Guidelines for Using Multiple Views in Information Visualization

Michelle Q. Wang Baldonado, Allison Woodruff
Xerox Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, CA 94304 USA
+1 650 812 4797, +1 650 812 4429
{michelle, woodruff}@parc.xerox.com

Allan Kuchinsky
Hewlett Packard Laboratories
1501 Page Mill Road
Palo Alto, CA 94304 USA
+1 650 857 7423
kuchinsk@ hpl.hp.com

ABSTRACT
A *multiple view system* uses two or more distinct views to support the investigation of a single conceptual entity. Many such systems exist, ranging from computer-aided design (CAD) systems for chip design that display both the logical structure and the actual geometry of the integrated circuit to overview-plus-detail systems that show both an overview for context and a zoomed-in-view for detail. Designers of these systems must make a variety of design decisions, ranging from determining layout to constructing sophisticated coordination mechanisms. Surprisingly, little work has been done to characterize these systems or to express guidelines for their design. Based on a workshop discussion of multiple views, and based on our own design and implementation experience with these systems, we present eight guidelines for the design of multiple view systems.

Keywords
Multiple views, information visualization, design guidelines, usability heuristics, user interfaces

INTRODUCTION
Multiple view systems—systems that use two or more distinct views to support the investigation of a single conceptual entity—are both common and useful [6,12,16,20,27,28,29,30]. Neurophysiologists at the University of Pittsburgh Medical Center [25] recognized the value of multiple views when they considered extending a multimedia system to support the task of identifying seizures in infants. These seizures are very subtle events and it is difficult to identify seizure activity using a single source of data. Their conclusion was that the identification process would be significantly improved by simultaneous review of physiological data and visual observation of the infant's movements.

As the initial example suggests, multiple view systems offer a variety of benefits. For example, North and Shneiderman observe that multiple window coordinations offer the following advantages: improved user performance; discovery of unforeseen relationships; and unification of the desktop [19].

However, multiple view systems are highly challenging to design. They often use sophisticated coordination mechanisms and layout. In addition, subtle interactions among the many dimensions of the design space complicate design decisions.

The fact that many unnecessary design mistakes are made in multiple view systems was made clear to the authors of this paper when we participated in the multiple views subgroup of the CHI '98 Workshop on Innovation and Evaluation in Information Exploration Interfaces, organized by Nicholas J. Belkin and Gene Golovchinsky [14]. Many members of this subgroup had implemented multiple view systems for information exploration, e.g., [2,11,15,17]. These implementers had observed complexities and inconsistencies in their own systems as well as in others. Such mistakes are particularly serious because, once made, they are often difficult to correct due to implementation intricacies inherent in multiple view systems.

The members of the workshop felt that the design process for multiple view systems could be improved by usability heuristics. Design guidelines for general user interfaces [18,26] are certainly of value. More closely related are the design guidelines that have been developed for multimodal systems [10]. Nonetheless, little specific guidance is currently available to designers of multiple view systems. For example, when considering the many general guidelines that exist, how does the designer know which are most salient to multiple views? Are there

AVI 2000, Palermo, Italy.

customized versions of these guidelines appropriate for multiple views? Further, what guidelines exist that are specific to multiple views? The members of the group felt that their collective knowledge could serve as the foundation for guidelines in this underanalyzed area. Therefore, after the workshop ended, the authors of this paper resolved to document their experiences.

In this paper, we present the results of these efforts in the form of design guidelines that we hope will be useful in heuristic walkthroughs [18] of both designs and fully implemented systems. In some cases, we present customized versions of general guidelines that address issues that are particular to multiple view systems. In other cases, we present guidelines that are largely unique to multiple view systems. We illustrate these guidelines through examples. Due to space constraints, we can not review all multiple view systems; however, many other interesting examples appear in [20]. Multiple views are common in a variety of environments, ranging from video games to book illustrations to television monitoring systems. Although much of our work extends to these domains, we focus in this paper primarily on multiple views in information visualization.

In the next section, we discuss costs and benefits the designer considers while working on a multiple view system. In the following section, we present a definition and a model of multiple view systems. In the subsequent sections, we present eight design guidelines. We organize these guidelines into two sets. First we discuss guidelines to help designers decide when multiple views are desirable; we hope these guidelines will help designers avoid unnecessary complexity in their systems. If the designer does decide a multiple view system is warranted, they incur a number of costs, e.g., cognitive overhead on the user. Therefore, we next discuss guidelines for the use of multiple views; we hope these guidelines will minimize the costs of using multiple view systems. As in any design situation, there exist trade-offs among these rules. Where space permits, we identify and discuss important trade-offs. In the final section, we conclude and suggest future directions.

COST-BENEFIT TRADEOFFS

Deciding when and how to apply multiple views to information visualization problems involves balancing a complex set of design tradeoffs. On the one hand, multiple views can provide utility in terms of minimizing some of the cognitive overhead engendered by a single, complex view of data. On the other hand, multiple views can decrease utility when added to a system, both in terms of higher cognitive overhead (e.g., for context switching) and in terms of increased system requirements. As we present our guidelines, we identify how they significantly impact cognitive overhead and system requirements.

The cognitive aspects of an information management task include:

- the time and effort required to *learn* the system
- the *load* on the user's working memory
- the effort required for *comparison*
- the effort required for *context switching*

The impact on system requirements engendered by multiple views include:

- *computational requirements* for rendering the additional display elements
- *display space requirements* for the additional views

In addition to considering the utility to the user, the designer must also take into account the resources required to design, implement, and maintain the system.

MODEL

In this section, we first define multiple view systems and then propose a model of multiple view systems that is based on three different dimensions: *selection* of views, *presentation* of views, and *interaction* among views. These dimensions emerged for us as we evolved a questionnaire (available from the authors upon request) for use in analyzing a number of existing multiple view systems. We have found them valuable for characterizing and critiquing multiple view systems.

Designers of multiple view systems necessarily begin by establishing a clear understanding of the user's task. The next step is to architect a system that is likely to be valuable to the user in accomplishing this task. Our hope is that the model we present here will give system designers useful language for articulating the structure of their systems (naturally, many other models of multiple view systems are also possible and valuable). With the user's needs and the system's architecture made explicit, designers will be well positioned to design and evaluate their systems according to the guidelines presented in the next two sections. In addition, designers will have a solid foundation on which to base user studies in order to engage in iterative design.

Definition

We define a *single view* of a conceptual entity as a set of data plus a specification of how to display that data visually. Note that a display may be either textual, e.g., in tabular form, or graphical, e.g., in a bar chart.

We say that views are *distinct* if they allow the user to learn about different aspects of the conceptual entity, e.g., by presenting different information, or by emphasizing different aspects of the same information. A *multiple view system* uses two or more such distinct views to support the investigation of a given conceptual entity. As a simple

example, Microsoft PowerPoint™ supports several views of a presentation. The slide view shows each slide individually in detail while the slide sorter view gives an overview of the presentation.

Our design guidelines and examples are oriented towards systems that present a coordinated set of views to a single user performing a given task. However, many of our rules extend immediately to multiuser systems in which each user sees only one view or systems in which each user uses different views to perform independent tasks.

Views can differ in their data or in the visual representation of that data. For a more detailed discussion of mapping data to visual form, see [8]. Some examples of ways in which data sets can differ include:

- One data set can be a subset of another. For example, one view might show all stock prices for a given time period, while another view focuses on the stock prices for a specific company during the same time period.
- One data set can contain aggregates of the individual values of a second data set. For example, one view might show average costs for each type of restaurant in an area, while a second view shows the location of each individual restaurant.
- Data sets can contain entirely different information. For example, a computer-aided design (CAD) system for chip design might show a schematic that represents the logical structure of an integrated circuit in one view and a detailed graphical layout representing the actual geometry of the circuit to be fabricated in a second view.

Regardless of whether the data sets differ, visual representations can differ, e.g., one view can show a bar chart while another view shows a scatterplot.

With this definition of multiple view systems in place, we can now delve deeper into the underlying dimensions of selection, presentation, and interaction. For each dimension, we articulate some of the design issues involved.

Selection

The first phase in the design process is the identification of a set of views to be used in a coordinated fashion in support of a given task. Note that some combinations of views may not be meaningful or interesting.

Presentation

Once a set of views has been chosen, the designer faces a number of issues related to their presentation. One issue is that views can be presented sequentially (for example, the user may use a menu to toggle between different views) or simultaneously. Another issue is that if multiple views appear at once, there are many possible configurations of these views on the screen.

Interaction

We next consider the interaction mechanisms supported by views. Each single view may have independent affordances, e.g., selection capabilities or navigation functionality such as pan and zoom. Often, these affordances are tied together so that actions in one view have an effect in another view.

One common interaction technique is *navigational slaving*, in which movements in one view are automatically propagated to other views.

Another common interaction technique is *linking*, which connects data in one view to data in another view [29]. A specific type of linking is *brushing*, in which the user highlights items in one view and the corresponding items in another view are highlighted by the system [3]. As a concrete example, imagine that two views, a scatterplot and a map, present a number of restaurants. Further imagine that if the user selects a restaurant in the scatterplot (e.g., the least expensive French restaurant), that same restaurant is highlighted in the map view. [7] discusses other types of linking, including spatial and temporal linking in animated sequence of views.

Both slaving and linking are typically governed by a *coupling function* that specifies a mapping from objects or navigational position in one view to objects or navigational position in another view. In the restaurant example, suppose that both the scatterplot and the map present the same data set and that each restaurant has a unique identifier. An elementary coupling function might specify that an object in one view is connected to the object with the same unique identifier in the other view.

Assuming that such coupling has been established, the designer must decide when to propagate interaction events from one view to the next. The designer must also have a model for propagating events across more than two views. For example, the propagation model might be transitive.

WHEN TO USE MULTIPLE VIEWS

This first set of guidelines addresses the question of view selection. Designers must make a cost/benefit trade-off between the benefits of multiple view systems and the corresponding complexity that arises. In this section, we introduce four design rules (diversity, complementarity, parsimony, and decomposition) to help designers and users assess whether or not multiple view systems are appropriate for their applications.

Rule of Diversity

Use multiple views when there is a diversity of attributes, models, user profiles, levels of

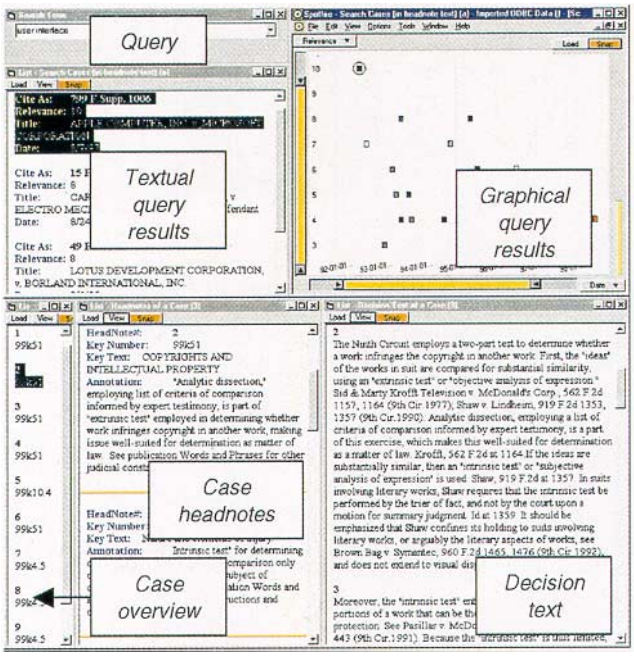

Figure 1: A multiple views presentation of diverse information relating to legal cases [20].

abstraction, or genres. A single view that accommodates many needs is likely to be a least-common-denominator view that is not optimal for any needs. Such a view may create significant cognitive overhead for the user by requiring them to simultaneously comprehend and assimilate a multitude of diverse data, some of which may not be relevant to their needs.

The presence of diversity is one of the foremost reasons for designing a multiple view system. In particular, multiple views are useful when one of the following types of diversity is present:

- **attributes:** e.g., creation dates vs. color histogram data for items in an image database
- **models:** e.g., the logical structure vs. geometric layout of an integrated circuit in a CAD system
- **user profiles:** e.g., preferences, levels of expertise, roles
- **levels of abstraction:** e.g., a detailed street map vs. an overview map of a metropolitan area
- **genres:** e.g., a block diagram vs. pseudo-code views of a software module

For example, if different levels of abstraction (increasingly detailed layers) are present, multiple views can support progressive disclosure. Alternatively, if different data models are present, each model may be most appropriately represented using a separate visual representation.

Figure 1 depicts a system that exemplifies the use of multiple views for data with different attributes and different levels of abstraction [20]. This tool shows legal information using the following views: query, query results in textual (bibliographic) form, query results in a graphical visualization, and, for the selected case, overview, headnotes, and decision text, all in textual form. Each of these views shows different sets of attributes, ranging from bibliographic fields to search relevance to case metadata. Further, while the cases are shown at a high level of abstraction in the bibliographic and graphical views of search results, they are shown in greater detail in the case overview, and they are shown in even greater detail in the headnotes and full decision text. Through these multiple views, the user can gain different perspectives on the cases at hand.

Rule of Complementarity

Use multiple views when different views bring out correlations and/or disparities. In a single view, a user may need to mentally extract and remember components they wish to compare. Maintaining and switching among these components can be cognitively demanding. Just as recognition is easier to perform than recall, so visual comparison is easier to accomplish than memory-based comparison. Multiple views leverage perceptual capabilities to improve understanding of relations among views.

Multiple views can help users understand complex relationships among different data sets. They are particularly helpful when coupling two or more views shows otherwise hidden relations.

The variety of information conveyed by the different views of Figure 2 demonstrates how a comprehensive understanding of a complex structure can be composed from multiple, complementary views of that structure. RasMol [24] is a tool for visualizing the molecular structure of proteins. Understanding the structure of a newly discovered protein or gene product is a critical aspect of the drug discovery process for pharmaceutical companies. A deep understanding of a protein's structure enables a pharmaceutical researcher to infer its function and, thus, its potential therapeutic usage.

The wireframe image in the upper left corner of Figure 2 shows the chemistry of the barnase molecule. The sequence of amino acids displayed here represents the primary structure of the protein. The spacefilling model in the lower left corner of Figure 2 shows the size and surface of the molecule. However, in order to have a complete picture of the protein structure it is also necessary to understand the details of chemical bonding that cause the protein to fold into a 3-dimensional

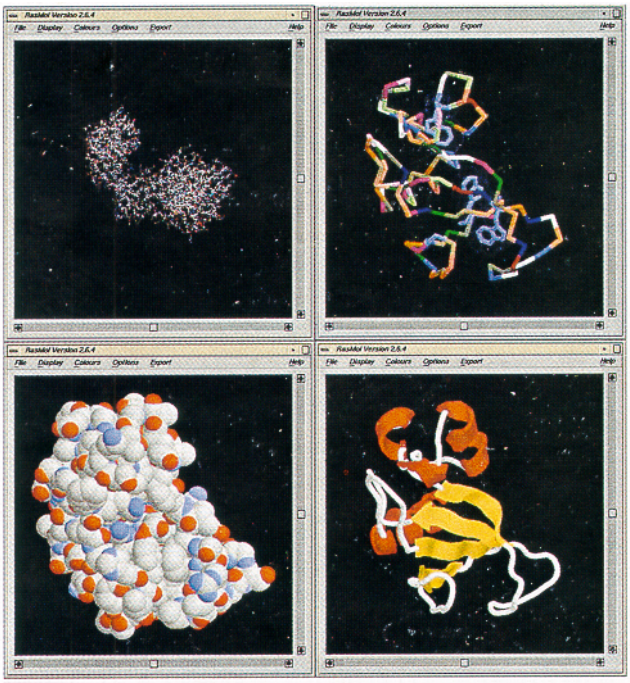

Figure 2: Complementary views of the barnase molecule [24]. Reprinted by permission of Wiley-Liss, Inc., a subsidiary of John Wiley & Sons, Inc.

configuration. It is difficult to determine the details of chemical bonding from either of these initial two views. To do so requires additional views to identify the secondary structure, i.e., the path that the polypeptide backbone of the protein follows in space; such views appear on the right side of Figure 2, showing the two most common types of secondary structure, alpha helix and beta sheet. Using these views to correlate the different aspects of protein structure, the biochemical researcher can learn how the protein functions, thereby gaining insights into its potential usage in life-saving drugs or disease therapies.

Rule of Decomposition

> *Partition complex data into multiple views to create manageable chunks and to provide insight into the interaction among different dimensions.* A single complex view can be cognitively overwhelming to a user. Multiple views can help the user to "divide and conquer," aiding memory by reducing the amount of data they need to consider at one time.

In some situations, the user benefits by viewing in isolation different aspects of a single, complex data object. For example, a large spreadsheet with many columns may contain more data than the user can easily comprehend simultaneously. In this case, the user may gain a better understanding by segmenting the data into multiple views. In the simplest case, the user may simply consider first one data set and then another. In a more complex scenario, the rule of decomposition can be used in conjunction with the rule of complementarity to give insight into the interaction among multiple dimensions, facilitating comparison tasks.

Figure 3 shows two views of a single underlying table of baseball data [12]. The top view is a trilinear plot that uses four of the five attributes in the table: player identifier, percentage of putouts, percentage of errors; and percentage of assists. The maximum values for the three percentage variables appear in the corners of the triangle. Each player is represented by a dot that is drawn towards the corner of the triangle that corresponds to the highest value attribute. The bottom view uses only the player identifier and position attributes; each bar represents a position and the length of the bar represents the number of players in that position. The views are coupled (using the many-to-one relationship between player identifier and positions) so that when the user selects one of the bars, all the players for that position are highlighted in the trilinear plot. Therefore, the user can see the different profiles for the different positions, e.g., that first basemen have a high percentage of putouts. Unlike a visualization that shows all attributes in a (probably cluttered) single view, these coupled views give the user insight through interaction.

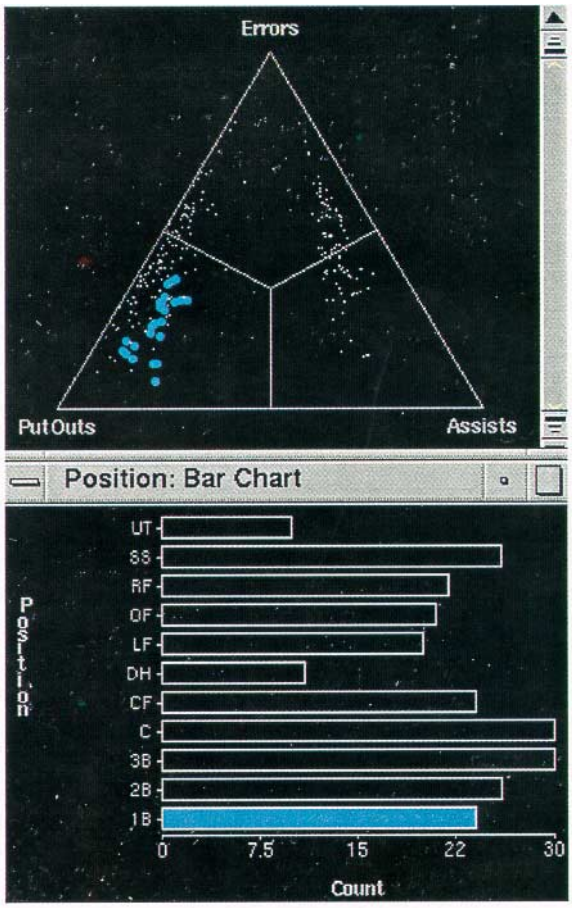

Figure 3: Two views of a single table of baseball data [12].

Rule of Parsimony

> *Use multiple views minimally.* A single view provides a user with a stable context for analysis; multiple views incur the cost of context switching. Further, multiple views introduce additional system complexity. Accordingly, the designer must be able to justify the user's learning costs and the computational and display space costs of an additional view by appealing to the rules of diversity, complementarity, or decomposition.

Additional views demand increased cognitive attention from the user, take up valuable screen space, and often require the user to learn more constructs. Therefore, additional views or additional complexity within a view should only be introduced when there is a compelling reason to do so (many such reasons are enumerated in the other rules in this section). Further, when two or more views have very similar semantics, the designer should consider merging them into one view.

As an example, one of the authors of this paper worked on a system, FotoFile, which supports the organization of digital photos and video [17]. FotoFile includes an image palette view in which users can store temporary search results and newly imported objects. It also includes a separate album editor view in which users can compose collections of objects. This design choice is in accordance with the rule of diversity, as the image palette and the album editor correspond to different genres of image collections (temporary storage versus formal composition). However, these views have very similar semantics. In fact, the image palette can be considered a temporary album. With this perspective in mind, FotoFile violates the rule of parsimony. In this example (and in many real-world examples), the two rules are in inherent conflict. Resolving these conflicts requires understanding the user's conceptual model. In this situation, the unified genre is probably easy for users to grasp. Careful consideration of the rules in this case leads to the conclusion that the rule of parsimony should have taken precedence. Thus, in retrospect, the views should have been merged.

The rule of parsimony can also be applied to the coupling of views. Coupling adds complexity to the system, both for the user and for the implementer. Therefore, when deciding whether or not to couple different views, the designer needs to consider how much value to the user such a coupling is adding.

HOW TO USE MULTIPLE VIEWS

With the issue of view selection for a multiple view system resolved, the designer must next contemplate the array of choices for view presentation and interaction. In this section, we introduce four design rules (space/time resource optimization, self-evidence, consistency, and attention management) to help designers make these decisions, as well as to help usability experts and system evaluators pinpoint trouble spots in an existing system.

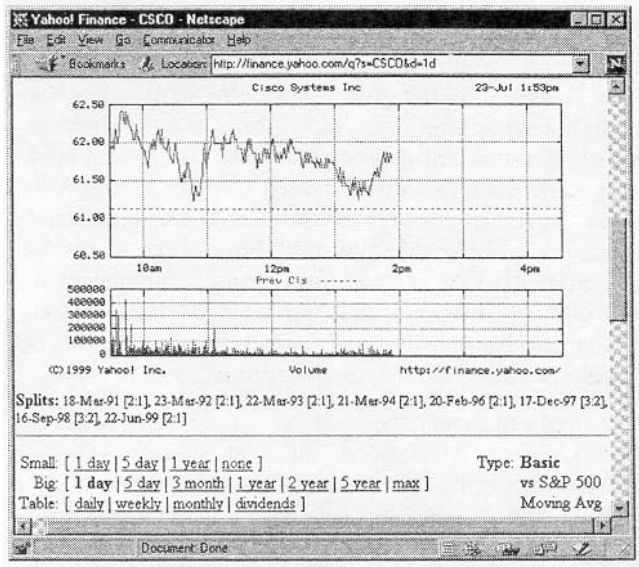

Figure 4: Multiple views of stock data, with a shared x-axis (time) to help the user easily compare the views. *Copyright 2000 Yahoo! Inc.*

Rule of Space/Time Resource Optimization

> *Balance the spatial and temporal costs of presenting multiple views with the spatial and temporal benefits of using the views.* It is easy to forget to account for the display space and computation time required to present multiple views side-by-side; likewise, it is easy to forget to account for the time saved by side-by-side views if the user's goal is to compare views.

One of the first decisions a designer must make is whether to present multiple views side-by-side or sequentially. Even if the application allows the user to make this determination, a good default is still critical. To make this decision, the designer should consider how much space and time are available to the user, as well as how much space and time each of the candidate views requires. While it is relatively straightforward to compute space costs, time costs are often trickier to compute. Hidden time costs include the time required for a user to context-switch from one view to another and the time required for a view to be computed and rendered.

Figure 4 illustrates this rule with a screen shot from Yahoo! Finance [32] that depicts stock performance. Notice that the user sees the closing price view and the volume traded view simultaneously. However, views at the 1-day, 5-day, 3-month, 1-year, 2-year, 5-year, and max time scales are shown one at a time.

Why not show all of the possible views simultaneously? First, the user would not be able to see all of the graphs at once. Though scrolling gives the user the illusion that the graphs appear spatially near each other, the user would be unlikely (on standard monitors) actually to see the graphs at the same time. Second, increased graphics on the Web means increased download time and thus increased consumption of the time resource. Both of these reasons stem from the rule of space/time resource optimization. The rule of decomposition provides a third reason: simultaneously showing all views would result in increased clutter and information overload.

Note that applying the rule of space/time resource optimization will lead to different conclusions for different platforms. For example, sequential views are likely to win over side-by-side views on small devices, e.g., a Palm Pilot.

Rule of Self-Evidence

> *Use perceptual cues to make relationships among multiple views more apparent to the user.* Static or dynamic, visual or auditory, perceptual cues can move view registration/alignment (the first step in view comparison) from the realm of cognition to the realm of perception. Additionally, users can learn systems with self-evident relationships more quickly.

Discerning the relationships among views can be a difficult task for the user of a multiple view system. The use of perceptual cues can make relationships more apparent to the end user.

There are many types of perceptual cues that can be applied. For example, SenseMaker [2], an information-exploration interface, uses highlighting to inform the user about what is new in the current view as compared to the previous view. The spatial arrangement of views is another commonly-used cue. In the Yahoo! Finance example shown in Figure 4, the closing price and volume traded views are vertically aligned so that they have a shared x-axis (time) to facilitate cross-referencing. Coupled interaction, described in the model section, is a third technique for helping the user understand the mapping from one view to another. The systems shown in Figures 1 and 3 use brushing to show the relationships among data in each of the views.

Perceptual cues can be extremely helpful to the user, but their application must be tempered by an understanding of their limitations and subtleties. We illustrate this point with two examples.

First, coupling, while powerful, introduces many complexities, and so it should be considered in the system design from the start. This is particularly critical for systems in which the user may make changes in more than one visible view or when the coupling functions are non-trivial. For example, the data in two views may be non-trivial transformations of each other, resulting in semantic questions (which objects in view A should be highlighted when the user brushes an object in view B corresponding to their mean value?). Additionally, once semantics have been decided upon, implementers must be careful to maintain the information needed to map between objects in all views. The DataSplash database visualization system [11] displayed data in a tabular view and a graphical view (derived from the data in the tabular view). Even though there was a one-to-one relationship between objects in the two views, brushing was unidirectional, which was disorienting for users. This was the result of a fundamental implementation decision that proved difficult to reverse: the derived view was the end-product of a rendering pipeline in which the information needed to map back to the tabular data was thrown away early. Finally, some data sets are too massive to allow extensional mapping, but not all intensional mappings can be inverted perfectly [31].

Second, not only is it important to employ cues that indicate the presence of relationships among views, it is also important to ensure that the visual and interactive components of the interface do not result in miscuing. The literature on visual search tasks has shown that mishighlighting affects user performance [21], and we can expect a similar deleterious effect for miscues in a multiple view system. False cues may suggest "false positives" – implied relationships that in fact do not hold among the views. The designer must also be careful with respect to "false negatives." For example, we know from the perceptual literature that if two events occur within 100 milliseconds of each other, the user perceives them as causally related [9]. Thus, changes propagated from one window to another should take place within 100 milliseconds or user may fail to recognize a relationship that exists between the views. Therefore, when quick updates are not possible due to computational limitations or other constraints, the views should be temporarily decoupled and this decoupling should be made evident to the user. For example, the system could gray out the view that has been temporarily decoupled. Alternatively, DataDesk [29] uses a small symbol to indicate that a view is out of date and allows the user to trigger an update on demand.

Rule of Consistency

> *Make the interfaces for multiple views consistent, and make the states of multiple views consistent.* The additional complexity introduced by multiple views must be balanced by ease of learning, which is facilitated by consistency.

Consistent states make comparisons easier. In contrast, view inconsistency can lead to false cognitive inferences by the user.

In addition to suggesting that designers follow the usual recommendations of consistency for interfaces [22,23], we note two specific areas in which multiple view systems should be consistent: system state and interface affordances.

System state encompasses both the data set and the user's viewpoint. For example, if one view shows a particular region, a related view should show the same region. Similarly, if objects are highlighted in one view, the corresponding objects in a related view should be highlighted as well. For some applications, this consistency may not be desirable, e.g., because the user wishes to use different views to preserve different states. Also, as observed above, this consistency may not always be possible to implement. In these cases, the rule of self-evidence dictates that decoupling be made clear.

Consistency in interface affordances makes multiple view systems easier to use and easier to learn. To apply this rule, the designer should first partition the systems' views into equivalence classes by analyzing their functionality. Next, the designer should ensure that all views in the same equivalence class have the same affordances. For example, the day, week, and month views of the Palm Pilot calendar tool belong to an equivalence class because they provide similar functions, albeit at different levels of granularity. In both the week and month views of this tool, a small icon is displayed to indicate an event. In the week view, tapping this icon brings up a tooltip describing the event. In the month view, the icon is too small to tap separately. Tapping in the region of the icon switches the system to the daily view of the day on which the event occurs. These unexpected context switches can be disconcerting to the user.

Rule of Attention Management

Use perceptual techniques to focus the user's attention on the right view at the right time. When events occur which require the user's attention, perceptual techniques can direct the user to a salient view. If the user is able to rely on the system, the user can monitor less frequently, which means they do not need to remember to context-switch to check other views.

An important challenge in a multiple view system is to ensure that the user's attention is in the right place at the right time. This requires both guiding the user to the currently important view and ensuring that the user is not distracted away from that view. Perceptual techniques—including animation, sounds, highlighting, and movement—are invaluable in designing for these goals (as well as for making relationships self-evident, as we have seen before).

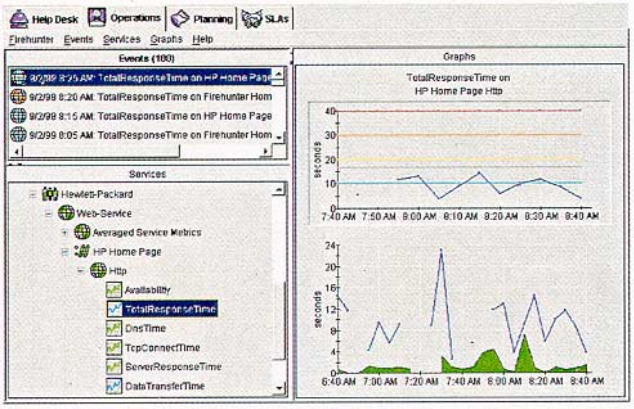

Figure 5: Multiple views of Internet service data. The view in the upper left draws the user's attention to priority events.

Figure 5 shows how the Agilent Technologies Firehunter system for Internet service management [13] successfully uses color highlighting for attention management. The event view in the upper left of the figure has a scrolling list of service events. Below it is the services view, which hierarchically organizes services. To the right are two different graph views that show service measurements (in this case, total response time for the HP and Agilent home pages). Consider a user of this system who is analyzing these graphs. If an important event takes place during this time, it is imperative that the user's attention be directed to the new event. By highlighting potentially significant events in red, the user's attention is appropriately managed.

Other perceptual considerations apply to visual updating as well. For example, consider a map-based system in which a user can manipulate a view box in a coarse-grained view. Displaying the corresponding changes in the fine-grained view can distract the user's attention if the rate of change is too fast.

CONCLUSIONS

Following in the established tradition of articulating design guidelines, we have presented a set of eight guidelines for the use of multiple views in information visualization. The first four guidelines—diversity, complementarity, parsimony, and decomposition—provide the designer with rules for the selection of multiple views. The last four guidelines—space/time resource optimization, self-evidence, consistency, and attention management—apply to the presentation and interaction design questions that arise in these systems.

These guidelines are summarized in Table 1. The first two columns list the names and brief descriptions of the guidelines. Each of the eight rules described in this paper

Rule	Summary	Major Positive Impacts on Utility	Major Negative Impacts on Utility
Diversity	*Use multiple views when there is a diversity of attributes, models, user profiles, levels of abstraction, or genres.*	memory	learning computational overhead display space overhead
Complementarity	*Use multiple views when different views bring out correlations and/or disparities.*	memory comparison context switching	learning computational overhead display space overhead
Decomposition	*Partition complex data into multiple views to create manageable chunks and to provide insight into the interaction among different dimensions.*	memory comparison	learning computational overhead display space overhead
Parsimony	*Use multiple views minimally.*	learning computational overhead display space overhead	memory comparison context switching
Space/Time Resource Optimization	*Balance the spatial and temporal costs of presenting multiple views with the spatial and temporal benefits of using the views.*	comparison computational overhead display space overhead	
Self-Evidence	*Use perceptual cues to make relationships among multiple views more apparent to the user.*	learning comparison	computational overhead
Consistency	*Make the interfaces for multiple views consistent, and make the states of multiple views consistent.*	learning comparison	computational overhead
Attention Management	*Use perceptual techniques to focus the user's attention on the right view at the right time.*	memory context switching	computational overhead

Table 1: Summary of rules and areas of major impact on utility.

can be characterized in terms of those aspects of utility they improve and those aspects of utility they impact in a negative way. This information appears in the third and fourth columns. Designers can identify aspects of utility of high importance to them and then check the table to make sure they have considered rules that particularly impact those aspects. It should be noted that this is a simplified characterization. For example, for many rules, there may be both positive and negative impacts upon a given cognitive aspect; in these cases we have mentioned the nature of only the most significant impact.

We have derived these guidelines by analyzing existing systems, drawing on our own experiences in designing such systems, and participating in discussions at a CHI '98 workshop on information exploration environments.

We expect these guidelines will continue to mature. For example, we have considered primarily systems with a single view per window. Among other areas, we would like to extend our guidelines to cover in more detail: (1) context-sensitive rendering [4,5]; (2) dynamic filtering [1,16,27]; and (3) multimodal systems, e.g., systems that integrate visual and audio representations of data.

Other topics for future consideration include a more formal model of multiple view systems and guidelines for other aspects of information visualization.

ACKNOWLEDGMENTS

We thank the organizers of the CHI '98 Workshop on Innovation and Evaluation in Information Exploration Interfaces, Nicholas J. Belkin of Rutgers University and Gene Golovchinsky of the FX Palo Alto Laboratory, Inc., for inspiring us to do this work. We thank Beth Hetzler of the Pacific Northwest National Laboratory, Rick Kopak of the University of Toronto, Allison L. Powell of the University of Virginia, and Birgit Schmidt-Wesche of the IBM Watson Research Center for providing stimulating discussions in the subgroup meetings. We would like to particularly acknowledge Laurence Nigay and Frédéric Vernier, who were active participants not only in the workshop but also in our informative post-workshop discussions. We also owe a debt of gratitude to our

management at HP Labs and Xerox PARC for supporting our work. Finally, we are grateful to Paul Aoki, Ed Chi, Beth Hetzler, and Jock Mackinlay for comments on earlier versions of this paper.

REFERENCES

1. Ahlberg, C., and Shneiderman, B. "Visual Information Seeking: Tight Coupling of Dynamic Query Filters with Starfield Displays." *Proc. ACM SIGCHI '94,* Boston, 1994, pp. 313–317.
2. Baldonado, M., and Winograd, T. "SenseMaker: An Information-Exploration Interface Supporting the Contextual Evolution of a User's Interests." *Proc. ACM SIGCHI '97,* Atlanta, 1997, pp. 11-18.
3. Becker, R.A., and Cleveland, W. S. "Brushing Scatterplots." *Technometrics*, 1987, 29(2):127-142.
4. Bederson, B.B., and Hollan, J.D. "Pad++: A Zooming Graphical Interface for Exploring Alternative Interface Physics." *Proc. ACM UIST '94*, Marina del Rey, CA, 1994, pp. 17-26.
5. Bier, E.A., Stone, M.C., Pier, K., Buxton, W., and DeRose, T. "Toolglass and Magic Lenses: The See-Through Interface." *Proc. ACM SIGGRAPH '93*, Anaheim, CA, 1993, pp. 73-80.
6. Buja, A., Cook, D., and Swayne, D. F. "Interactive High-Dimensional Data Visualization." *Journal of Computational and Graphical Statistics*, 5(1):78-99, 1996.
7. Buja, A., McDonald, J.A., Michalak, J., and Stuetzle, W. "Interactive Data Visualization Using Focusing and Linking." *Proc. IEEE Visualization '91*, San Diego, CA, 1991, pp. 156-163.
8. Card, S., Mackinlay, J., and Shneiderman, B. (eds.). *Readings in Information Visualization: Using Vision to Think.* Morgan Kaufmann Publishers, San Francisco, 1999.
9. Card, S.K., Moran, T.P., and Newell, A. *The Psychology of Human-Computer Interaction.* Lawrence Erlbaum Associates, Hillsdale, NJ, 1983.
10. Coutaz, J., Nigay, L., Salber, D., Blandford, A., May, J., and Young, R. "Four Easy Pieces for Assessing the Usability of Multimodal Interaction: The CARE Properties." *Proc. Interact '95*, Lillehammer, Norway, 1995, pp. 115-120.
11. DataSplash. http://datasplash.cs.berkeley.edu/
12. EDV: The Exploratory Data Visualizer. http://www.bell-labs.com/~gwills/EDVguide/edv.html
13. *Firehunter Service Assurance for E-Business* from Agilent Technologies. http://www.firehunter.com/
14. Golovchinsky, G., and Belkin, N.J. "Innovation and Evaluation of Information Exploration Interfaces: A CHI98 Workshop." *SIGCHI Bulletin* 31(1):22-25, 1999.
15. Hetzler, B., Whitney, P., Martucci, L., and Thomas, J. "Multi-Faceted Insight through Interoperable Visual Information Analysis Paradigms." *Proc InfoVis '98*, Research Triangle Park, NC, 1998, pp.137-144.
16. Jacobson, A. S., Berkin, A. L., and Orton, M. N. "LinkWinds: Interactive Scientific Data Analysis and Visualization." *Communications of the ACM*, 37(4):42-52, 1994.
17. Kuchinsky, A., Pering, C., Freeze, D., Creech, M., Serra, B., and Gwizdka, J. "FotoFile: A Consumer Multimedia Organization and Retrieval System." *Proc. ACM SIGCHI '99*, Pittsburgh, 1999, pp. 496-503.
18. Nielsen, J. *Usability Engineering*. AP Professional Press, Boston, 1994.
19. North, C., and Shneiderman, B. "A Taxonomy of Multiple Window Coordinations." Univ. Maryland Computer Science Dept. Technical Report #CS-TR-3854, 1997.
20. North, C., and Shneiderman, B. "Snap-Together Visualization: A User Interface for Coordinating Visualizations via Relational Schemata." *Proc. AVI 2000*, Palermo, Italy, 2000 (this proceedings).
21. Philipsen, G. "Effects of Six Different Highlighting Modes on Visual Search Performance in Menu Options." *International Journal of Human-Computer Interaction* 6(3):319-334, 1994.
22. Polson, P. "The Consequences of Consistent and Inconsistent User Interfaces." In *Cognitive Science and its Applications for Human-Computer Interaction* (ed. R. Guindon), Lawrence Erlbaum Associates, Hillsdale, NJ, 1988.
23. Rubinstein, R. and Hersh, H. *The Human Factor: Designing Computer Systems for People.* Digital Press, Bedford, MA, 1984.
24. Sayle, R. and Milner-White, E. J. "RasMol: Biomolecular Graphics for All." *Trends in Biochemical Sciences*, 20:374-376, 1995.
25. Sclabassi, R. J. "Computational Support for the Development of Medical Multimedia Applications." Center for Clinical Neurophysiology, University of Pittsburgh Medical Center, 1993.
26. Shneiderman, B. *Designing the User Interface: Strategies for Effective Human-Computer Interaction.* Addison-Wesley, Reading, MA, 1987.
27. Spotfire. http://www.ivee.com/
28. Tweedie, L.A., Spence, R., Dawkes, H., and Su, H. "Externalising Abstract Mathematical Models." *Proc. ACM SIGCHI '94*, Vancouver, BC, 1994, pp. 406-12.
29. Velleman, P. *The DataDesk Handbook.* Odesta Corporation, Ithaca, NY, 1988.
30. Visage. http://www.maya.com/visage/
31. Woodruff, A. and Stonebraker, M. "Supporting Fine-Grained Data Lineage in a Database Visualization Environment." *Proc. 13th ICDE*, Birmingham, England, 1997, pp. 91-102.
32. Yahoo! http://www.yahoo.com/