

Interactions for Addressing Shortcomings Encountered when Physically Creating and Manipulating Large Affinity Diagrams

Benjamin Koh, Ray Su
Human Computer Interaction Institute, School of Computer Science
Carnegie Mellon University
Pittsburgh, Pennsylvania 15217
{benkoh@cmu.edu, rjsu@andrew.cmu.edu}

ABSTRACT

In this paper, we investigate problems encountered when physically manipulating large affinity diagrams. This task is not unlike the common task of creation, visualization and manipulation of significantly large amounts of data. Current methods of addressing this particular task involve using physical tools, bending existing digital software tools, or creating new software tools. These existing solutions have their advantages, but each has inherent shortcomings that still leave this task a difficult one to accomplish. We will address the task of Affinity Diagramming by exploring software interaction techniques. We propose a software application, equipped with these interaction techniques, for overcoming these issues.

Keywords

Information visualization, affinity diagrams, interaction techniques, zoomable user interfaces (ZUI),

INTRODUCTION

Affinity diagrams are an essential component of contextual design, allowing designers to view high level abstractions and general trends in large amounts of data. As such, one would expect the availability of specialized tools for creating these diagrams. Unfortunately, there are not many tools in existence expressly for this purpose, and those that exist fall short of providing an adequate environment for authoring and visualizing these documents.

Affinity diagrams are typically created from words and phrases written on sticky notes and grouped together on a large surface. This presents several problems, especially when dealing with large quantities of data. While there are no explicit limitations on the number of notes contained within an Affinity diagram, affinities in the order of 200+ notes are typical (Beyer/Holtzblatt).

Issues that arise when dealing with large Affinity diagrams include:

1. Finding an area large enough to contain all the notes
2. Manipulating notes
3. Visualizing a large quantity of notes efficiently

4. Storage and Transportation of notes
5. Classifying and Searching for notes
6. Sharing and accomplishing Collaborative work on the diagram with other team members/stakeholders

In the pages that follow, we present our perspective on how new interactions could be introduced to make the process of creating and managing large affinity diagrams more efficient. We will also discuss our experience creating a software application that incorporates these interaction techniques and other methods for generating ideas from existing affinities.



Figure 1: Example of Traditional Affinity Diagramming

ANALYSIS OF EXISTING TOOLS AND PROCESSES

Traditional Tools

As discussed in the introduction, the traditional method of building affinity diagrams is for individuals to write phrases on notes and post them on a large surface, usually a bulletin board of sorts. Then, as a group, individuals will attempt to group individual notes with other notes that have similar meaning. As new affinities/groups are created, they need to be labeled and visually separated into distinct groups.

Pros:

This tool is particularly effective in providing the capacity to visualize a large quantity of notes. The large board offers a persistent quality that integrates into the user's

workspace, allowing the users to be immersed in the ideas/concepts of the Affinity diagram. Team members in the same work area can also collaborate on ideas fairly easily.

Cons:

The advantages that this tool brings to the user have inherent tradeoffs. Being able to visualize the large amount of notes requires a large enough board that can be dedicated to the diagram for the duration of the process and/or project. Also, storage and transportation of the diagram is very difficult. Saving the state of diagram usually entails taking a photograph (which can be difficult to read) or physically transferring each note and/or group of notes onto a paper medium. This is usually a time consuming task. Finally, note manipulation can be tedious as each post-it note needs to be individually moved if groups need to be rearranged or space needs to be cleared for new notes.

Generic Tools

Generic tools exist for creating notes and relating them to each other. Microsoft Visio (PC) and OmniGraffle (MacOS) are popular applications for depicting hierarchical and flow diagrams. While these tools can be applied to creating Affinity diagrams, they were not designed for this purpose and have many shortcomings that make them difficult to use.

Pros:

Generic tools are readily available. Most intermediate computer users should be aware of their existence and be familiar with how to use them to complete simple tasks. The use of such software applications yields clear benefits, in that the user has the ability to save and restore sessions, facilitating transport and storage of their diagrams.

Cons:

Unfortunately, because generic diagramming applications must cater to the needs of many, they fail to fully support users who wish to create a specific kind of diagram. In the case of affinity diagrams, users may find themselves spending many hours tweaking low level details, such as bounding boxes and layout within groups.

Specialized Tools

Specialized tools such as Pathmaker and CD (Contextual Design) Tools contain modules that address the creation of affinities. However, these tools also attempt to cover the entire process of Contextual Design, and in doing so fail to address specific user needs in the affinity creation process.

Pros:

Similarly to generic tools, specialized software for creating affinity diagrams allows users to save and restore their work. As these software packages typically attempt to address the entire Contextual Design process, the use of these tools can be beneficial for integrating the affinity

diagram with the rest of the contextual design methods and for tracking data which can appear in multiple models.

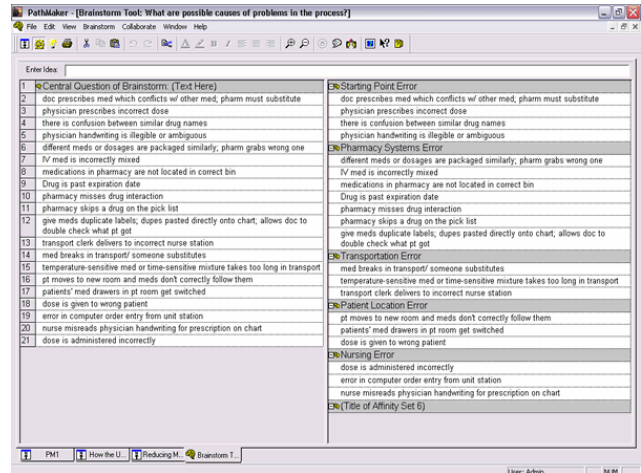


Figure 2: Pathmaker: Affinity Diagramming Tool

Cons:

The downside to using specialized tools, however, is that they are quite complex to learn and use as they consist of many modules and components. They can be restrictive, in that they typically do not easily support the creation of a simple diagram, but require the user to set up a project and to follow a specific process. This is analogous to software development environments that require the user to start a "project" in order to edit a single file. Additionally, because tools such as Pathmaker and CD Tools cater to the entire Contextual Design process, they fail to address some of the specific challenges of creating of affinity diagrams, such as data visualization and manipulation.

Research

Research projects such as the Designer's Outpost (UC Berkeley) touch on certain aspects of information grouping and relationships, but do not address the affinity creation process as a whole.

Pros:

The use of a research tool such as Designer's Outpost yields many benefits. The tool combines the affordances of paper with the advantages of electronic media to support collaborative information design. Users can physically interact with the board, posting notes that the board recognizes and can represent digitally even after the note is removed. Users can then move the digital notes by direct haptic manipulation, draw links between notes, and save the contents of the board.

Cons:

Although it addresses certain manipulation challenges of affinity diagram authoring, Designer's Outpost does not handle very well the issues of space and data visualization. The Designer's Outpost, as it currently exists, is the size of a large whiteboard. It can therefore only support the display

of a limited amount of information at a single time. The Designer's Outpost also doesn't appear to address issues of layout when organizing and reorganizing notes into groups.

SHORTCOMINGS OF PHYSICAL DATA MANIPULATION

Size of Workspace

Finding a workspace of a suitable size is an obvious problem when working with large data sets. In the particular case of affinity diagrams, this space must be large enough to hold all the individual notes. This is problematic since such surfaces are usually difficult to find. One must, in addition, correctly estimate how much space is needed in order to prevent a situation that would require transferring the diagram to another surface when the current one is outgrown.

Our Solution and Design Rationale:

We chose to design our solution in the Piccolo framework (<http://www.cs.umd.edu/hcil/piccolo/>). This framework provides a Zoomable User Interface (ZUI) which allows for a virtual work area of infinite space. In representing the affinity diagram within this virtual workspace, we can leverage this infinitely large area to post and organize notes. This solution enables the user to overcome space issues without fear of outgrowing the workspace. However, offering an infinitely large virtual work area naturally presents problems of visualization, navigation, manipulation and searching of notes. While these issues are common problems with building affinity diagrams (or any task involving large amounts of data), they become more so when opening the workspace with no upper bounds. The following sections will attempt to address these issues.

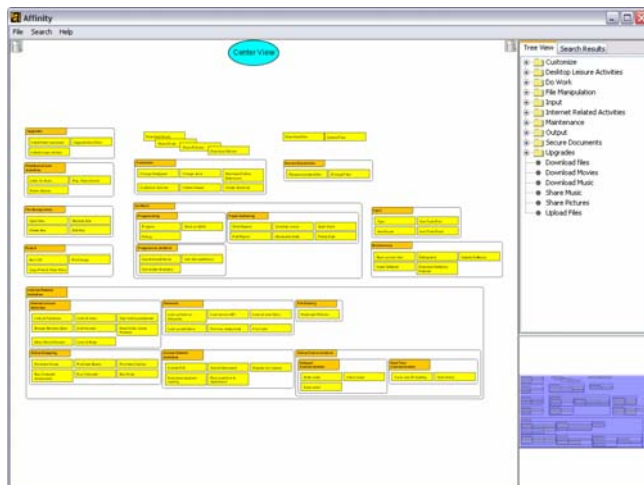


Figure 3: Affinity: Our Affinity Diagramming Solution

Manipulation of Notes

The main activities in constructing an affinity diagram are (1) posting individual notes to the workspace and (2) grouping notes and groups with other notes and groups of similar meaning. This becomes very impractical using

regular sticky notes. To begin, the adhesive properties of a sticky note diminish every time it is repositioned. Reorganizing groups is also a hassle, as each individual note has to be relocated. Borders drawn around old groups must be erased and redrawn to denote the new formation. Finally, repositioning an entire group may require making space for it on the workspace, which could involve recursively repositioning other groups. Furthermore, editing of notes need to be accomplished by making changes on the post-it note itself, or creating a completely new post-it note. Both of these tasks can be awkward and can present potential repetitive stress injuries if done frequently enough.

Our Solution and Design Rationale:

Our implementation of the virtual workspace offers direct manipulation of the notes as if they were post-it notes with “sticky” qualities. Each note can be picked up and dragged to a new location. There is no limitation to where the note can be placed; notes can be overlapped, or grouped by proximity. This level of interaction provides the user with the freedom to represent the data as he/she pleases. While grouping by proximity (the usual ad-hoc method employed by the traditional physical method of affinity diagramming) can still be used, we also provide the ability to group the notes within a higher level “group note”. These “group notes” can be labeled as any other note, but have the special property that any other note or group note can be dragged into it to form a semantic grouping. The effect of this is that n-level hierarchy of grouping can be created.

Within each group note, the sub notes are automatically laid out using a hybrid of flow and grid layout. This hybrid allows for effective packing of the sub notes to use minimal amounts of space while still maintaining organization, legibility and ease of serial search. Dropping a new note into the group note will automatically place the note into the end of the flow layout.

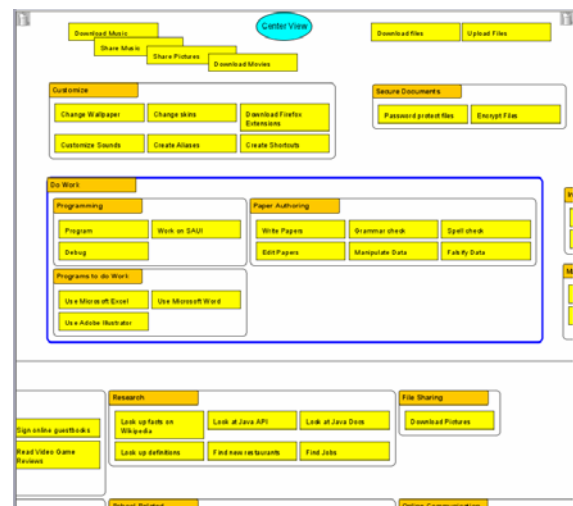


Figure 4: Flow+Grid Hybrid Layout and Grouping of Notes

Additionally, our solution also automatically handles recursive layout of groups. When a note/group note is added/removed from another group note, the borders of the group note will dynamically “grow” to the bounds of the containing sub notes. This automatic layout manager should eliminate the work of having to resize/redraw new group borders to accommodate the dynamic qualities of the affinity diagram. This ease of interaction should greatly reduce the burden upon the designer to manually manipulate groupings of notes and focus upon the content of the affinity diagram instead of the appearance.

Visualization of Notes

One of the problems with working with large amounts of data that need to be visualized spatially, is providing a way to see the data at granular levels of detail as well as higher levels of detail. The traditional method of physical affinity diagramming accomplishes this fairly well assuming a large enough workspace is available. Moving into the digital domain, we are constrained by the display sizes of current computing systems. These systems currently have displays that inherently limit the amount of data that can be viewed at one time.

Furthermore, when an affinity diagram contains a large number of notes, it can be useful to provide more than one way of visualizing the diagram. While a spatial organization of information can be useful, certain situations call for a hierarchical search through the affinity diagram.

Our Solution and Design Rationale:

The zoomable user interface provided by the Piccolo frame work addresses the issue of data visualization at different levels of detail. We provide the user the ability to navigate into meaningful levels of zoom. Instead of providing continuous zooming levels as is the default provided by Piccolo, we allow the user to selectively choose which notes or groups of notes to zoom into. Using mouse and keyboard combinations, the user can specify to zoom to fit a note, a group of notes or even drag out a selection area to zoom into in a very fluid fashion. (See figure 3 for zooming at the highest level. See figure 4 for zooming at a group node level: notice the group node border is highlighted to indicate focus of zooming.) This fluid interaction gives the user a constant awareness of where in the diagram they are and provides sufficient control for any level of visualization and editing.

Additionally, a “bird’s-eye” radar view is also provided so that no matter the level of zoom the main work area is in, the user can still get a high level perspective of the entire affinity diagram. This radar view is also fully interactive; the user can not only see where the current view of the workspace is in relation to the entire diagram, but drag this small perspective window to change and update the workspace view.

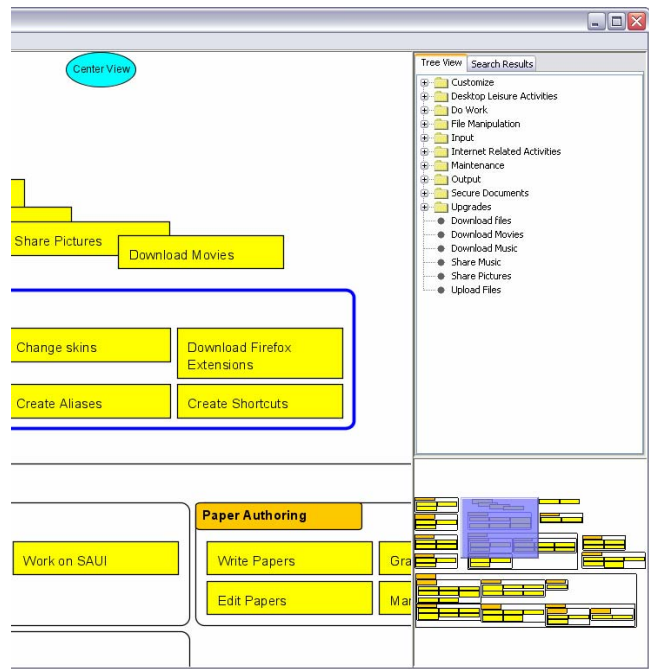


Figure 5: Bird's Eye Radar View (bottom right corner) and Tree View Navigation-Zoom (top right corner)

We provide both spatial and hierarchical representations of the affinity diagram. The main work area displays the spatial representation of the affinity diagram, while a tree view also caters to those who wish to view the diagram hierarchically.

In deciding how to display a large quantity of notes onscreen, we chose to implement a ZUI, or Zoomable User Interface for the main work area. This enables us to display the high level structure of the affinity diagram, while allowing us to show detail as needed.

Storage and Transportation of Notes

Once an affinity diagram has been created, the problem then becomes how to share the diagram with others. A few methods are regularly employed: (1) digitizing the diagram using a drawing tool or other general diagramming software (ex. Visio), (2) taking a digital photograph of the diagram, and (3) transferring the notes to poster paper. The first method can be extremely time-consuming and frustrating if the user must manually draw low level details of the diagram (lines, bounding boxes). The second method requires a high resolution digital camera. However, even when taken at a high resolution, legibility of the notes could still be an issue because of lighting conditions, illegible handwriting, and poor contrast between the text and background color. Transferring the diagram to poster paper is the most direct means of sharing the affinity with others, but it is also the most inconvenient of the three methods. Ensuring that notes remain stuck to the poster paper during transportation can prove to be a demanding task, especially when the diagram is rolled up to facilitate storage.

Our Solution and Design Rationale:

Allowing the user to share his or her affinity diagrams is critical, as the diagrams are used as communication tools for creating a shared language and understanding of data. Our implementation of an Affinity Diagram authoring environment allows the user to both save and restore diagrams. It also provides an export to JPEG function, which enables the diagram to be viewed on any software application capable of opening standard image files.

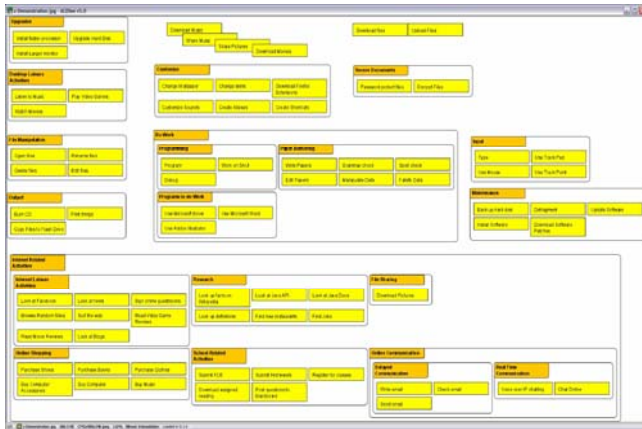


Figure 6: Addressing persistence by export to JPEG. Notice the exported image file is identical to the application workspace view.

Taking it a step further, the application also allows the ability to save the state of any affinity diagram into a file. This file can be transferred and shared between team members, allowing team members to work on the same data set in remote locations. This remote collaboration is a potential area that can be improved upon and extended so that the application can communicate across networks so that remote users can simultaneously work on the same diagram. This will be discussed in the Future Direction section.

Searching for Notes

The ability to locate notes becomes increasingly difficult as the amount of data represented in the affinity diagram increases. However, when designing from data, this activity is of critical importance to link design ideas to data. Locating related notes is also important when classifying a new note or group. Knowing which notes in the diagram are related to the new note helps to avoid duplication and aids in classification.

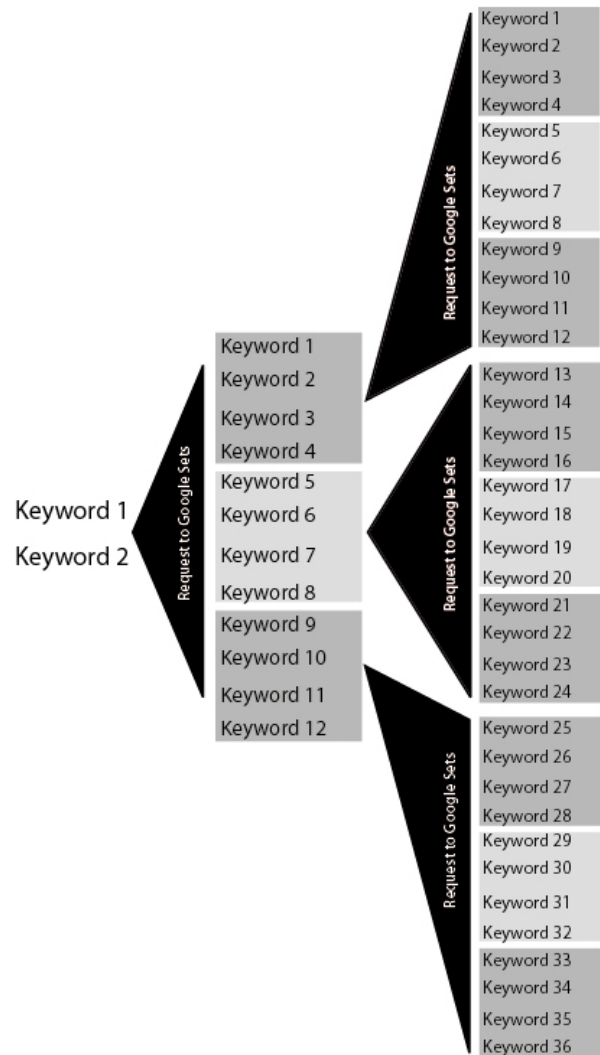


Figure 7: Recursive set generation using Google Sets

Our Solution and Design Rationale:

Our prototype of an augmented affinity diagram authoring environment provides two ways of searching for notes and groups. The first is by simple text matching. We also provide an “Extended” search function that searches for notes containing related words to the original search query. To do this, we leverage Google Sets to generate broadened search criteria. We employ a recursive set generating algorithm for expanding upon the results returned by Google Sets, as the maximum number of results from Google Sets generally does not exceed 15 (see figure 7). Results from a single request are separated into “chunks” and then resubmitted to Google Sets to get around this limitation.

ADDITIONAL ENHANCEMENTS AND CONSIDERATIONS

Generation of Ideas from Established Affinities

While one of the goals for this tool was to explore interaction techniques to improve the way affinity diagrams are created and manipulated, we also considered ways to

improve the generation of ideas during affinity diagramming. The effectiveness of an affinity diagram not only depends on the way ideas are grouped together, but also on the quality of data generated. To help improve on the quality of data, we leveraged the resources of the internet (Google Sets specifically) to generate more ideas. Specifically, we enable the user to enter keywords into a search feature. These key words can be existing words in the affinity diagram. The search will then present the user with additional words that are related to the initial words and concepts. The intent of this is to help spur idea generation during periods of “creative drought” and also to expand the diagram into broader areas of exploration.

Fitt’s Law Trash Cans

Deletion of notes is an important task that we wanted to make as effortless as removing a post-it note from a white board. We designed two trash can icons in the upper left and right corners of the workspace. The user is able to drag and drop (using the same drag and drop interaction of note placement and note grouping) notes and groups of notes into these trash can icons and delete them.

Although the trash can is small, we get around Fitt’s Law by forcing the cursor to be constrained within the workspace area whenever any note/group note is being dragged. The user can now drag a note a quickly by flicking the cursor upwards in any diagonal corner. This type of interaction mirrors the act of lifting a post-it note up from a white board. Note deletion can possibly be a repetitive task, and this solution will help reduce the effort required to do so.

FUTURE DIRECTIONS

Peer-to-Peer / Online Collaboration

As mentioned earlier in the discussion on storage and transportation, it would be extremely beneficial to enable the ability to collaborate the affinity diagramming process with other remote users of the tool. The process of affinity diagramming is usually accomplished with a multidisciplinary project team. And these team members may not be located in the same location, city, state or even country. Thus providing this capability will help adapt this tool to how actual design projects are accomplished.

The main idea is to make the interaction of the tool so that users can simultaneously manipulate the same state in remote locations. There are many ways this collaboration can be accomplished. The tool can be made to interact peer-to-peer, client-server or even deployed online. The exact implementation details have yet to be determined and can be explored further in the next phase of development.

Brainstorming by Pictorial Association

“A picture is worth 1000 words”. This idea is similar to the concept of Generation of ideas from Established Affinities. We can take this idea further by not only showing additional related key words, but also images. Images can

not only hold more than a singular idea, but multiple interpretations that are user dependant. This idea can also leverage resources from the internet by querying Google Images. Images can be shown automatically based on what notes are already in the affinity diagram. Whether a singular image or a collage of images should be displayed has yet to be determined and could use more thought and consideration in future development.

Semantic Display of High Level Information

When an affinity diagram has grown to a size with more than 50 notes, it can be difficult to view the diagram at a high level view with any significant understanding. At this view, the labels can be too small to be legible. Thus we’ve looked into a solution where at low levels of zoom, Groups at higher hierarchy levels should change their appearance to be more visible and legible. A working prototype version follows in the Figure below.

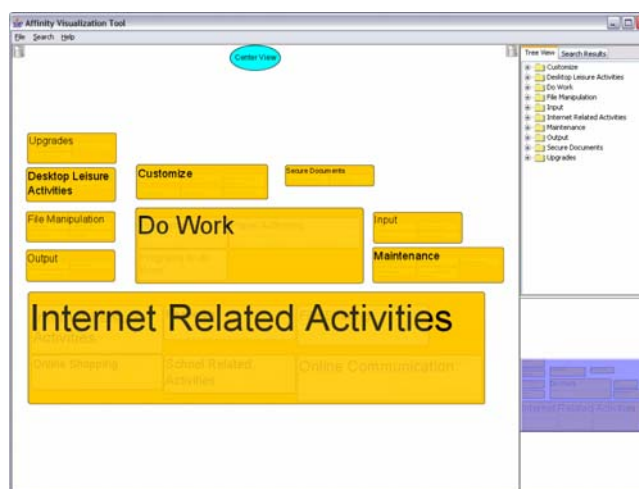


Figure 8: Semantic Display of High Level Group Notes. Note: the diagram is the same diagram as Figure 1.

Notice in the figure, that the high level groups are now legible at a glance. If a high detail of granularity is desired, the user can simply ask to zoom into a particular group note and the original group view will show all of the groups sub notes. This feature has been implemented but hasn’t been thoroughly tested, hasn’t been included in the latest release and could use further development.

User Studies

Many of the features and design decision were based on the experiences of the authors in creating affinity diagrams in the Introduction to Human Computer Interaction Methods course at CMU. Since the “user is not like me”, the application could benefit from user studies. Usability Evaluation methods such as Think-Aloud, Cognitive-Walkthroughs and even Keystroke Level modeling can be used to improve on the usability of the current application. To evaluate the necessity of certain features, contextual inquiries can be performed on designers and teams who create affinity diagrams.

LESSONS LEARNED

We take away several important points from our implementation of a zooming affinity diagramming application. Most importantly, we discovered that while ZUIs are effective at providing detailed, as well as overarching, views of data, they are difficult to design in respect to usability principles. The ability to zoom in and out at will is generally not needed in most circumstances. We found the need to restrict the user's zooming ability in order to prevent the user from zooming out to infinite, shrinking the affinity diagram to a dot and preventing the user from returning the application to its original state. We also implemented mechanisms for resetting the application's view, so as to provide the user with an escape route in case of disorientation.

Preventing disorientation is another challenge when implementing ZUIs. Navigating/Zooming between various parts of a large affinity diagram, if done too quickly, can leave the user in a state of disorientation. To overcome this problem, we were forced to slow down the zooming and to insert pauses between sequences of animations, allowing users to orient themselves briefly before resuming the animation.

Creating an effective layout manager was particularly challenging when displaying notes of varying sizes. Current layout managers have advantages and disadvantages and trying to implement one of those for displaying notes within group notes did not effectively display the notes efficiently and legibly. We designed our tool with a hybrid of two existing layout types to help address each layout type's weakness with the strength of the other. This mutually beneficial coexistence helped to create a better layout manager.

While ZUIs are difficult to design, they are also difficult to implement. Each object in a ZUI may contain its own coordinate system, which itself can be affected by the transforms being applied to the object and its parents. These transforms are updated dynamically as users zoom in and out of the canvas.

Leveraging a ZUI toolkit, such as Piccolo, is critical as it handles most of the underlying mathematical details of transforms and GUI damage and repaint. While Piccolo greatly facilitated our implementation, it was still necessary to understand the basic operations and coordinate systems at work for purposes of hit detection and graphic display.

CONCLUSION

We believe the application we have created includes many interactions that greatly simplify the process of creating and manipulating affinity diagrams. Although ZUIs are not the easiest kind of interface to design and implement, we believe they should be further explored as a means of representing large amounts of data, within affinity diagrams and other data structures.

ACKNOWLEDGMENTS

Jason Hong for guidance in the definition and scoping of this project. Jesse Grosjean for his help and prompt response to Piccolo related questions.

REFERENCES

(Beyer/Holtzblatt) Contextual Design: Defining Customer-Centered Systems