Visualization of Mappings Between Schemas

George G. Robertson, Mary P. Czerwinski, John E. Churchill
Microsoft Research
One Microsoft Way, Redmond, WA
{ggr, marycz, echurch}@microsoft.com

ABSTRACT
In this paper we describe a novel approach to the visualization of the mapping between two schemas. Current approaches to visually defining such a mapping fail when the schemas or maps become large. The new approach uses various information visualization techniques to simplify the view, making it possible for users to effectively deal with much larger schemas and maps. A user study verifies that the new approach is useful, usable, and effective. The primary contribution is a demonstration of novel ways to effectively present highly complex information.

ACM Classification Keywords
H5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

Keywords
Visualization, schema mapping, hierarchy visualization, interaction techniques, XSLT, XML

INTRODUCTION
A common problem in electronic business applications is transforming data from one XML (Extensible Markup Language [7]) schema into another. For example, data may come into a company in some industry-standard schema and must be transformed into a company-specific and/or need-specific schema. Ultimately, this is done with an XSLT (eXtensible Stylesheet Language: Transformations [7]) style sheet. However, for complex schemas and mappings, defining that XSLT style sheet is very difficult.

One current, well-received solution to this problem is exemplified by the Microsoft BizTalk Schema Mapper [3], which provides a visual means of building a functional mapping from a source schema to a destination schema. Figure 1 shows a map between two simple schemas. The source schema is on the left, the destination schema is on the right, and the mapping is shown between them. The mapping is a network of *functoids* (functional operations) connected by links to schema elements and other functoids. Once a mapping is visually defined, an XSLT style sheet is compiled for use.

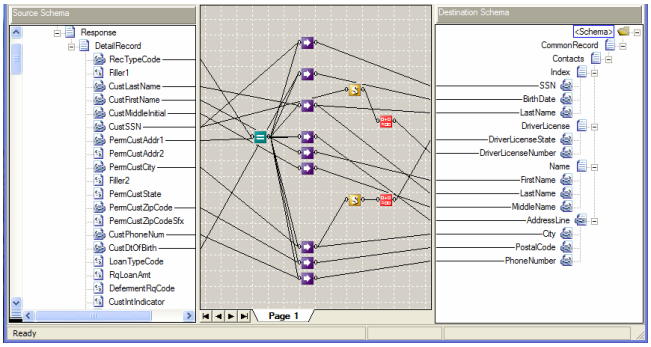

Figure 1. BizTalk Schema Mapper for a simple map.

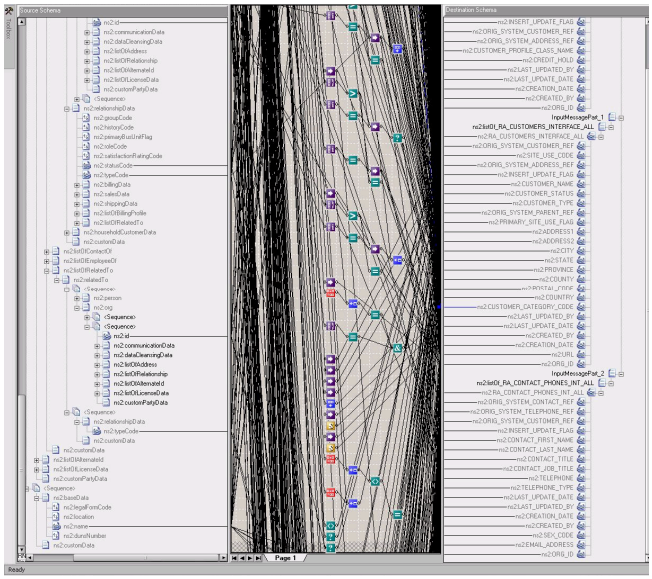

Figure 2. Example of failure to scale well for large maps.

The problem with the current solution is that it does not scale well to large schemas or large maps, and yet that is exactly what businesses need. Figure 2 is an example of such a failure, with thousands of elements in each schema and dozens of functoids. The details of interest become lost in a maze of complexity. In the current solution, a user may

CHI 2005, April 2–7, 2005, Portland, Oregon, USA.

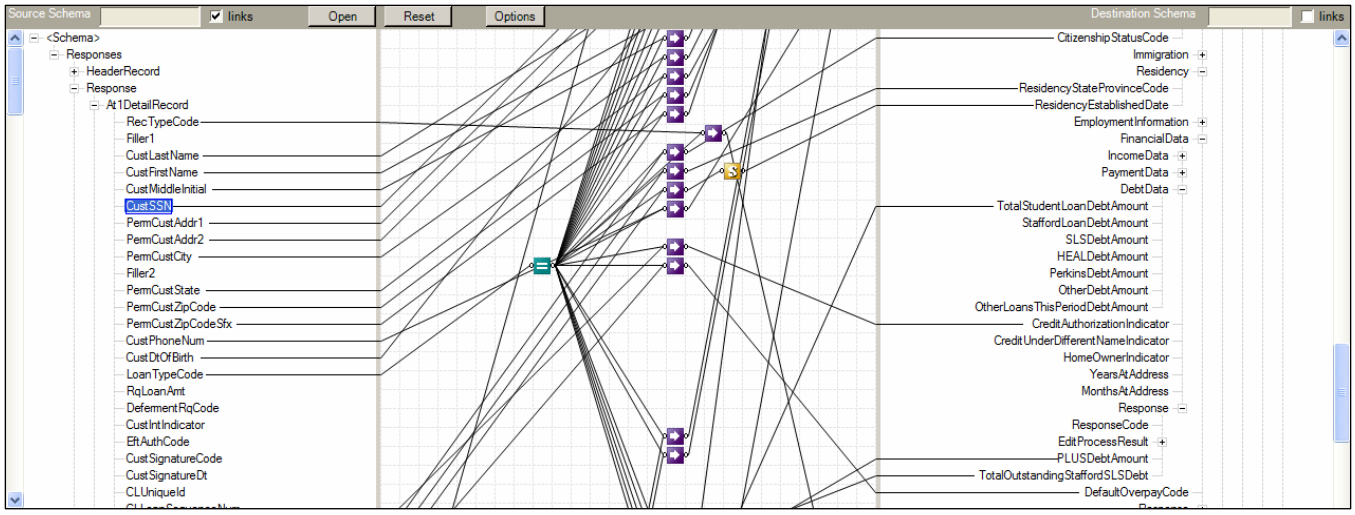

Figure 3. Baseline configuration; similar to original BizTalk Mapper

select any schema element, link, or functoid, but only that item is highlighted. This means that to find a relationship between a source schema element and a destination element requires multiple selections and much scrolling of both the schemas and the map in order to establish a reasonable view.

Through customer visits, we have collected many examples of maps in real use that are similar to or even larger than the example shown in Figure 2. An informal survey of BizTalk users indicates that complex maps like this may occur about 10% of the time, but they take up in excess of 50% of the end user's edit and creation time because of the complexity.

BizTalk's approach to schema mapping is similar to a number of other schema mapper systems, including BEA WebLogic Workshop [4], IBM WebSphere [9], TibCo BusinessWorks [16], Altova [2], Stylus Studio [15], Cape Clear [5], Sonic Software [14], and ActiveState [1]. Each of these systems has the same problems of scale, and the solutions we present could be applied to any of them.

Users of these schema mapper systems fall into two broad categories. The primary users are developers who use these tools to define how data flows in businesses. Initially, a developer spends much time creating and editing these mappings. Later, as new or different data is introduced into a business, a developer might need to revisit a mapping to make improvements. For that task, the developer may spend time navigating through the mapping to learn (or relearn) how the mapping was constructed, then spend time editing old parts or creating new parts of the mapping. The secondary users are business managers who examine the mappings to ensure that they properly reflect business process policy. These users tend to only navigate through the schemas and mappings. Both kinds of users suffer when schemas and mappings get large. Both the navigation task and the semantic editing task become harder.

Based on several years of feedback from real users of these systems, there appear to be two primary problems. First, when a developer is editing a mapping, help is needed in understanding the semantic relationship between elements and the semantic meaning of the mapping. This problem gets much harder as the schemas and mappings get large. While we do have research underway in this area, it is not the focus of this paper, as it is a less frequently performed mapping activity.

Second, basic navigation tasks (e.g., finding what schema elements are linked to what other schema elements) are very common and are seriously impaired when the schemas and mappings get large. This is the primary area of research reported on in this paper. The user study task selection and performance criteria are based on solving these navigation problems. The study participants were experienced schema mapper users and they confirmed that our task selection reflected typical tasks and accounted for where they spent most of their time while using schema mappers.

In this paper, we propose a new Schema Mapper visualization which addresses the problems of scale experienced by current users for navigation tasks. We use novel visualization and interaction techniques inspired by the information visualization community to solve these problems. After describing these new techniques, we describe the results of a user study that verifies the usefulness, usability, and effectiveness of the new techniques.

SCHEMA MAPPER VISUALIZATION IMPROVEMENTS

The basic approach to scaling to large schemas and maps is to focus on the most relevant items of interest and de-emphasize or remove items of no relevance for a particular interaction. This approach is similar to the way dynamic queries [13] are used as a filtering mechanism for visual information seeking. However, the techniques reported here

are driven by item selection rather than query sliders. We will describe the new techniques in order of importance. The figures used to illustrate each technique are screenshots from our implementation of Schema Mapper and the map used in our user study; a map from an actual BizTalk user.

We started this design process by implementing a prototype with the same visualization and behavior as BizTalk Schema Mapper. Figure 3 shows this configuration. Notice that one of the source schema elements has been selected, but only that item is highlighted.

Highlight Propagation

The most important change is to propagate highlighting whenever any item (schema element, link, or functoid) is selected. By propagation, we mean that all links are followed in both directions, and every schema element, link, and functoid that is relevant to the selected item is highlighted as well. For complex maps, that would not be sufficient because of the density of links. So, in addition, we de-emphasize all the links and functoids that are not highlighted. The de-emphasis is accomplished by drawing the links in gray and the functoids with 30% transparency. Figure 4 illustrates highlight propagation.

Notice that we still have a problem if the functoids or schema elements of interest are not visible, since the basic interactive behavior does no auto-scrolling.

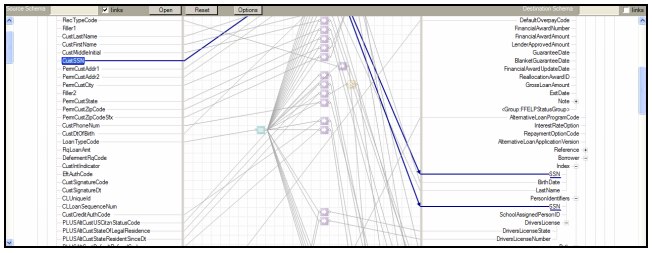

Figure 4. Highlight propagation after selecting element.

Auto-scrolling

Text editors have had some form of auto-scrolling for many years. If you type past the end of the visible region of a document, the editor auto-scrolls to the appropriate place so that you can continue typing. In a similar way, we introduce three kinds of auto-scrolling to the Schema Mapper visualization. Each technique is driven by a user selection of an item.

Auto-scroll Map

After highlight propagation, the map is auto-scrolled so that the mid-point between the top-most and bottom-most highlighted functoids is vertically centered on the vertical position of the selected schema element.

An alternative is to auto-scroll so that the mid-point between the top-most and bottom-most highlighted functoids is vertically centered on the center of the window. However, our intention is to line up the related schema elements and functoids; to accomplish that, both schemas would have to scroll as well. Informal user feedback confirmed that this alternative involved too much motion, so instead we center on the selected schema element.

Auto-scroll Columns within Map

When the map is auto-scrolled, it is often still the case that the functoids along the selected path (between source schema element, functoids, and destination schema element) are not aligned. This happens partly because the maps are laid out by hand and partly because of inherit complexity of the linking network. To address this problem, we use a technique originally suggested by the Cone Tree visualization [12]. In Cone Trees, the path from a selected node to the root is centered, so that at each level of the tree the appropriate node is centered. The 2D equivalent is to move each column of the map so that the next functoid on the selected path is centered on that path. This is only done for the top-most highlighted functoid in each column. The result is that the entire selected path is centered on the originally selected schema element.

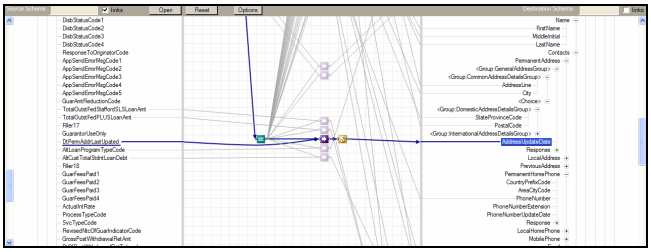

Figure 5. Auto-scrolling of map, columns, and schemas.

Auto-scroll Schema Tree Views

To complete this centering, we also need to center the highlighted schema element(s) in the related schema. When we do this, we would like to animate the scrolling of the node in the tree view to its new location. This is inspired by the Polyarchy visualization [11], which used animated transitions to ensure that users do not lose track of the nature of complex transitions. The standard Windows TreeView control [17] does not support centering of a node or animated scrolling. We introduced these two techniques into a modified version of the TreeView control to allow their use in the Schema Mapper visualization.

Figure 5 shows the effects of these three auto-scroll techniques after selecting a destination schema element. While this helps a great deal, we still may have a problem if the distance between related schema elements is large, because of intervening information not relevant at the moment. Figure 5 shows such a case, with one of the highlighted source schema elements not visible.

Coalescing Trees

To address the problem of non-relevant information being displayed in the schema tree view, we introduce a method for coalescing nodes deemed not relevant at the moment. This technique was partly inspired by the Polyarchy

visualization [11], which only shows tree nodes relevant to the current user query. It was also inspired by the Favorite Folders technique [7], which provides a way to manually mark which folders to keep in a tree view. But instead of marking items manually, Schema Mapper uses implicit relevance based on two factors: whether a schema element or any of its descendants has a link, and the selected and highlighted schema elements. Figure 6 shows the result of the same selection as in Figure 5, but with coalescing trees. Notice that the highlighted source schema elements are now much closer together and fully visible on the screen.

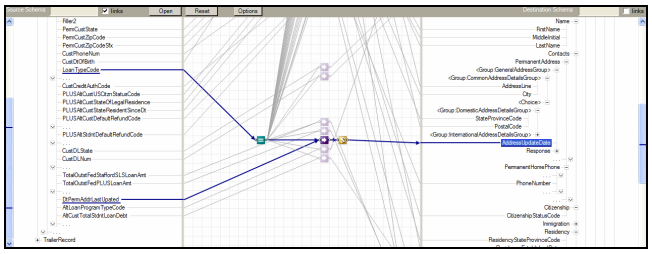

Figure 6. Coalesced trees.

Figure 7 shows a close-up view near one of the coalesced nodes of the source schema. Hovering over a node will produce a tooltip that describes what has been coalesced. If you click on the coalesced node's down-arrow, the coalesced nodes are made visible and the down-arrow becomes an up-arrow. Clicking on the up-arrow will re-coalesce those nodes. Notice that there are multiple coalesced node sets at the same level in this hierarchy. Favorite Folders actually puts all of its coalesced nodes for a level into one ellipsis at the end of the level. However, for schema management, the order of the nodes has meaning; hence, it is desirable to have coalesced node sets appear in place rather than being combined.

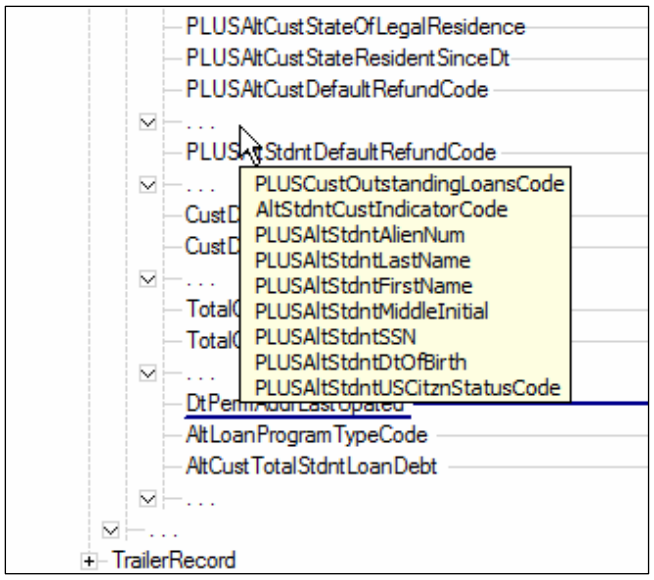

Figure 7. Close-up of coalesced nodes with tooltip.

Multi-Select

On some occasions, a user wants to know how multiple schema elements interact. Currently that requires sequential selection and human memory. We have added multiple selection capability to the Schema Mapper visualization to address this problem. The first selection is done with a single click of the mouse button. Additional selections are done with a mouse click while holding down the Shift key. Figure 8 shows selection of three elements of the source schema. Notice that highlight propagation and auto-scroll are driven by the multiple selections.

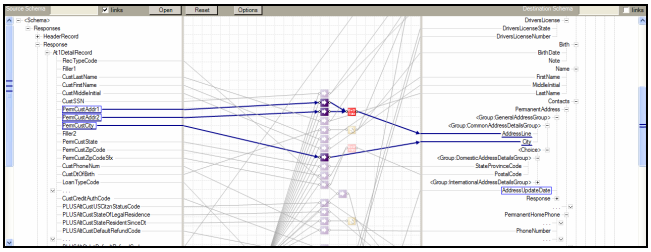

Figure 8. Multi-selection of three source schema elements.

Incremental Search

Coupling search capabilities with these new visualization features should provide powerful and fast ways for the user to find relevant information. We have added a search type-in box above each schema (see Figure 10), along with a checkbox to indicate whether the search should be done only on linked elements or on all elements. The search is incremental, in that it shows the results after each keystroke in the type-in box. The multi-select mechanism is used to automatically show all the relevant information for the search hits. Figure 9 shows the results of an all-elements search for the string "ssn" in the destination schema.

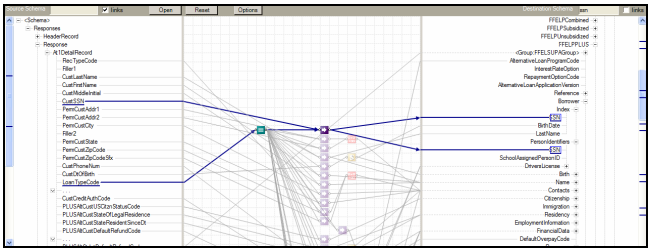

Figure 9. Search for "ssn" in destination schema.

Automatic Parental Tree View Collapse

To display the most appropriate results of an incremental search, we want to show only the nodes in the searched schema that have search hits or have descendants with search hits. All other nodes should be collapsed. This is inspired by the Polyarchy visualization [11], which also shows only the information relevant to the query. To implement this required changing the basic behavior of the TreeView control, to support automated collapsing of un-marked nodes.

Interactive Scrollbar Highlighting

Notice that the scrollbars in Figure 10 have tick marks. These represent all of the search hits in the entire schema. The scrollbar highlighting was inspired by similar search results scrollbar highlighting in the DateLens [6] calendar visualization. The difference here is that the scrollbar's highlighted tick marks are interactive. If you hover over one, a tooltip will describe the element. If you click on one, an animated scroll will bring the desired element to the center of the tree view, or as near to the center as possible.

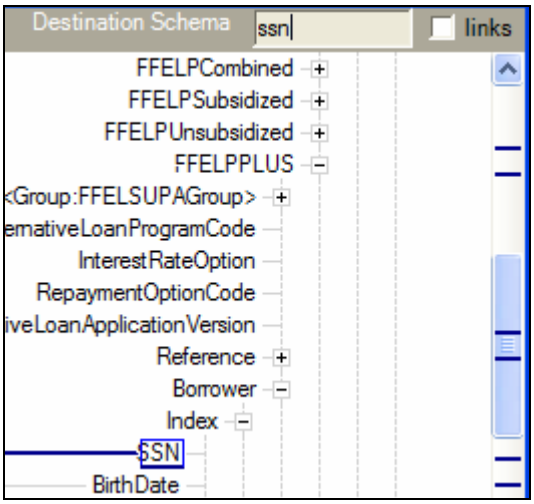

Figure 10. Search type-in box and scrollbar highlights.

The tick marks are color-coded. Blue marks represent elements that are selected. Red marks represent search hits that are not currently selected. After a search, all of the tick marks will be blue because they are multi-selected. Each time the user types the Enter key, the system will (single) select the next search hit. Shift-Enter will (single) select the previous search hit. This gives the user a way to see all search hits simultaneously (the default) or to sequence through individual search hits. When sequencing through individual search hits, one of the ticks will be blue and the others will be red. Figure 10 is showing the default.

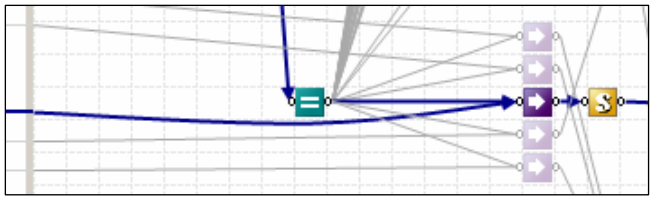

Figure 11. Close-up of bent link.

Bendable Links

When highlighting a path between schema elements, occasionally a link will pass directly behind a functoid. When this happens, it becomes visually ambiguous; the user cannot be sure if the link connects to the functoid or not. In addition, the link may obscure an existing link to the functoid. To avoid these problems, we detect this case and visually bend the link around the functoid. Figure 5 shows an example of this, where a link has been bent below a functoid. Figure 11 shows a close-up of the same link. Since we know which functoid to avoid, drawing a bent link is done simply by drawing an arc with a control point either directly above or below the functoid being avoided.

Focus on Linked Elements

Like the original BizTalk Schema Mapper, our Schema Mapper visualization collapses tree view nodes that have no linked elements as descendants. To complete this visual behavior, we augment the TreeView interactive behavior to match the focus on linked elements. In particular, the use of the up/down arrow keys to advance to the previous/next element has been modified to advance to the previous/next linked element. If the user holds down the Shift key while typing the up/down arrow key, the original behavior is performed.

USER STUDY

In order to test the usability and usefulness of the redesigned Schema Mapper visualization, we built a fully functioning prototype of the tool in a manner that allowed us to systematically turn on and off particular features. This provided us with the ability to incrementally investigate the influence of each in comparison to the baseline version of how the Schema Mapper works in the original BizTalk mapper. Therefore, our baseline Schema Mapper visualization was simply the existing user interface, with one critical addition. All four versions of the mapper that we tested included a search capability for both the source and destination schema tree controls. This was also true for the baseline condition (feature set A), shown in Figure 3. The second condition in our study (feature set B) included highlight propagation, as described earlier (see Figure 4). The third condition (feature set C) involved a version of the mapper that included these features, but also added the three auto-scrolling mechanisms described earlier (auto-scroll map, auto-scroll columns, and auto-scroll schemas), as shown in Figure 5. The final condition in our study (feature set D) used a version of the mapper that included all of these features and in addition provided sibling coalescence and search result tick marks to visualize search hits in the scrollbar (Figure 6).

So, the study was a four-way within-subjects design of various incremental improvements to the Schema Mapper visualization. To control for order effects, the order in which participants experienced each of the versions of the mapper was counterbalanced using a partial Latin Square design (partial because only 8 participants were run through the study, so all orders could not be tested; however, the squares were balanced).

Participants

Eight very experienced BizTalk users were recruited for participation. From an analysis of a background questionnaire, the participants had an average of 21 years of

Task Set A	Task Set B
Go to the last linked element in the source schema and find out what it's connected to in the destination schema.	Go to the second to last linked element in the source schema and find out what it's connected to in the destination schema.
Go to the first linked element in the source schema and tell me what it is connected to in the destination schema.	Go to the last linked element in the destination schema and tell me what it's connected to in the source schema.
Go to the second scripting functoid and determine what it's connected to in both the source and the destination schemas.	Go to the first scripting functoid and determine what it's connected to in both the source and destination schemas.
Find the last value mapping functoid and find out what it is connected to in both the destination and source schemas.	Find the first value mapping functoid and find out what it is connected to in both the destination and source schemas.
Find the source schema element that has the most connections to destination schema elements.	Find all of the source schema elements that are directly connected to a destination schema element (i.e., no functoids are between them).
How many times does "Signature" occur in the Destination schema, whether linked or not? How many items are linked to it?	How many times does SSN occur in the destination schema with something linked to it?

Table 1. Two of the task sets used in the study.

computer experience, over four years of experience using BizTalk, on average, and were 38.4 years old, on average (ranging from 28 to 49 years old).

Tasks

Six tasks involving finding elements in the source and target schema maps, their related functoids and their connections were devised. An effort was made to keep the tasks isomorphic so that the participants experienced similar tasks as they viewed each version of the mapper. To ensure that no one task set was accidentally more difficult than the rest, however, we rotated the task set through the visualizations. Two of the task sets are shown in Table 1 by way of example. The map that was used for the experiment was typical of the kinds of maps created in large corporate organizations, and came from a BizTalk customer. Figures 3-11 were created with the map used in the study. The aspect ratio of the window used for the study, as shown in those figures, was chosen to require scrolling of both schemas and the map for these tasks. This essentially simulates the behavior required for a more traditional aspect ratio on larger schemas and a larger map. All sessions were run with a single participant and lasted around one hour. Participants received lunch/dinner coupons for the local cafeteria for their participation.

RESULTS

Task Times

A 4 (mapper feature sets A-D) x 6 (tasks) Analysis of Variance (ANOVA) with repeated measures was carried out on the task time data, both with and without a log transformation of the task times (log transformations are utilized to reduce the heavy skewing in response time data in order to better adhere to the assumptions of ANOVA). The pattern of significant results observed for both tests was the same, so the results from the logged data will be presented. A significant main effect was obtained for the mapper feature set used, $F(3,21)=45.1$, $p<.001$, and the task, $F(5,35)=9.01$, $p<.001$. In addition, a significant interaction between the mapper feature sets and tasks was obtained, $F(15,105)=2.3$, $p<.001$.

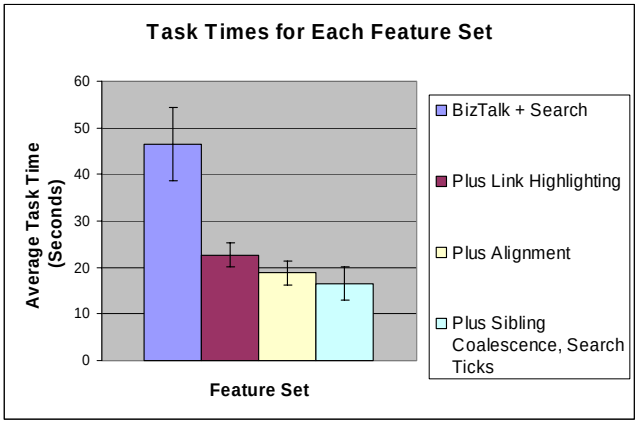

Figure 12. Average task times for each of the four feature sets for the Schema Mapper visualization.

Post-hoc analyses (with Bonferroni corrections for multiple tests) showed that feature set A (the original Schema Mapper plus search) was significantly slower than each of the other feature sets at the $p=.05$ level. In addition, feature

set B (adding highlight propagation) was significantly slower than feature set D (all features including sibling coalescence) at the p=.05 level. No other differences were observed between the features sets, with sets C (highlight propagation and auto-alignment) and D not significantly different from each other. These results are shown in Figure 12.

Interaction between feature set and tasks reveals that certain tasks were harder than others (in particular tasks 3 and 5), and that some tasks (e.g., 1-4) were especially difficult when using the original Schema Mapper, feature set A. The average task time by trial data is shown in Figure 13.

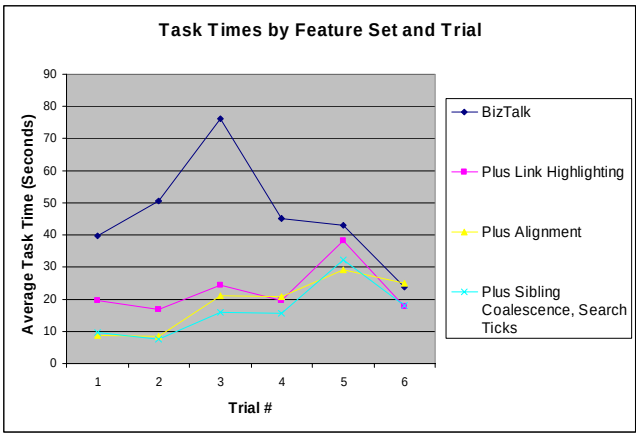

Figure 13. Average task times by feature set and task (trial). Note that tasks 3 and 5 were most difficult, with tasks 1-4 being especially difficult when using feature set A, the original Schema Mapper.

Satisfaction Data

A user satisfaction questionnaire was completed by participants at the end of the study session. To improve methodological rigor, some of the statements were asked in a favorable way toward the mappers tested, and some were phrased in a negative manner. Responses were collected using a 7-point Likert scale with 1=Disagree (or Low) and 7=Agree (or High). In order to improve readability and to analyze the data using a single ANOVA, questions which required a lower response to reflect positive satisfaction were flipped (e.g., if the user rated a question with a 1, meaning the highest possible value, it was flipped to a 7) prior to analysis.

A 4 (mapper feature sets A-D) x 10 (questionnaire question) repeated measures ANOVA was performed on the users' ratings. A significant main effect for mapper feature set was observed, $F(3,21)=26.1$, $p<.001$, as was a significant main effect for questionnaire item, $F(9,63)=4.6$, $p<.001$. A significant interaction was also obtained, $F(27,189)=3.1$, $p<.001$. Post-hoc analyses utilizing Bonferroni adjustments for multiple tests revealed that mapper feature set A was rated significantly lower than each of the other feature sets, but there were no other significant differences between feature sets B-D in terms of their satisfaction ratings. The overall average ratings for each feature set are shown in Figure 14. The interaction by questionnaire item simply underlines the fact that some of the questionnaire items were more sensitive to the user interface manipulations used in this study than were others. All of the satisfaction data is included in Table 2.

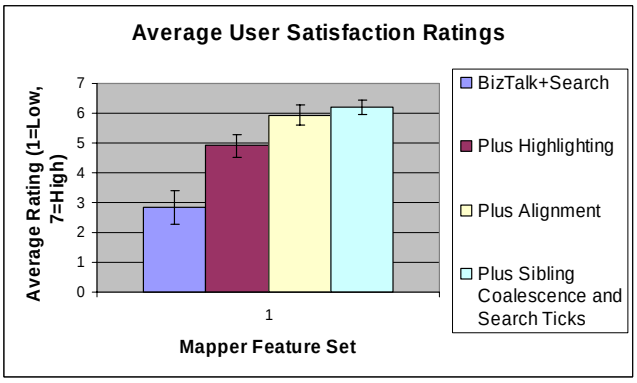

Figure 14. Average user satisfaction ratings for the four mapper feature sets studied.

Usability Issues

We did observe some usability issues, which we intend to improve in future designs. For example, even with the enhancements, some users would still not see a second source link to a functoid or target element because they had to scroll the map, or because there were so many links. They wanted a "functoid overview" (e.g., a right-click on the map to get an overview that is navigable). In addition, some users were confused as to why a previous selection was still highlighted in the hierarchical tree control after they performed a search. Users were a little confused about the difference between the red and blue search result ticks. One user who figured it out thought that if "fuzzy search" is allowed, as he called it, it should be an option that the user sets specifically. He thought the default should be string search starting from the beginning of the node name. This user also claimed that users typically know node names very well. One user did not like the fact that functoids were grayed out. He wanted all functoids to remain fully rendered. Several users requested a search feature that searched through functoid scripts for keywords. Alternatively, they requested a filter on functoid types. Two users asked for a numeric count to appear next to the search box so they did not have to count the ticks or the highlights in the search results. Some users did not understand that when "links" was turned off during search, it meant searching both linked and unlinked items. A few users thought it was only searching through unlinked items, thinking the checkbox was a toggle.

Overall, however, the participants overwhelmingly preferred the new feature sets over the existing Schema Mapper, and they preferred the version of the mapper with all the features (set D, including sibling coalescence and the search result ticks in the scrollbar) over all the others unanimously.

Questionnaire Item (note that negatively phrased questions have been positively rephrased for expository reasons).	**Biztalk + Search (set A)**	**Plus Highlight Propagation (set B)**	**Plus Auto-Scroll (set C)**	**Plus Sibling Coalescence and Search Ticks (set D)**
Ease of finding elements in map?	3.4	5.8	6.3	6.6
Ease of aligning related elements?	1.3	2.5	6.1	6.8
Overall ease of use?	2.3	5.0	5.9	6.1
Discoverability of features?	3.4	5.4	6.4	5.5
Mental load?	3.0	4.8	5.0	5.9
Physical load?	2.5	4.3	5.3	6.0
Temporal demand?	3.5	4.9	6.0	6.0
User performance?	3.0	5.8	6.4	6.0
Overall satisfaction?	2.9	5.4	6.1	6.5
Lack of frustration?	3.3	5.5	5.9	6.6

Table 2. Average user satisfaction ratings for the 4 versions of the mapper (1=negative, 7=positive). Higher ratings indicate higher satisfaction for all questions.

DISCUSSION

A usability study, run with expert BizTalk users, systematically investigated various feature additions to the Schema Mapper visualization. Study results revealed a significant time advantage for each of the new feature sets over the existing user interface. In addition, user satisfaction ratings corroborated those performance results, with the new feature sets receiving significantly higher ratings than the original Schema Mapper. Comments from study participants assured us that our tasks had very high ecological validity, and that they hoped these new features will be made available. Finally, usability issues were observed that should be addressed in future designs.

IMPLEMENTATION DETAILS

Many of the techniques introduced in this paper could become part of a new TreeView control and be put to use in a variety of other contexts. These new features include multiple selection, animated scrolling, centering, coalesced tree nodes, and incremental search with interactive scrollbar highlighting.

These techniques were implemented on top of the current Windows TreeView control in C# using the .Net framework. In order to abstract these features, new classes (AnimTreeView and AnimTreeNode) were introduced to replace the standard TreeView and TreeNode classes. Since some features are logically associated with the mapper application rather than the tree control, a MapTreeView class was introduced as a specialization of AnimTreeView. For example, the rendering of the link from a schema element name to the boundary of the map is done in the OnTreeNodePostPaint method of the MapTreeView class. Highlighting of schema elements is also done in this method. This method is invoked in one of two ways. If coalesced tree nodes are totally disabled, the standard TreeView rendering takes place and the NMCustomDraw interface is used to get control after the TreeView control paints a node. However, coalesced tree node support requires a totally different rendering of the tree because vertical placement of the nodes is different and ellipses are added. In this case, the style of the control is set for user painting (UserPaint & AllPaintingInWmPaint) and mouse control (UserMouse). The entire tree is rendered in the OnPaint method and OnTreeNodePostPaint is invoked after the node is rendered.

Centering and animated scrolling is accomplished with a 50 millisecond timer which uses SetScrollPos to adjust the position of the vertical scrollbar, and therefore the part of the tree that is rendered. In addition, a WM_VSCROLL message is sent to adjust the scroll thumb size and position.

Interactive scrollbar highlighting was the most challenging to implement because the standard TreeView uses a private implementation of scrollbars, hence specializing the standard scrollbar class was not an option. To get around this problem, we used a transparent forms panel which we call an SBOverlay, and placed it on top of the vertical scrollbar. Any time the TreeView changes size (e.g., when the window size is changed), the corresponding SBOverlay size and position is updated so that it always fits directly on top of the vertical scrollbar. The scrollbar highlighting marks are drawn in the SBOverlay OnPaint method. The other problem is that the TreeView's scrollbar gets re-

rendered at various times. To catch re-render events, we monitor the WndProc message stream for WM_HSCROLL, WM_VSCROLL, WM_NCMOUSELEAVE, and MC_NCMOUSEMOVE messages and update the scrollbar overlay at those times. This works for all cases except that there is some flicker in the overlay while the scrollbar thumb is being dragged. Obviously, a new implementation of the TreeView control could solve that problem.

FUTURE DIRECTIONS

The visualization work reported here has focused on non-editing scenarios. To enable these new features to work in an editing environment, the coalesced tree view must be enhanced so that if a coalesced node is a drag and drop target, it will temporarily un-coalesce. It should re-coalesce if the drop target changes and coalescence should be re-evaluated if a new link is created to an element that had been coalesced.

CONCLUSION

Mapping between two schemas is an increasingly common business need and current techniques for visually defining mappings between schemas do not scale well. A significant contribution of this paper is that we have described a series of visualization improvements that enable practical use of much larger schemas and maps. The new techniques were inspired by several existing information visualization techniques. We have demonstrated the usefulness, usability, and effectiveness of these new techniques with a user study, and identified directions for future work.

Finally, this work can easily be generalized for a wide variety of applications. About half of the techniques described here involve improvements to the Windows TreeView control, which is used in hundreds of existing applications. The other half involves autoscrolling and highlighting techniques which could be applied across a wide set of interfaces and visualizations.

ACKNOWLEDGMENTS

We thank Bongshin Lee, Patrick Baudisch, Greg Smith, Brian Meyers, Prasad Sripathi Panditharadhya, Alvaro Miranda, Tan Bao Nguyen, Tatyana Yakushev, Uday Bhaskara and A.S. Sivakumar for feedback on design issues. We thank Kaivalya Hanswadkar for help in recruiting participants for the user study. We thank the user study participants for their thoughtful suggestions.

REFERENCES

1. ActiveState Visual XSLT. http://www.activestate.com/Products/Visual_XSLT.
2. Altova XML-to-XML Mappings. http://www.altova.com/features_xml2xml_mapforce.html
3. BizTalk Schema Mapper. http://msdn.microsoft.com/library/en-us/introduction/htm/ebiz_intro_story_jgtg.asp
4. BEA WebLogic Workshop. http://www.bea.com/framework.jsp?CNT=demos.htm&FP=/content/products/workshop/learn
5. Cape Clear XSLT Mapper. http://www.capescience.com/education/tutorials/index.shtml#cc5
6. Bederson, B, Clamage, A., Czerwinski, M., & Robertson, G., DateLens: A fisheye calendar interface for PDAs. In *ACM Transactions on Computer-Human Interface*, ACM Press (2004), 11, 1, 90-119.
7. Extensible Markup Language (XML). http://www.w3.org/XML
8. Kay, M., *XSLT Programmer's Reference*. Wrox Press Ltd., Birmingham, UK, 2000.
9. Lau, C., Developing XML web services with websphere studio application developer. In *IBM Systems Journal*, July 2002.
10. Lee, B. & Bederson, B., Favorite folders: a configurable, scalable file browser. Demo paper in *UIST 2003 Conference Supplement*, 2003.
11. Robertson, G., Cameron, K., Czerwinski, M., & Robbins, D., Animated visualization of multiple intersecting hierarchies. In *Journal of Information Visualization*, Palgrave (2002), 1, 1, 50-65.
12. Robertson, G., Mackinlay, J., & Card, S. Cone Trees: Animated 3D visualizations of hierarchical information. In *Proceedings of CHI'91*, ACM Press (1991), 189-194.
13. Shneiderman, B., Dynamic queries for visual information seeking, *IEEE Software*, 11, 6 (1994).
14. Sonic Software Integration Workshop. http://www.sonicsoftware.com/products/docs/integration_workbench_0604.pdf
15. Stylus Studio XSLT Mapper. http://www.stylusstudio.com/xslt_mapper.html.
16. TibCo BusinessWorks. http://www.tibco.com/resources/software/business_integration/bw_scrn_pop.html
17. Windows TreeView Control. http://msdn.microsoft.com/library/en-us/cpref/html/frlrfsystemwindowsformstreeviewclasstopic.asp