllymaki, J., Wenger, K., "DEVise: integrated querying and visual exploration of large datasets", *Proc. ACM SIGMOD'97*, pp. 301-312, (1997).

[Nor00] North, C., "Snap-Together Visualization", University of Maryland, Computer Science Dept. Doctoral Dissertation, (Spring 2000, forthcoming).

[RLS96] Roth, S., Lucas, P., Senn, J., Gomberg, C., Burks, M., Stroffolino, P., Kolojejchick, J., Dunmire, C., "Visage: a user interface environment for exploring

information", *Proc. Information Visualization*, IEEE, pp. 3-12, (October 1996).

[Shn92] Shneiderman, B. "Tree visualization with treemaps: a 2-d space-filling approach", ACM Transactions on Graphics, 11(1), pp. 92-99, (Jan. 1992).