Discovering Structure of Data to Create Multiple Perspective Visualization

by

Yao Li

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2004

Author .....
Department of Electrical Engineering and Computer Science
May 7, 2004

Certified by .....
Howard E. Shrobe
Principal Research Scientist
Thesis Supervisor

Certified by .....
Patrick H. Winston
Professor of Artificial Intelligence and Computer Science
Thesis Supervisor

Accepted by .....
Arthur C. Smith
Chairman, Department Committee on Graduate Students

Discovering Structure of Data to Create Multiple Perspective Visualization

by

Yao Li

Submitted to the Department of Electrical Engineering and Computer Science
on May 7, 2004, in partial fulfillment of the
requirements for the degree of
Master of Engineering

Abstract

The goal of visualization is to help human understand, reason, and learn better and quicker. In pursuit of this goal, this thesis presents an architecture for intelligent visualization systems, which supports multiple perspectives in a parallel structure and offers high flexibility in the addition, deletion, and modification of different visualizations. It validates its design through a prototype containing a number of innovative arrangements and options, which effectively improved trial users' understanding.

Thesis Supervisor: Howard E. Shrobe
Title: Principal Research Scientist

Thesis Supervisor: Patrick H. Winston
Title: Professor of Artificial Intelligence and Computer Science

Acknowledgments

I would have found this thesis work impossible to conduct without the help of my advisors, many friends, colleagues, and my family.

I am eternally indebted to my thesis advisors, Howie Shrobe and Patrick Winston, for their guidance, inspiration, and support, and to my supervisor, Dr. Paul Keel, in particular, for generously sharing his knowledge and for always being there to rescue me out of crisis.

Among my colleagues in AIRE and EWall, I am indebted to Stephen Peters for his patience and for always showing me extra kindess; to Max Van Kleek for his friendship and for debugging my code at 3am, to Harold Fox for his unique insights and sense of humor. I also thank everyone in lab for making coming in something to look forward to everyday.

I have been lucky to have had the chance to collaborate with Alvise Simondetti and Duncan Wilson from ARUP. I thank them for constant feedback on my work.

I would also like to express my sincerest gratitude to Dr. Robert Laddaga, Professor Randall Davis, and especially to Aaron Adler and Dr. Paul Robertson for painstakingly going through my thesis drafts and providing invaluable comments.

Last, but not least, I am forever grateful to my parents for their love and for putting up with me, my little brother Zhenye Xak Mei and my adorable Kelly Clancy for their indulgence and going out of their way to help me out and keep my sanity intact, my beloved Kazutaka Takahashi for his patience, support and care.

Contents

List of Figures

Chapter 1

Introduction

The goal of visualization is to help human understand, reason, and learn better and quicker. In pursuit of this goal, this thesis presents an architecture for intelligent visualization systems, which supports multiple perspectives in a parallel structure structure and offers high flexibility in the addition, deletion, and modification of different visualizations. It validates its design through a prototype containing a number of innovative arrangements and options, which effectively improved trial users' understanding.

1.1 The Significance of Visualization

"A picture is worth a thousand words." Before character systems were established in civilization, human records consisted solely of pictures. Today, we still rely heavily on visualization to gain better and quicker understandings, by presenting abstract concepts in visual forms or geometric metaphors. For example, software diagrams

and data structures are used in learning abstract software concepts; growth rate, heartbeats, and brain waves are also frequently plotted to help detect abnormal patterns. Human reasoning is often visual in nature: hierarchical family structures are named “trees” and social groups “circles”. As Patrick Winston stated in [19], “Vision makes it possible to solve problems that would otherwise be difficult or impossible.”

Ever since 6200 BC [8], when one of the oldest known maps was made, visualization has been used for navigation and exploration. The need for statistical graphs was initiated during development of precise physical quantity measurements in the 16th century, marking the beginning of advancement in visualization. During the following century, new graphical forms were invented such as topographic maps. Development in statistical theory in the 18th century further fueled the rapid growth of visualization, since large quantities of data could then be analyzed. As statistics in social, economic and other fields were gathered, and official state statistical offices were established throughout Europe, the importance in the study of visualization was publicly recognized. Towards mid-19th century, visualization was used to provide new insights in other fields, such as physics and biology. Finally, innovations in graphical approaches and evolution in computers lead to a bloom in new visualization techniques.

In today’s world, visualization has become increasingly significant as the quantity and complexity of data grow with the rapid interactions between geographical regions, between commercial industries, and between academic disciplines. Good visualization not only reduces the difficulty caused by language and expertise dif-

ference but also makes huge quantities of data manageable.

1.2 Goal

This thesis work aims to provide an architecture for intelligent visualization systems, and to help users better understand data. It presents simple perspectives with different foci on individual aspects of the same processed information. In a wider context, it will serve as a platform for future user studies in human visual processing, leading to a better understanding of human intelligence.

The specific goal of this project was to design, implement, and test the HAPPIE system [1], which is highly adaptable to a wide variety of inputs and allows easy addition, deletion, and modification of various visualizations.

[1] Hetro-Aesthetic Parallel-Perspective Intelligent Eye, a system intended to help users "see" better. It is named after my windows box, happy.csail.mit.edu, who got increasingly depressed during the course of thesis work, but will hopefully recover shortly hereafter.

1.2.1 Problem

This thesis work began with an extensive survey and analysis of both commercial visualization applications and academic research projects (see Chapter 2). Three main shortcomings were identified as common among them.

First, all information properties are packed into one intricate visualization through multiple mappings with copious visual dimensions, as seen in [4] and [15]. To locate a specific property, users must find out the correct mapping in the visualization. As a result, they are burdened with an investigation process, with a duration proportional to the complexity of the visualization. During this process, users have to consider all visual dimensions in use, sometimes more than twenty of them, as illustrated in Figure 1-1,

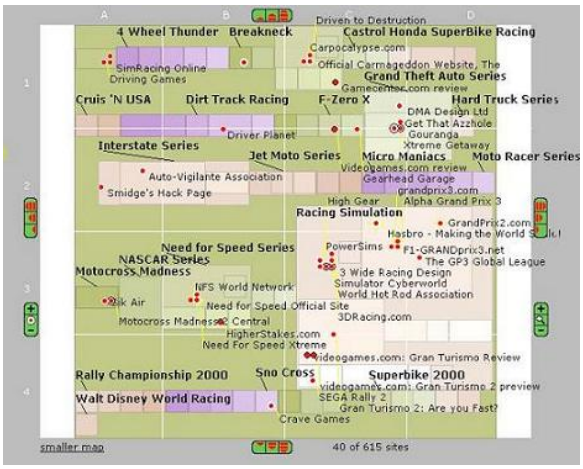


Figure 1-1: A visualization by the Map.Net program [15] using symbols, and small, medium, and large objects to represent various levels of websites. Color, size, border thickness, text stroke, and other visual dimensions, with a total of more than twenty, are used to encode various properties.

before they can locate the information of interest. For example, in the image, the number of pages found at each website is represented by the size of the red dot inside the symbol for the site. Unfortunately, before users discover this association,

they must explore the meaning encoded in the color, shape, and size, as well as many other visual dimensions associated with screen objects, in addition to those associated with links between objects, background, text, and all other visual components.

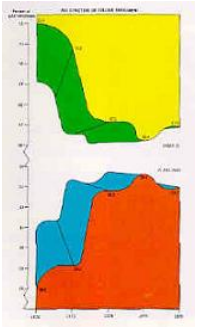


Figure 1-2: A plot of five data points [17] using all 3D spatial dimensions, color, text, and mirror image of the entire plot. [2]

Second, the number of visual dimensions used exceeds what was necessary. Users' navigation process, as described in the first problem, is further aggravated by the unnecessary multiplicity of visual encoding of content information, as illustrated in Figure 1-2. Clearly, a visualization with twenty visual dimensions, using four dimensions for each information property, takes much more labor to decipher than a visualization with five visual dimensions, in which each represents an information property. Because users have to learn a long list of mappings when a few would have been sufficient, this drawback not only wastes time and energy but also leads to potential confusion.

Third, the amount of information presented to users is overwhelming. For example, Figure 1-3shows over one million data points packed into a single visualization. To accommodate such vast data sets and to allow users to view a specific data point in detail, extensive zooming features are required. One approach was to put the zoomed views on the same page as seen in [4] and [16]. This layout not only costs space and brings distraction, but also reduces the possible size of the main

[2]Edward Tufte commented, "This may well be the worst graphic ever to find its way into print." [17].

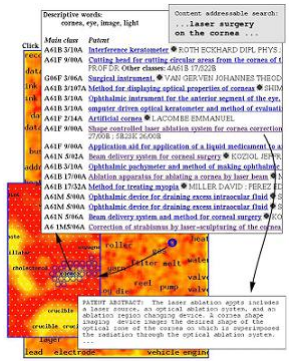


Figure 1-3: A sample exploration of a visualization by [18] for up to 7 million articles.

visualization, therefore reducing its potential content volume and complexity. Another approach was to allow users to zoom in and out on the same visualization, as seen in [2] and [14]. With a data set of high magnitude, it takes a few magnifications to reach an actual object representing a data point, identical to the common map navigation process. However, in data navigation processes, users are unfamiliar with the structure of the output. As a result, after several magnifications, it is no longer trivial to recollect and grasp where the current detailed branch on the screen fits in the big picture of the overall visualization, causing a decrease in users' understanding of the conceptual structures.

1.2.2 Guiding Principles

The direction of this thesis work is shaped by three principles, in order to improve upon the three weaknesses described in the previous section.

Simplicity: Less is more.

Parallel Perspectives: Plots, tables, figures, and many other visualization forms are customarily used to exhibit the same idea and aid the understanding process. Because each representation focuses on a separate aspect and characteristic of the same content, the only way to take advantage of all visualizations and have different visualizations complement each other is to provide all of them in a parallel structure

for the user. Each occupies the full screen one at a time, to allow maximum capacity.

Intelligence: Regardless of the complexity involved in visualization, a system cannot be considered intelligent if it always presents everything it is given and everything it knows to users. To be able to assist users, a system must first understand its own content, and then present partially interpreted information, reducing users' work load. Finally, the system can obtain feedback from users to gain additional knowledge regarding users' preferred ways of thinking. The system then improves, using the feedback, by itself or with the developer's help.

1.2.3 Implications

This thesis work is a step towards the larger goal of understanding human intelligence. As Winston stresses in [19], "It is hard to imagine how we can understand how the brain thinks unless we understand how it sees." Like Winston, many AI scientists believe that human intelligence lies partially within the vision channel.

This thesis work will contribute towards the goal of building a real AI system, one that substantially understands human intelligence and is capable of imitating the human thought process. An "imperfect" AI system, such as HAPPIE, with limited knowledge of human intelligence, can aid human users in some parts of the thought process. Specifically, HAPPIE, with an understanding of visual processing, can provide tangible assistance in three ways: 1) by participating in recognition and digestion of relevant data; 2) by facilitating the interpretation of data and its association with existing knowledge; and 3) by exposing users to alternative perspectives

and unrecognized insights. Finally, the results from user studies, conducted at the end of the thesis work, may shed light on human vision processing and enrich our understanding of human intelligence.

1.3 Examples

The following are examples of some of the implemented arrangements and options.

1.3.1 Matrix Arrangement

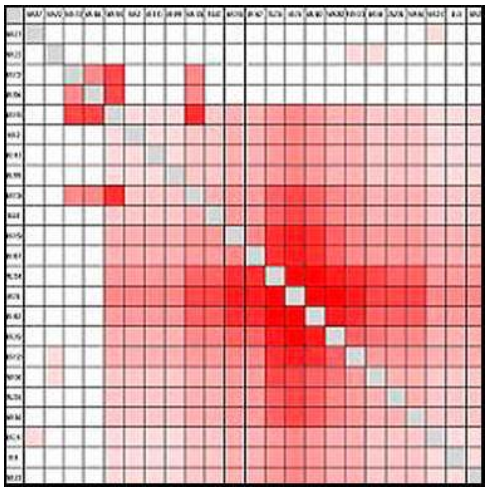


Figure 1-4: Matrix Arrangement screen shot.

The Matrix Arrangement conducts link analysis by presenting each link as a grid and sorting the indices, which represent information objects, to cluster the most important links around the diagonal.

1.3.2 Landscape Arrangement

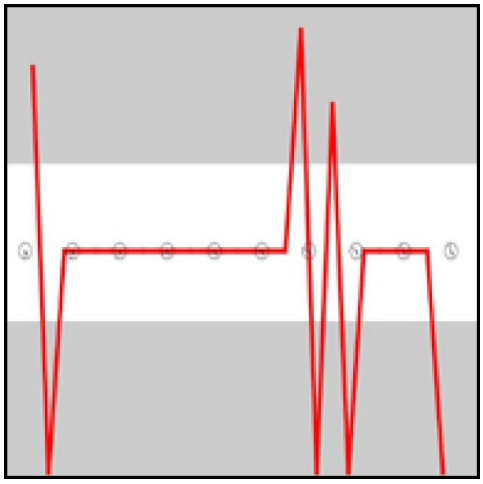


Figure 1-5: Landscape Arrangement screen shot.

The Landscape Arrangement presents a development trend by graphing the relationship between two information properties.

1.3.3 Highway Option

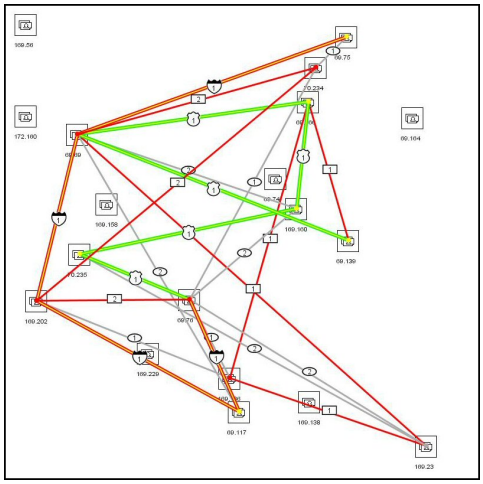


Figure 1-6: Highway option screen shot.

The Highway option calculates and highlights the most important paths in any arrangement, in which connections are visualized as edges.

1.3.4 Sizing Option

Figure 1-7: Sizing option screen shot.

The Sizing option alters the size of information objects displayed on the screen according to one of three user specified criteria.

1.3.5 Convergence Option

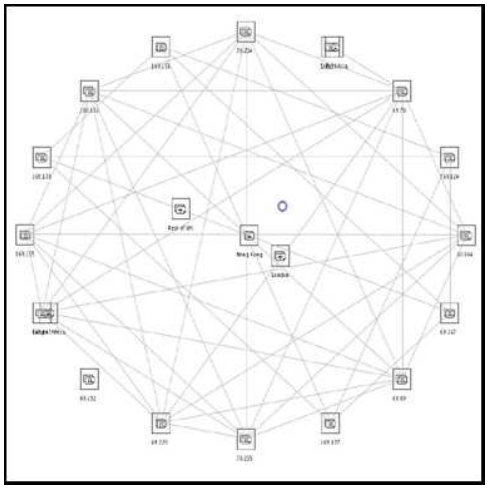


Figure 1-8: Convergence option screen shot.

The Convergence option combines different types of data and analyzes the relationship between them.

1.4 Specialized Definition

The terms defined below have specific meanings throughout the scope of this thesis.

It is important to distinguish data, information, and knowledge, as each refers to the input at different stages of the data processing procedure.

data Data are factual unprocessed values, carrying no significance on their own. Examples of data are images, text, and visitor logs.

information Once data are interpreted, they take on meaning and become processed and subjective information, containing a purpose and a meaning within a given context. Information is a collection of evidence, which provides answers to “who”, “what”, “where”, and “when” questions. Examples of information are pictures from an event, emails, and activity pattern plots.

knowledge Knowledge is a collection of applicable information, which provides answers to “how” and “why” questions. Only users can acquire knowledge but the system aims to help users by filtering out irrelevant information and hopes to inspire new insights by providing alternative perspectives. Examples of knowledge are evaluation of how successful an event was based on the pictures, agreement on an issue based on emails exchanged, and insight on a better distribution of security force based on the visitor traffic plot.

A clear distinction is made between the usage of visualization and arrangement.

visualization Visualizing refers to the action of presenting content and visualization refers to the end result on the screen. An example of visualizing process is the laying out of images. An example of visualization is a family tree chart.

arrangement Arrangements are visualization mechanisms that map the given data set to a specific visual pattern in order to focus on one unique aspect of the data, such as connectivity, distribution, or change over time. Arrangements carry out the visualizing activity and present the final visualizations to users. Examples of arrangement are random distribution, circle layout, and table structure.

For clarification, other terms with specific meanings are defined as follows:

content in context In the visualizing process, content is preserved though transferred to different contexts. In other words, the material remains unchanged as it is shifted through different frames, in which conceptual dimensions are mapped to visual dimensions.

deep copy Deep copy, also called clone, is a copy of an object that contains the entire encapsulated information of the original object, without sharing information with the object. By having no association with the original object, the clone can be used independently of the object.

visual dimension Visual dimensions are independent alterable properties in the display. Examples of visual dimension are border color, link thickness, and text size.

weblog Weblogs are logs produced by a webserver in order to record accesses to individual pages controlled by the server. Generally, entries in weblogs contain details about the pages being accessed, identities of the parties accessing the page, time stamps of the accesses, and other information that the webserver recognizes.

1.5 Overview

Chapter 2 of this thesis provides additional background information, specifically on collaboration groups and the data provider. This thesis proceeds to describe the HAPPIE system in detail. Specifically, Chapter 3 explains the design considerations and the architecture. Chapter 4 covers the data interpretation carried out by the back end. Chapter 5 and Chapter 6 list arrangements and options in the visualization front end. This thesis concludes with related work in the data visualization field, highlighted in Chapter 7, followed in Chapter 8 by a summary of the contribution that this work may offer.

Chapter 2

Related Work

This thesis work began by surveying various commercial visualization applications and academic research projects. Six of them, each representative of the various visualization philosophies, were examined in detail. I isolated and built upon the more successful aspects of these applications while avoiding the pitfalls inherent in each one.

2.1 ThinkMap

ThinkMap from Plumb Design [16] is an elaborate application with easily comprehensible user interface. It presents a diverse range of knowledge and relationships with the aid of three separate displays, as shown in Figure 2-1. The user can view connections between the current data topic and other objects in the database, while exploring the topic in more detail in the spatial and chronological dimensions.

ThinkMap generates visualizations collaboratively with users. Clients first com-

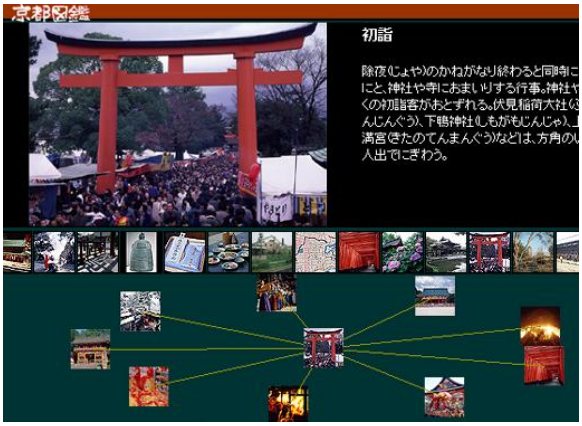


Figure 2-1: ThinkMap screen shot.

plete a data table and a relation table, and then select display properties for the appearance and interactions of data objects. Next, developers create, test, and deploy the implementation with Preview Window. Users browse and select data objects on all displays concurrently with actions customized by the developers to activate fields and changes in displays. Commonly, the last selected object moves to the center of the spider display, surrounded by connected data objects, while other displays lay out related information or related data objects.

ThinkMap is useful especially in commercial applications given its impressive, complex, and comprehensive nature. That is indeed the functionality of most of its developed applications. However, its high graphic intensity imposes high system requirements. In addition to having a multi-visualization approach, the same data objects are shown repetitively on multiple displays with no alteration in their appearance. Both of these features impose a limit on the size and complexity of each display.

2.2 Conversation Map

Conversation Map [4] visualizes text messages of a communication archive database supplied by the user. It provides a global view of the database as a whole, close-ups of specific threads and topics, as well as full details of individual messages with links to related documents. The system analyzes data and relationships to aid understanding and interpretation of knowledge. It functions efficiently with minimum inputs from the user, requiring minimal network bandwidth and graphical resources due to its mostly text based output and simple graphics, as illustrated in Figure 2-2.

Figure 2-2: Conversation Map screen shot.

The output of Conversation Map has three hierarchical levels:

Database level: Conversation Map computes four fields to display interaction, commonality, correlation, and activeness in communication threads. The Social Networks field plots participants and citations/replies as nodes and edges where the rate of interaction dictates edge lengths. The Discussion Themes field analyzes the lexical cohesion in messages and lists all topics. The Message Thread field graphs

the vivacity of communication as webs in equally sized boxes, showing density of contacts in each thread.

Communication thread level: A window made of three separate panels lists the participants, the themes, and plots a spider web graph centered at the initial message with all responses as linked nodes branching out.

Message level: System lists the subject, author, organization, group, date, ID, references, links to previous and next messages, and the message body with hypertext quotations lead quoted message.

Conversation Map is useful for modestly sized communication archives. It is not scalable due to its lack of hierarchical categorizations. Only a limited amount of legible text can fit in a given screen size. Scrolling leads to confusion and elimination of text labels will lead to ambiguity. Also, because there is no mechanism to avoid overlaps in display, group labels will eventually be overwhelming and unreadable. Its displays and functions are only developed and useful for communication archive type databases.

2.3 Graph Layout

Graph Layout repositions data points and edges to create a clean display with the least number of overlapping edges. It calculates the coordinates of nodes representing data objects and edges representing relationships based on specific criteria given by the user. A graph is then displayed on the screen with the nodes and edges, as illustrated in Figure 2-3.

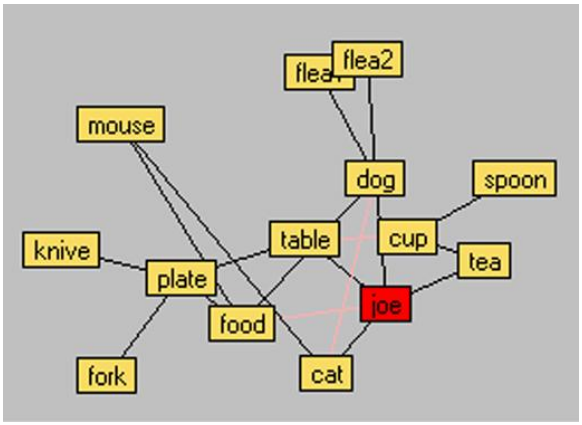


Figure 2-3: Graph Layout screen shot.

There are two deficiencies in Graph Layout. With limited visual dimensions, Graph Layout cannot represent complex data relation dimensions. Also, Graph Layout is not applicable for highly connected data, for the number of edges grow at order of n squared, and they will overlap and tangle, regardless of the algorithm.

In conclusion, Graph layout is not useful unless the purpose of visualization is to present a few types of relationships and the connections between data objects are sparse.

2.4 Visual Who

Visual Who gives the user flexibility to define grouping layout and generates a clear overview of all document correlation and global distribution.

Objects are rendered with a minimum set of visual dimensions representing all system states and developments. The system automatically repositions with addition, movement, or deletion of anchors, insuring its flexibility and scalability. It has

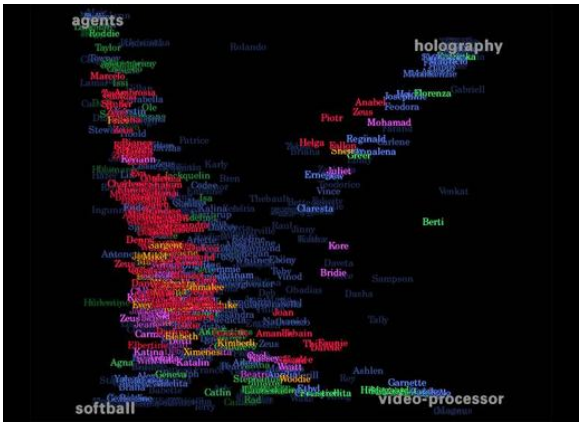


Figure 2-4: Visual Who screen shot.

fast loading time because the displays only consist of colored text user names, as illustrated in Figure 2-4.

Unfortunately, Visual Who only gives an overview of a community without browsing or reading abilities, since names overlap on the display, making it impossible to analyze concentrated areas and therefore limiting the system's effectiveness.

Visual Who utilizes a minimum of resources to illustrate a clear overview of a large community. Unfortunately, that is the only use for the system, due to lack of hierarchical structure and overlap in display.

2.5 Websom

WEBSOM automatically categorizes documents into groups based on their content. Related groups cluster to allow approximations of topics.

WEBSOM is convenient because input documents are automatically organized into labelled clusters with related neighbors. Related categories are displayed ad-

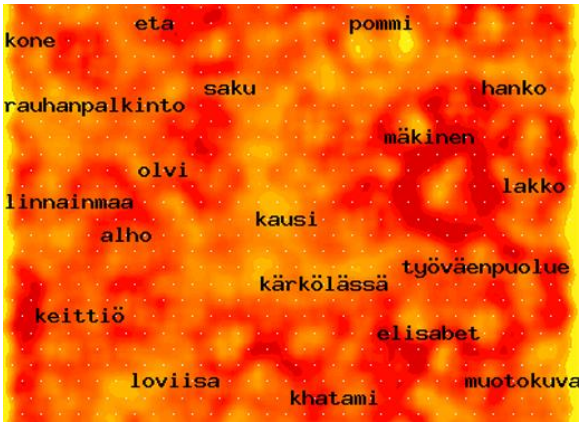


Figure 2-5: Websom screen shot.

jacently for approximation of groupings, allowing categories to overlay instead of having rigidly separated ones, as illustrated in Figure 2-5.

The number of categories is highly restricted. In a given window size, only a limited number of legible text labels can be displayed. It has a homogeneous display of clusters, but the identical look and lack of spatial pattern makes it challenging to identify clusters. It is difficult to find the correct documents because the user needs to know the exact region of the document category to retrieve it.

WEBSOM is meant to be helpful in narrowing down searches and locating related groups. However, familiarity with the categories automatically generated by the system is required to obtain accurate search results.

2.6 Chat Circle

Chat Circle is a primitive collaboration display application. It relies purely on external inputs to group messages by user name, user chosen topics and record them by

time. It is efficient in utilizing graphical visual dimensions.

Chat Circle has two screen modes: conversation mode and history mode. Solid circles and message bars are related to the user's interest threads during a specific time period; the rest are not filled.

When Chat Circle is in conversation mode, it displays each user name next to a colored circle which brightens and expands for a few seconds when the participant inputs a message, then shrinks down gradually to a dot as participant idles, as illustrated in Figure 2-6.

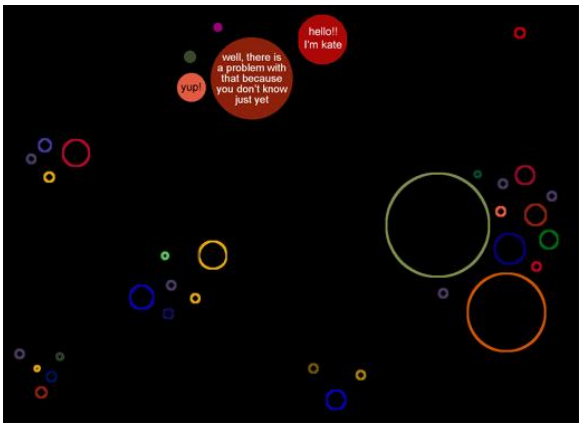


Figure 2-6: Chat Circle conversation mode screen shot.

When Chat Circle is in history mode, archives of conversations are stored as colored vertical rods for individual participants symbolizing login sessions over time. Messages posted by each participant are saved as horizontal bars placed at the posting time on the participant's own rod. Each message bar displays its message when rolled over by the mouse, and its length is proportional to its message size, as illustrated in Figure 2-7.

Chat Circle is comprehensive and focused. Both the global overview of commu-

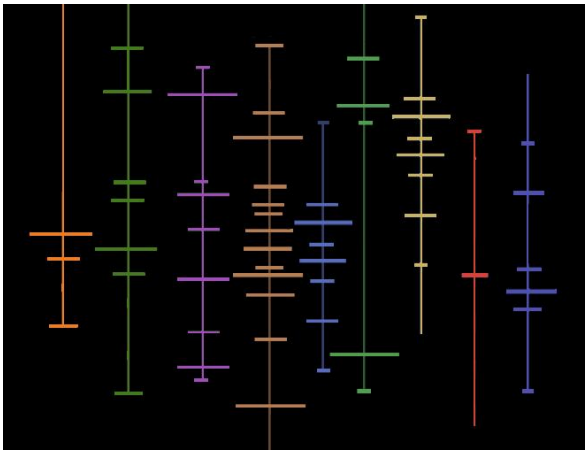


Figure 2-7: Chat Circle history mode screen shot.

nity activities as well as individual interactions are equally well presented. Especially important threads can also be highlighted. It maximizes display resources by having a one to one mapping between different data fields and visual dimensions such as position, size, color, and transparency.

Chat Circle is inconvenient because users are not allowed to determine the position of threads, and therefore cannot monitor all interesting threads unless they are near each other on the graph. Within a given window size, only a limited number of messages and users be shown at once, restricting the system's capacity. It is not efficient since regions of Chat Circle need to be re-graphed with every input, limiting the screen refresh rate. Because each user is represented by a small colored dot with a user name, it is likely to get lost in a large chat window and fail to locate a user. Finally, it does not support privacy because messages are strictly public information.

Chapter 3

Background

3.1 EWall

The EWall project concentrates on providing users with an intelligent environment to improve the efficiency and effectiveness of information collection, manipulation, and organization, as well as inter-user communication and collaboration. EWall's salient idea is the appropriate modularization level of a "card" for encapsulating information. A card puts restraints on the complexity of objects displayed, as only a modest amount of information can be presented in the area of each card. Yet, a card's information domain is not as limited as its display domain: it can contain other information such as links to related files. In addition, the format of a card provides standardization for potentially radically different information.

3.1.1 EWall modules

There are five divisions in the EWall project: the EWall Workspace Module, the EWall Exchange Module, the EWAll NewsView Module, the EWall Database Module, and the EWall Visualization Module. This thesis work implements the EWall Visualization Module.

EWall Workspace Module

The EWall Workspace Module provides a designated area for users to gather, edit, and arrange information. It standardizes each entry, regardless of the content or the format, into an isolated information object, in the predefined ECard format. An entry could be an idea, a piece of evidence for an idea, or a news item, in the format of pure text, an image, or a file.

EWall Exchange Module

The EWall Exchange Module coordinates collaboration and information sharing between individual users, between groups, and between single users and groups. Communications between Workspace Modules are recorded as transactions, containing only the updated information properties (see Section 5.1.2), to minimize communication flow and therefore increase scalability.

EWall NewsView Module

The EWall NewsView Module collects updates on existing information, messages exchanged between collaborating Workspace Modules, and outside news bits via servers communicating with news sources.

EWall Database Module

The EWall Database Module archives all ECards and modifications to them within Workspace Modules, as well as transactions between communicating Workspace Modules. The module creates relations between objects in its depository, connecting previously isolated clusters.

EWall Visualization Module

Although the HAPPIE system is designed as a stand-alone application, it also functions as the EWall Visualization Module. It can visualize the content of a single user's work on one Workspace Module. It can visualize combined content from several user's work on multiple Workspace Modules linked through the Exchange Module. It can also visualize partial or all information stored inside the NewsView module and the Database Module.

3.2 AIRE

Agent-based Intelligent Reactive Environments, AIRE [1], is a research group at the MIT Computer Science and Artificial Intelligence Laboratory. AIRE is dedicated to

examining how to design pervasive computing systems and applications for people. To study this, AIRE designs and constructs Intelligent Environments (IEs), which are spaces augmented with basic perceptual sensing, speech recognition, and distributed agent logic.

This thesis work has collaborated with AIRE group to explore potential applications and testing platforms for the system.

3.3 ARUP

ARUP is an international engineering consulting firm. This thesis work was conducted in close collaboration with ARUP personnel. Throughout the development of HAPPIE, weblogs and communication logs were provided by ARUP for testing. Bi-weekly meetings with employees from ARUP were also held, providing feedback for constant improvements to the system. A week-long site visit was made after a prototype had been completed. After presentations of the work, ARUP engineers and managers contributed further comments on the effectiveness and potential applications of the system.

Chapter 4

Architecture

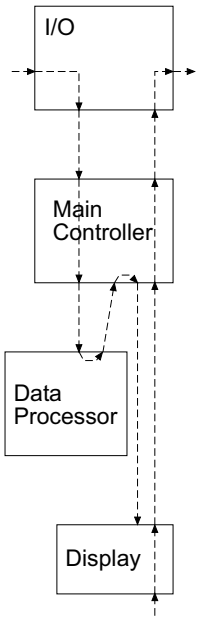


Figure 4-1: Communication flow through the functional units.

Maximizing the flexibility and the expandability of the HAPPIE system are the two central goals of the architecture of the system, as illustrated in Figure 4-1. Starting from the I/O unit of the system, detailed in Section 4.4, using the interlingua approach [3], a standard format is enforced while mini translators are created to allow various other file and transaction formats. The addition and deletion of supported formats can be conveniently accomplished by introducing new translators or removing existing translators, as explained in Section 4.4.3.

The data processor is modularized such that the entire procedure of input data interpretation can be customized, by modifying or augmenting components in the data processor responsible for the specific steps, as

described in Section 4.5.4. The operation could also be elaborated by inserting new components into the processor at any step.

Finally, the display unit allows simple addition, deletion, activation, and deactivation of its components, which are arrangements and options, as detailed in Section 4.6.3.

4.1 Model View Paradigm

The design follows the model-view-controller paradigm, explained in detail in Chapter 41 of [21]. The guiding principle of the model view paradigm is to completely isolate the model from the view, such that elements containing content information are insulated from elements containing display information. No direct communication is allowed between these two types of elements. Instead, messengers between the elements are used to deliver notifications regarding modifications, additions, and deletions. Messages from the model to the view are transported via messengers called observers, whereas messages from the view to the model travel via messengers called listeners.

The main controller, described in Section 4.3, acts as the application class, which creates the model, the view, the observers, and the listeners, initiates connections from the model to the observers to the view, as well as connections from the view to the listeners to the model.

The model is the master copy of content information, maintained in the main controller. Individual copies of information object views and connection views are

produced based on the master model copy then sent to the arrangements, detailed in Section 4.6. Skeleton information including images and text content to be visualized are included in the initial copy created for each arrangement from the main controller. Individual arrangements define all other view information, such as weight, height, and color for each information object view and each connection view during the visualization process. The final visualization, composed of all arrangements, extracts those field values from the local view copy. Visualizations are detailed in Chapter 6 and Chapter 7.

User input received by arrangements are stored in arrangements. A notification is sent to the observer responsible for the particular view element with the change. The observer probes the view element for the change and transmits the change to the main controller to update the corresponding model element in the master model copy. Symmetrically, after the main controller receives input through the I/O unit (see Section 4.4), runs it through the data processor (see Section 4.5), and updates the master model copy accordingly, and the listener affiliated with the changed model element is informed. The listener obtains the change from the model element and relays the information to connected view element.

The model view paradigm highly simplifies the architecture by allowing each arrangement to define the appearance of its view elements - in other words, how to display them. What to display, on the other hand, stays exclusively in a single master model copy with all updates amended directly on that copy. This satisfies the design goals, by allowing flexible view copies to be removed and modified, and places no restraint on the number of new view copies to be added.

4.2 Functional Units

There are four functional units: the main controller, the I/O interface, the data processor, and the display component. As illustrated in Figure 4-1, data enters the I/O unit, and then moves to the main controller. The main controller feeds the data into the data processor and receives refined output: information objects as well as the connections between those objects. The information objects and connections are then sent to the display to be presented to the user. Users can interact with the display and provide feedback, which is then propagated back to the main controller, directed to the I/O unit, and outputted from there as files or transactions.

4.3 Main Controller

The main controller is connected to all other functional units. It directs the flow of data, information, and requests from one unit to another. Information is archived, cloned, and updated as needed in the main controller.

The main controller saves a master copy of the information objects and connections. It then distributes a deep copy of the master copy to each visualization arrangement in the display unit. This way, each arrangement has independent control over its own copy, without influencing the work done in all other arrangements.

4.4 I/O Interface

The I/O interface consists of an input and an output unit, illustrated in Figure 4-2.

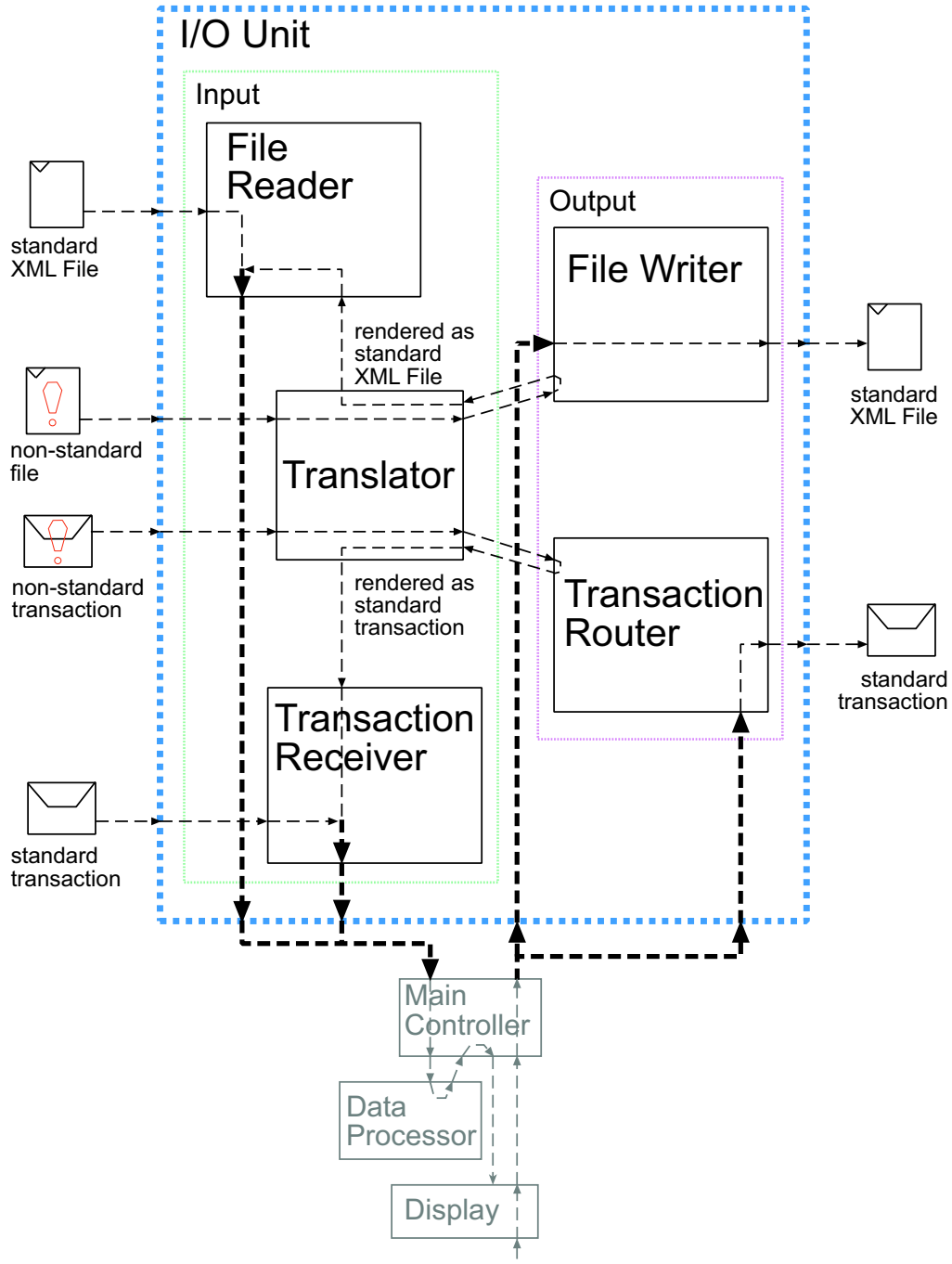


Figure 4-2: Inputs are processed by components in the input unit, then sent to the main controller. Outputs are circulated from the main controller and written into the appropriate formats by the designated components in the output unit.

4.4.1 Input

The input unit is composed of three components, as illustrated in Figure 4-2. Each of the components is designated to a specific input data type: non-standard files, standard XML files, or transactions. Non-standard files or non-standard transactions are first converted to standard format XML files or standard transactions by the translator, then sent to the file reader or the transaction receiver. Standard XML files and standard transactions are received directly by the file reader and the transaction receiver, respectively.

Translator

The translator is a collection of mini-convertors, each of which is created for a special file format or a special transaction format. All mini-convertors are linked to the file writer or the transaction router. When data are properly extracted out of the files or transactions, they are then rendered as standard XML files or standard transactions. Currently two convertors have been made, one for weblogs and the other for communication logs. Both convertors parse incoming files line by line, extracting data according to preprogrammed ordering and field value locations.

Weblog Convertor

In the following example, the weblog convertor first takes a line from a weblog, then parses the line into values for individual fields, such as time, IP address, username, retrieval method, and URL stem. Next, the file writer obtains the extracted field

values and renders them as a standard XML file, as illustrated in Figure 4-3.

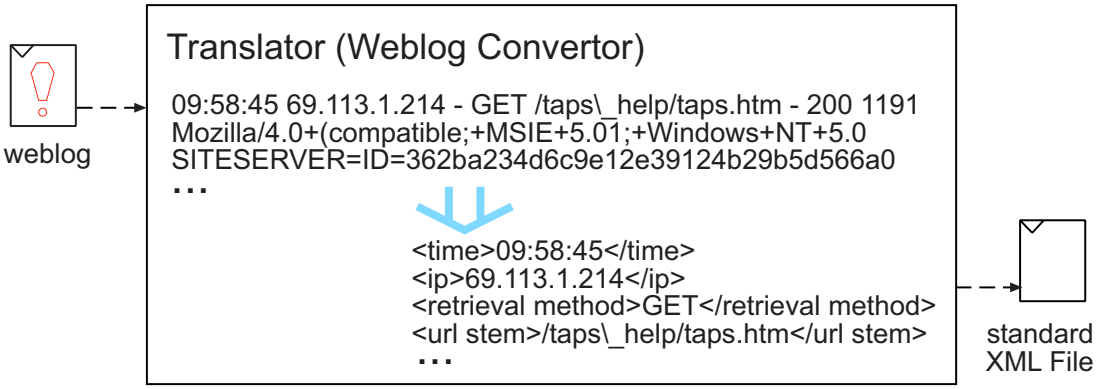


Figure 4-3: A sample line from the incoming weblog is converted to the standard XML format.

Communication Log Convertor

Similarly, the communication log convertor takes a line from a communication log, then parses it into values for individual fields, such as “from name”, “from address”, “to name”, “to address”, “cc name”, “cc address”, and “subject”. Next, the file writer obtains the extracted field values and renders them as a standard XML file, as illustrated in Figure 4-4 .

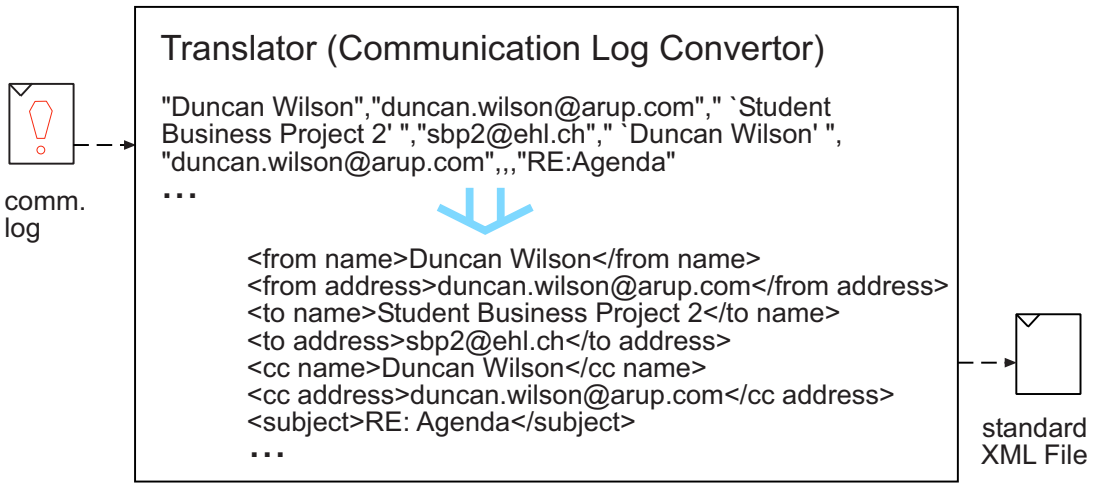


Figure 4-4: A sample line from the incoming communication log is converted to the standard XML format.

Non-standard Transactions Convertor

For a non-standard transaction, a convertor identical to the two built for weblogs and communication logs can be easily implemented. The convertor will go through the same procedures to separate data fields from the given identifier, extract data from each separated field in the specified order, and render the extracted data as a standard transaction by the transaction router.

File Reader

The file reader accepts files in the standard XML file format, as described in Section 5.1.1. Values wrapped inside the XML tags are saved as file bits, indexed by the XML start tags. The file bits are later transformed into information objects and connections, as described in Section 4.5.

Transaction Receiver

The transaction receiver accepts transactions in the standard transaction format, as described in Section 5.1.2. Values wrapped inside the XML tags indicate updates to the existing information. The values are saved as data bits, indexed by the XML start tags. The data bits are later transformed into updates on existing information objects and connections, as described in Section 4.5.

4.4.2 Output

The output unit is composed of a file writer and a transaction router, each designated to one of the two specific output types, standard files or standard transactions.

File Writer

The file writer outputs files in the standard XML file format, as described in Section 5.1.1. The output file has two parts, information objects and the connections. Each information object is wrapped inside the object's XML start and end tags signifying a complete object, as illustrated in Figure 4-5. Inside the start and end tags, start and end tag pairs indicating attribute names are placed to separate attribute values. Explanation of the information objects, connections, and standard attributes can be found in Section 5.2. Similar tags envelop individual connections and their descriptions.

```
<object>
   <ID>12345</ID>
   <time stamp>1074198308588</time stamp>
   <weight>0.6</weight>
</object>
```

Figure 4-5: A sample line from the standard XML file output.

Transaction Router

The transaction router outputs transactions in the standard transaction format, as described in Section 5.1.2. The output transaction indicates four essential properties for the update, including the type of change being made, the ID of the user requesting the update, the time stamp of the update, and the content of the update, all detailed in Section 5.2.3. The transaction is wrapped inside its XML start tag and end tags signifying a complete transaction, as illustrated in Figure 4-6. Inside the start and end tags, start and end tag pairs enclose individual information.

```
<Transaction_ version="0.1">
        <Transaction_TransactionType>4</Transaction_TransactionType>
        <Transaction_ID>
                <Identifier_ version="0.1">
                        <Identifier_Name>ben</Identifier_Name>
                        <Identifier_Hostname>maklaptop</Identifier_Hostname>
                        <Identifier_HostIp>192.168.0.4</Identifier_HostIp>
                        <Identifier_HostPort>17778</Identifier_HostPort>
                </Identifier_>
        </Transaction_ID>
        <Transaction_Timestamp>1083677226783</Transaction_Timestamp>
        <Transaction_Args>
                <SetTextTrans_Args version="0.1">
                <SetTextTrans_ID>1083677203179</SetTextTrans_ID>
                <SetTextTrans_Text>test</SetTextTrans_Text>
                </SetTextTrans_Args>
        </Transaction_Args>
</Transaction_>
```

Figure 4-6: A sample standard XML transaction output.

4.4.3 Discussion

Independent translators maximize control of the exact input file and transaction formats. When a new input format needs to be recognized by the system, only a new translator corresponding to that format needs to be added to the I/O unit. Likewise, when an input format is no longer in use, the matching translator can be removed from the I/O unit.

Equivalent control can be built for output file and transaction formats as well. Although outputs from the current version are only produced in the standard format, translators similar to the ones in the input unit can be easily implemented for the output unit and convert standard outputs into other formats.

4.5 Data Processor

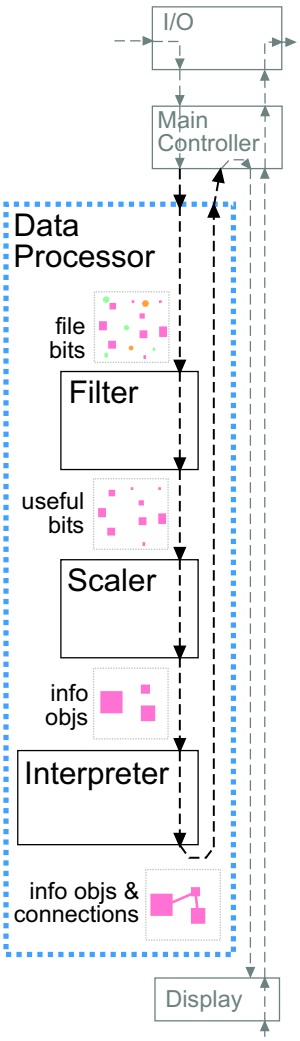


Figure 4-7: File bits sent from the main controller to the data processor are transformed into information objects and connections by the filter, the scaler, and the interpreter, and then sent back to the main controller.

The data processor is made of three parts: the filter, the scaler, and the interpreter, as illustrated in Figure 4-7.

4.5.1 Filter

File bits sent by the main controller are first passed through the filter in the data processor. The filter then parses the bits for useful data. The selection criteria are specified in Section 5.3.

4.5.2 Scaler

The filtered useful bits are output by the filter and passed on to the scaler. The scaler defines information properties with the necessary data fields. Connection properties are also defined if the input data provides data on connections. If the number of data points exceeds the predefined threshold, the points are grouped into information objects, as described in Section 5.3. Otherwise, the scaler transforms them one by one into information objects.

4.5.3 Interpreter

Finally, if no predefined connections are found between the information objects, the interpreter creates connections, as detailed in Section 5.3. The finished products of information objects and connections are then sent back to the main controller.

4.5.4 Discussion

The data processor functions as an independent entity with each data treatment step contained in separate components. Any step, or even the entire process, can be removed or replaced by changing of components. Extra components can also be added at any point of the process to further refine the data.

4.6 Display

The display unit presents the processed information objects and connections to users. The display is partitioned into arrangements and options, as illustrated in Figure 4-8, where arrangements defines the basic visualization structures and options control the outlook of the final visualizations. The Model-View-Controller paradigm, described in Section 4.1, guides the distribution of content and view information. After input data is processed into a set of information objects and connections, the set is circulated back to the main controller. The controller makes deep copies of the set and sends a copy down to each arrangement in the display.

4.6.1 Arrangements

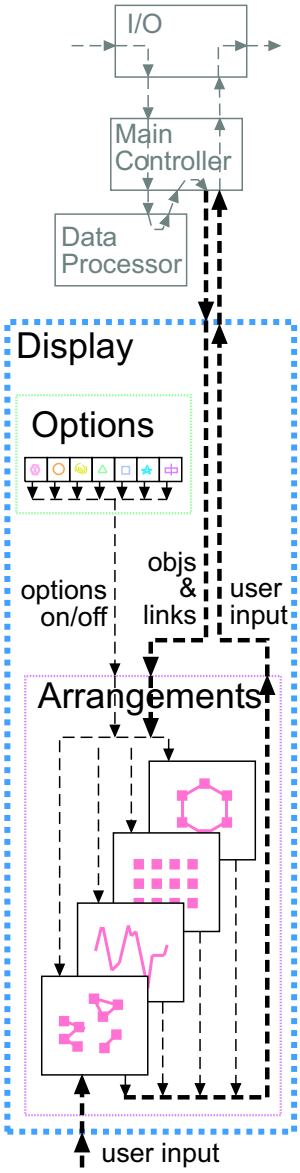


Figure 4-8: Preprocessed information flows down from the main controller to the display unit. User inputs flow back up from the display unit to the main controller.

Detailed in Chapter 6, arrangements are visualization mechanisms that map the given data set to a specific visual pattern in order to focus on one unique aspect of the data, such as connectivity, distribution, or change over time. Examples of arrangement are the Matrix Arrangement, the Circle Arrangement, and the Landscape Arrangement.

Each arrangement functions on its own and manipulates its own copy of the information objects and connections as needed. Arrangements are not aware of the existence nor the activities of other arrangements. No information is exchanged directly between arrangements at any given time. All messages travel back to the main controller and are then delivered to the appropriate components.

When users provide feedback through an arrangement, the input is directed back to the main controller for two purposes. First, the main controller adjusts the master copy, and second, it distributes the information to components, including the I/O unit for output as well as other arrangements for modification on their copies.

4.6.2 Options

Detailed in Chapter 7, options are additional features functioning on top of arrangements to control the final outlook of the visualization produced by arrangements. Examples of options are the Age Option, the Weight Option, and the Priority option, highlighting a specific information property; the Highway option, adding an extra interpretation of the existing information; and the Convergence option, allowing a few parallel information properties to be examined collectively.

Whenever users change the selection status of any of the options, that option is designated to notify all arrangements of its status change. The option requests a list of enabled arrangements from the main controller, and informs each arrangement of the change. It is then up to individual arrangements to modify their visualization following the message from the option, or to take no action if the particular option is not applicable.

A list of active options are maintained in the main controller. The main controller also keeps a lookup table, indexed by options, indicating all arrangements in which a particular option could be used. It is possible to compare the list of arrangements under a given option and the active arrangements list, when the option status changes. Therefore, when the option requests a list of arrangements to notify, only arrangements found in both lists need to be sent.

4.6.3 Discussion

Centralization of information is achieved by routing all communication flows through the main controller. An arrangement or an option can be easily added, removed, or deactivated as the only list of active items resides in the main controller.

As a result, efficiency is achieved by eliminating the synchronization process and minimizing the amount of communication. Updates in the data set are immediately acknowledged in the master copy retained by the main controller. The main controller then broadcasts the changes to components on the active items list. By doing so, information remains consistent throughout all active arrangements, and only the relevant information is distributed to a limited number of components, namely the I/O unit and all active arrangements.

An extra step to filter out the directly affected arrangements can be added when the main controller selects arrangements for change announcements. However, potential inconsistency is introduced as some active arrangements will not be aware of the changes in information.

Chapter 5

Back End

The HAPPIE system is divided into an invisible data processing back end and a visible display front end. The intelligence lies within the back end, which reads in data files, such as weblogs, news reports, and brainstorming section records, extracts useful information, and modularizes them into information objects and connections. The back end can suggest appropriate visualization selections for the particular input, based on the program's understanding of the data structure and content. The front end displays all processed information objects and connections in the selected visualizations, detailed in the next two chapters.

5.1 Input

Input to the back end includes, but is not limited to, standard format XML files and standard format transactions. Inputs of other formats are first translated into the standard formats, detailed in Section 4.4.1, before being processed by the back end.

5.1.1 File

Input data files are collections of data bit descriptions. Inside individual descriptions, detailed values for all data fields are provided. A number of data fields are marked as "required", in order to obtain basic information essential for display, such as ID, time, and title. They are identical to basic attributes for information objects, described in Section 5.2.1. In addition, there are optional data fields, providing values to other possibly useful data fields, such as notes, credibility, and website links. When value for a field can not be found, the data processor will fill in a default value or make up a value for fields that requires distinct values, such as ID.

Standard input files are in XML file format, as illustrated in Figure 5-1.

```
<Group>
        <ID>493872</ID>
        <Entries>
                <Entry1>
                        <IP>169.56.1.106</IP>
                        <Time>03:42:27</Time>
                        ...
                </Entry1>
                <Entry2>details</Entry2>
                <Entry3>details</Entry3>
                ...
        <Entries>
        ...
</Group>
...
```

Figure 5-1: A sample standard format XML file input.

Inside the start and end XML tags of the data bit (in this example, a group), all required and optional field attributes are wrapped by XML tags composed of the field name. Super attribute, such as an entry, embodies multiple sub-attributes or is comprised of multiple values. Descriptions for the sub-attributes are contained in

the super attribute's start and end tags. Inside the tags, each sub-value description is wrapped by an unique XML tag prevent the file reader from overwriting values with identical tags. Non-standard format input files are translated into the standard format, as described in Section 4.4.1.

5.1.2 Transaction

Transactions, detailed in [11] and [9], increase the efficiency in communication between components by carrying minimal information: the transaction type, ID, time stamp, and arguments. Similar to a message subject line, transaction type describes the purpose of the message, such as deletion, addition, or update requests. The unique ID provides a systematic way of distinguishing transactions even if they have identical values for other fields. The time stamp enables prioritization and archive of transactions. Finally, the arguments contain the content of the message.

Similar to standard input files, standard input transactions are also in XML format, as illustrated in Figure 5-2. An start and end XML tag pair, containing the version number of the transaction, envelops the entire transaction. Each of the four parts are wrapped individually by a start and end tag pair. Inside the tag pair for ID, user information, host IP, and other identifier values are further defined by another layer of XML tag pairs. The message body has an identical structure.

```
<Transaction_ version="0.1">
        <Transaction_TransactionType>4</Transaction_TransactionType>

        <Transaction_ID>
                <Identifier_ version="0.1">
                        <Identifier_Name>ben</Identifier_Name>
                        <Identifier_Hostname>maklaptop</Identifier_Hostname>
                        <Identifier_HostIp>192.168.0.4</Identifier_HostIp>
                        ...

                </Identifier_>
        </Transaction_ID>

        <Transaction_Timestamp>1083677226783</Transaction_Timestamp>

        <Transaction_Args>
                <SetTextTrans_Args version="0.1">
                        <SetTextTrans_ID>1083677203179</SetTextTrans_ID>
                        <SetTextTrans_Text>test</SetTextTrans_Text>
                        ...

                </SetTextTrans_Args>
        </Transaction_Args>
</Transaction_>
```

Figure 5-2: A sample standard format XML transaction input.

5.2 Output

Output from the back end consists of information objects and connections between the objects. The following are examples of such information objects and connections: website readers and association from their common visits to the same pages, individual articles and similar news content, arguments or pieces of evidence and support for/opposition to relations between them.

5.2.1 Information Objects

Identification

Each information object carries two IDs, an original ID and a current ID. When information objects are created by the back end, a unique ID is assigned to each information object as both its original ID and current ID. When information objects are cloned, for distribution to individual front end arrangements, a different unique ID is created to overwrite the value of the current ID in each information object. This allows separate arrangements to manipulate their own copies of information objects, without affecting copies of the same object in other arrangements. While the current ID makes the cloned information objects independent, the original ID allows backtracking from those copies to the master copy in the main controller. In cases when users change information objects on screen, through the active front end arrangement, the change can be propagated back to the master copy and be updated in all other arrangements.

Basic Attributes

Including the original and current ID, each information object has seven basic attributes, time, title, image, weight, and type. The type attribute has three values in the current implementation - people, location, time - but could be easily expanded to include other categories. These seven attributes allow basic data analysis, including comparison, ordering, and categorization, as well as visual display, including representation, organization, and augmentation. When values for these attributes

were not found in the input, default value can be substituted in or generated when an unique value is required.

Optional Attributes

Optional attributes are stored in an expandable hash table. The hash key is the unique name of the attribute, and the value retrieved from through the key is the value of the specific attribute.

5.2.2 Connections

Connections have three basic attributes, from-object, to-object, and weight. When a connection is undirected, the from and to objects are interchangeable. An optional attribute table is also designed to record additional information.

5.2.3 Transactions

Transactions outputted by the back end communicate updates, which are requested by users through the front end, to connected components such as the EWall Exchange Module or the EWall Database Module. Transaction format and content are identical to the standard format input transactions, detailed in Section 5.1.2.

5.3 Data Processing

Extraction

The back end first parses the incoming data for useful information. For example, it will look for text strings in the form of XX:XX:XX for time information, text strings starting with “http://” for source information, and other predefined standard formats.

Compression

The back end scales the information based on the input’s magnitude and converts them into information objects. When the size of the input is small, each information bit can be made into an information object. When the input is too large for the display, the back end groups similar information bits together to a small number of information objects. Using dynamic programming, a longest common subsequence [5] detector can be built to look for text string similarities.

Extension

The back end is also capable of creating connections between information objects based on file content’s similarities. Similar to the compression mechanism, the back end can look for information objects with similar values for random properties. A threshold should be placed to prevent a near fully connected graph being created.

Fitting

Finally, the back end can learn to select appropriate arrangements for the input data set. Depending on the data structure, a near fully connected graph, for example, can start with the Matrix Arrangement where all connections are presented on the screen in a clear manner. On the other hand, if the system recognizes an incremental link density among the nodes, the Circle Arrangement can be recommended.

Chapter 6

Front End: Arrangements

The front end of the HAPPIE system visualizes information objects and connections from data files that have been preprocessed by the back end, which is described in the previous chapter. Arrangements have been implemented to provide multiple perspectives of the same data, to verify the flexibility in the system design, and to explore innovative visualization approaches. The front end consists arrangements and options. Options are detailed in the next chapter.

The design of the front end components are based on the original designs [10] and collaboration work with Paul Keel from EWall [11]. The design follows two basic principles, mentioned in Section 1.2.2. First, it favors simple rather than complex visualizations. It uses the minimum number of visual dimensions, such as thickness, darkness, and size, to represent individual properties of the information. In other words, the front end maps only one one information property to visual dimension in order to ensure that all dimensions have an unique purpose before a new dimension is explored. The second principle is to use many visualizations in a

parallel structure rather than a single one. Each of these visualizations presents a specific aspect of the data, minimizes distraction and confusion from unnecessary information, and allows users to concentrate on one aspect of the data at a time.

6.1 Overview

Arrangements are visualization mechanisms that map the given data set to a specific visual pattern. Each arrangement guides users to focus on one unique aspect of the data, such as connectivity, distribution, or change over time. To maximize the potential capacity and visibility of each visualization, visualizations are displayed on the user's screen one at a time, allowing each to occupy the entire window space. The toolbar, shown in Figure 6-1, is placed above the window and allows users to browse from one visualization to another by selecting the corresponding button.

The spacial arrangements visualize information objects as nodes and connections as edges between the information objects. The basic nodes and edges are ones from the original information objects and connections; the basic graph is the one formed by the basic nodes and edges.

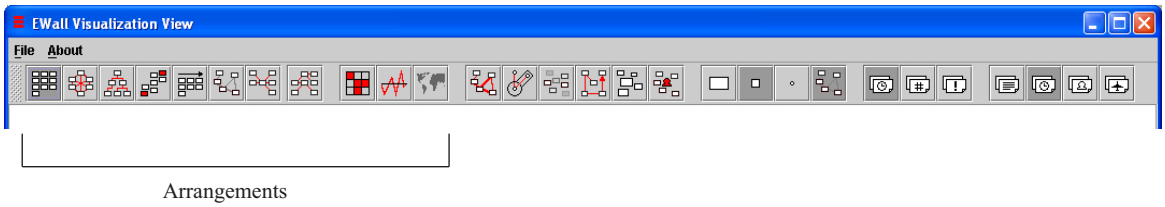


Figure 6-1: The toolbar. Buttons in the bracket correspond to various arrangements.

Four categories of arrangements have been defined based on functionality: evaluation, content, connection, and association arrangements.

6.2 Evaluation Arrangements

The evaluation arrangements are designed to serve as a reference for comparison with other arrangements. It is possible that unexpected results, providing new insights to the data, will arise with different data inputs. Two evaluation arrangements have been designed and implemented: Default Arrangement and Random Arrangement.

6.2.1 Default Arrangement

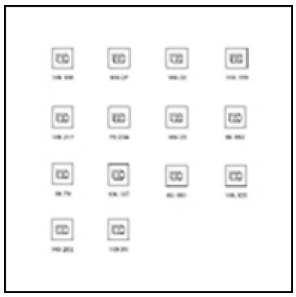


Figure 6-2: Default Arrangement screen shot.

Design

The Default Arrangement is tabular, aligning all the nodes in the order that they were created and continues onto the next line once the current line is filled, as illustrated in Figure 6-2. This arrangement presents users with all the information objects to be examined linearly.

6.2.2 Random Arrangement

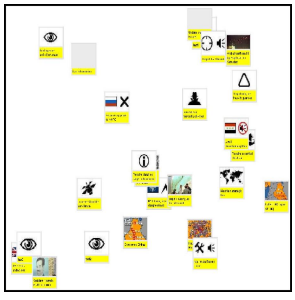


Figure 6-3: Random Arrangement screen shot.

Design

The Random Arrangement in contrast places all nodes in a haphazard fashion by randomly selecting a location for each node, as illustrated in Figure 6-3. This arrangement presents users with all the information in an arbitrary manner with minimal interference from manual direction

and built in algorithms. It grants any piece of the information an equal chance of winning the user's attention with no association between the placement/visibility and the information's actual importance.

Implementation

A random number generator assigns node placements.

Case One

When the number of nodes is large, the generator's only concern is to place each node fully inside the window. The generator picks a random real number *x*, for the x-coordinate to place the center of the node. The value of *x* falls into a range as follows: the lower limit is half the node's width as a normal size node (see Section 7.2.1); the upper limit is the length of the window minus half the node's width, as illustrated in Figure 6-4(a). The generator then picks a random real number *y*, for the y-coordinate of the node. The value of *y* falls into a range as follows: the lower limit is half the node's height as a normal size node; the upper limit is the height of the window minus half the node's height, as illustrated in Figure 6-4(b).

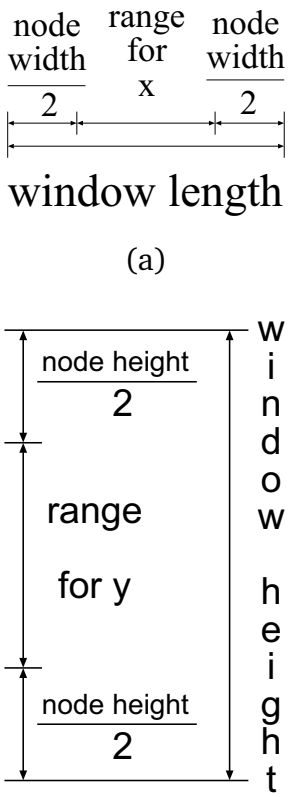


Figure 6-4: Value ranges for node center placement's (a)x- and (b)y-coordinates.

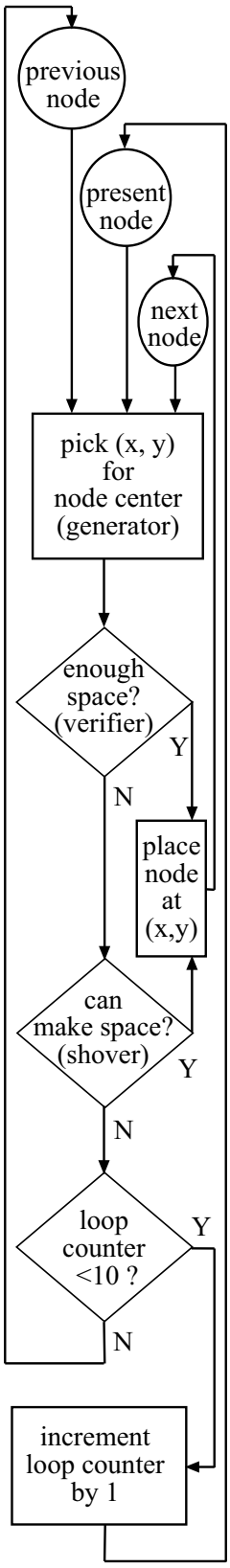


Figure 6-5:
Case two flowchart.

Case Two

When the number of nodes is small, the generator tries to fit all nodes in the window space without overlap. An extra placement step is inserted after the generator selects a node location and before the node is actually placed on the window, as illustrated in Figure 6-5. A verifier runs through the area that the node covers and ensures no previously placed nodes are already drawn in there. If the area is empty, the present node is placed and the generator proceeds to the next node. If the area is already partially occupied, a shover will push the neighboring nodes away to make room for the present node. When enough space is found, the placement is confirmed for the present node and the generator is called to locate the next node.

If the neighboring nodes are pushed against the border of the window or into other nodes before enough room can be made for the present node, the generator is called to reproduce a new possible location. In the event of an excessive number of unsuccessful attempts, currently set to ten, by the generator and the shover, the previous node

is lifted from the screen and relocated by the generator. The number of attempts is saved in the loop counter. Nodes are lifted and repositioned recursively until all nodes are placed on the window with no overlapping between nodes.

Lastly, a restart is made possible to prevent the occurrence of an infinite loop that lifts and replaces the same nodes over and over. This happens because the search is at a dead end and could not get far enough out of the search branch to explore another alternative branch. A recorder stores the ID of the lifted nodes and alerts the arrangement, when the exact same sequence of nodes is lifted consecutively for more than a predefined threshold, currently set to five repetitions in a row. Upon reception of the alert, the arrangement resets the window to blank and reinstates the generator to start from the beginning and place the first node.

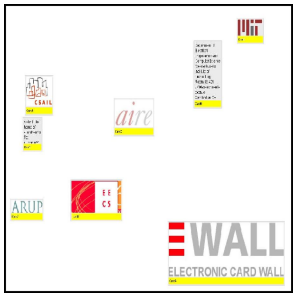


Figure 6-6: Case two screen shot.

The upper bound for the number of nodes present in this case should always be set to a small number so that the Random Arrangement will generate a result, as illustrated in Figure 6-6, within a reasonable amount of time. If a

considerable amount of time has elapsed before a satisfying layout is found or the number of nodes is considered to be too large, the arrangement will revert to the simple placement method for a large number of nodes, described in case one.

In summary, the Random Arrangement adjusts neighboring nodes' positioning when necessary to place each node on the window unobscured. When sufficient space is not found at multiple locations, the arrangement lifts and replaces the previous node in an attempt to create better spacing between nodes. Finally, the arrangement restarts from the beginning if repeated lifting and replacing appears unsuccessful.

Improvement

Ideally, the Random Arrangement will visualize information objects as physical rectangular blocks on the same level plane and use collision-detection algorithms [13] to improve the efficiency of the node placement process.

First, because nodes will self-observe and will not overlap, the verifier process will be eliminated. Moreover, upon

contact with another object, a physical object generally reacts by moving in the same direction if space allows. Therefore, imitating physical objects' reactions upon contact will empower the shover to influence not only the immediately neighboring nodes, but also the placement of all nodes in the direction of the pushing force. In other words, the present node will be given the entire available space in all directions. As a result, both the number of recursive node lifts and restarts will reduce.

Running the collision detection algorithm does not require much run-time or computational power. Incorporating it into the Random Arrangement reduces its run time and its complexity.

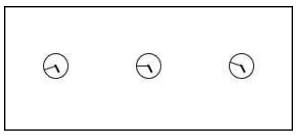
6.3 Content Arrangements

Unlike the evaluation arrangements, the content arrangements focus on the content of information objects by presenting users with addition, deletion, and alteration in content over time. Two content arrangements have been designed and implemented: Landscape Arrangement and Timeline Arrangement.

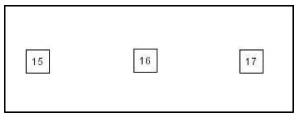
6.3.1 Landscape Arrangement

Design

The Landscape Arrangement plots the activity pattern of the data over time. The activity pattern is defined as a ratio count between two active components or attributes. For example, the number of nodes to the number of edges, the number of locations to the number of authors, or the number of light edges to heavy ones.



(a)



(b)

Figure 6-7: Time duration indicators in (a)time and (b)date format.

The x-axis represents the entire span of the activity of the visualized data set. The time duration is divided into even time intervals, represented by an array of clocks or calendar dates, as illustrated in Figure 6-7, depending on the extent of each time interval. The time symbols are placed on the center line of the window.

The y-axis covers the range from the highest ratio to the lowest. At each point in time, the ratio is calculated and a straight line is drawn between the previous point and the current one. Resulting upward peaks are named mountains, indicating the occurrence of high ratios. Downward peaks are called valleys, representing periods of low

ratios. The center line signifies the medium ratio, currently set to 1.

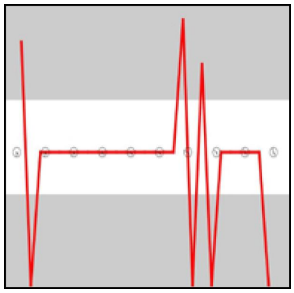


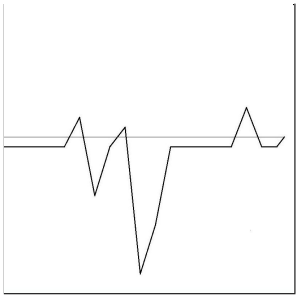
Figure 6-8: Landscape Arrangement screen shot.

Two gray rectangles, one on top and another at the bottom, are drawn on the window as alert zones, as illustrated in Figure 6-8. When a peak enters the gray zones, the ratio is either too high or too low. The contrast between white and gray zones immediately indicates balance and imbalance respectively.

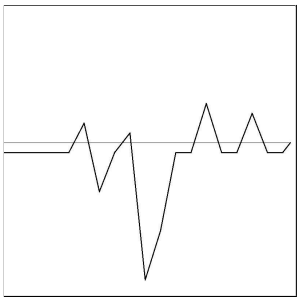
Potential Applications

Activity Pattern

The daily weblogs recording the access history of ARUP's internal newsletter site were tested and visualized in the Landscape Arrangement. The ratio between the number of active readers and the number of unique websites visited are plotted at each hour of the day by the Landscape Arrangement. High mountain peaks indicate the hours during which many users are browsing the same few pages, presumably the latest news; whereas low valley dips represent the hours during which few users are exploring a large number of assorted pages.



(a)



(b)

Figure 6-9: Screen shots showing similar daily pattern in weblogs from two different days where the number of access range from (a)a few hundred to (b)more than three thousand.

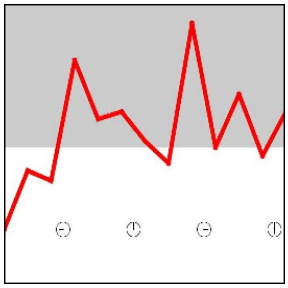
Weblogs from randomly selected days with radically different total numbers of accesses were visualized by the Landscape Arrangement, and ARUP was shown to have established daily activity patterns. A large number of limited page accesses, shown as high peaks, occurs routinely once around the morning when people arrive work and again around noon time when most people take their lunch breaks. Highly explorative page accesses happen in the afternoon, when a few workers browse the site extensively, as illustrated in Figure 6-9.

An additional feature is added to the Landscape Arrangement for the weblogs. When the user's mouse pointer rolls over the tip of a peak, extra information is provided in the form of tool-tip text. At the summits of mountains, which are created when many people browse the same few pages, the urls of the most accessed pages are listed. At the bottoms of valleys, created when few people browse many different pages, the group affiliation IDs are shown. With the extra information, users can easily associate common activity patterns with general interest groups and the specific material associated with each group. A practical

application of such knowledge is to improve the effectiveness of advertisement by targeting the desired audience with better tailored content and packaging at more appropriate locations.

Orderliness Meters

(a)



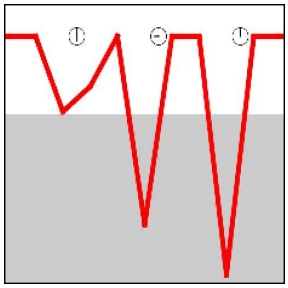
(b)

Figure 6-10: When contents in the EWall Workspace Module are (a)disorganized, Landscape Arrangement graphs (b)high peaks.

The developers of the EWall Workspace Module use the Landscape Arrangement as an orderliness meter. As users place ECards on their individual workspaces, the Landscape Arrangement visualizes the ratio between the number of active nodes and the number of connections from and to these nodes.

When users browse and collect scarcely related materials, the number of active nodes containing the materials is much larger compared to the number of connections. In this case, the Landscape Arrangement graphs a high mountain peak, indicating a high level of disorganization, as illustrated in Figure 6-10. On the other hand, when users are engaged in organizing their workspaces by filtering, refining, and grouping the information, the number of relations created is large compared to the number of modified ECards.

(a)



(b)

Figure 6-11: When contents in the EWall Workspace Module are (a)organized, Landscape Arrangement graphs (b)low peaks.

A deep valley inclination is graphed by the Landscape Arrangement to denote a high level of organization, as illustrated in Figure 6-11. In both cases, the Landscape Arrangement alerts the user through the peaks and facilitates the activity to strive for a moderate balance between structure and exploration.

Similar to the potential application of the Landscape Arrangement in the EWall Workspace Module described above, the EWall Database Module and the EWall Exchange Module also can use the Landscape Arrangement as a monitor for the organizational level of information being stored and exchanged over time, respectively.

Implementation

The Landscape Arrangement first goes through all information objects and connections and saves the earliest and latest active time stamps of the components defining ratio counts. It then divides the time duration of the components into equal length intervals and calculates the ratio count for each interval. The lowest and highest ratio counts are saved during the calculation process. Finally, the Landscape Arrangement computes the location to graph each time interval check point, as x and

y-coordinate values.

$$\text{window width} * \frac{\text{time stamp} - \text{min time}}{\text{max time} - \text{min time}}$$

The x-coordinate value is the time stamp, scaled to the window width; that is equal to the product of the window width and the quotient of time elapsed since the minimum time by the total time duration, which equals the difference between the minimum and maximum time stamp.

$$\frac{\text{window height}}{2} * \frac{\text{ratio} - 1}{\text{max ratio} - 1}$$

The y-coordinate value is the ratio, scaled to the window height; that is determined for three cases. When the ratio count equals 1, the check point lies on the center line with a y-coordinate of half the window height. When the ratio count is greater than 1, the check point is placed above the center line, forming a mountain peak. The height of the mountain is equal to the product of half the window height and the quotient of the ratio, with an offset of 1, by the maximum ratio, also with an offset of 1. When the ratio count is less than 1, the check point is positioned below the center line, forming a valley trough. The depth of the valley is equal to the product of half the window height and the quotient of 1 minus the ratio by 1 minus the minimum ratio.

$$\frac{\text{window height}}{2} * \frac{1 - \text{ ratio}}{1 - \text{ min ratio}}$$

6.3.2 Timeline Arrangement

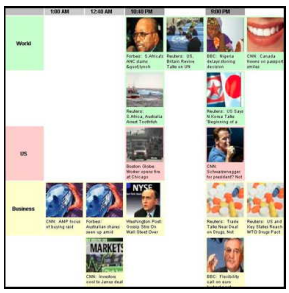


Figure 6-12: NewsView screen shot.

Design

The Timeline Arrangement places all nodes into grids inside a table. The horizontal index indicates time period and the vertical index indicates category, as illustrated in Figure 6-12. Each information object is placed in a grid corresponding to its time period and its classified category. Time is constantly updated to reflect the current active information objects. This option helps users keep track of additions and updates to existing collections of information objects, and gradually flushes outdated information objects out of the screen.

Potential Applications

Presentation Helper

A Confusion Meter [7], co-developed with Harold Fox from AIRE, provides instant feedback to the presenter using the Timeline Arrangement. Slides for the presentation are displayed in sequential order throughout the horizontal time index. The vertical category index is defined by participants attending the presentation. Through the

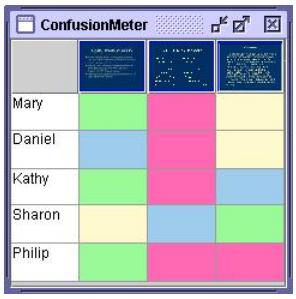


Figure 6-13: Confusion Meter screen shot.

meeting capture program [7], participants can input four instant feedbacks: agree, disagree, confused, or neutral, with neutral as the default input in case the program received no feedback from a participant. As illustrated in Figure 6-13, he four feedbacks are encoded as green, red, yellow, and blue grid background colors, respectively. Participants can also input detailed feedback through a stylus directly on top of each slide, by interacting with their tablet PC or IPad, and draw on top of the slide displayed on their screen. The feedback is then displayed in a grid in the Confusion Meter table, indexed by the individual participant and by the corresponding slide, on a screen exclusively for the presenter. At a quick glance, the column's majority background colors easily update the presenter with the audience's reactions to each slide, which allows the presenter to react immediately. For example, the presenter can pause for a quick clarification when most audience members indicate the current slide as confusing. When the presentation is finished, the presenter can examine participants comments leisurely by selecting the concerned grid and have the program display the comment over the corresponding slide.

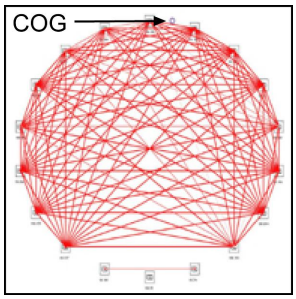
NewsView

A NewsView application has been developed by Professor Patrick Winston to offer the latest news information to users. According to users' selected preference, a news server brings the latest news bits from various sources and updates them at the specified rate. Each news story is placed in a grid indexed by the corresponding time period and the appropriate content category, as illustrated in Figure 6-12.

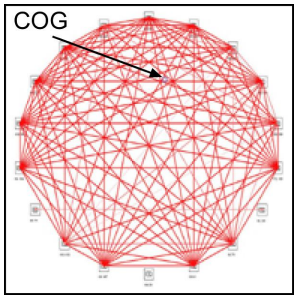
6.4 Connection Arrangements

The third category of arrangements: connection arrangements, focus on connections between nodes. They present users with density, distribution, and other characteristics of edges. A specific type of connection arrangement has been designed and implemented, called the Circle Arrangement.

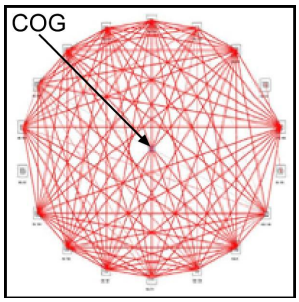
6.4.1 Circle Arrangement



(a)



(b)



(c)

Figure 6-14: Node ordering changes as the center of gravity (marked by an arrow) is moved from (a)the rim (b)towards the center of the circle until it reaches (c)the center.

Design

The Circle Arrangement lays out all the nodes in a circular configuration. A small purple ring, representing the center of gravity, is initially placed in the center of the circle for users to drag around and vary the ordering of all nodes around the circle, from random to fully ordered by their connectivity. Heavily connected nodes are attracted to the center of gravity, while lightly connected ones are repulsed by it.

When the center of gravity is placed directly on the rim of the circle, as illustrated in Figure 6-14(a), all nodes are arranged in the order of their connectivity around the circle. As users drag the center of gravity towards the center of the circle, as illustrated in Figure 6-14(b), heavily connected nodes move in the same direction as the center of gravity, while lightly connected nodes fill in the gap at the heavy nodes' previous positions. When the center of gravity arrives at the center of the circle, as illustrated in Figure 6-14(c), all nodes distribute around the circle evenly in random order.

The center of gravity allows users to dictate the ordering of nodes around the circle and observe the density distribution of the nodes' connectivity.

Potential Applications

Communication Density

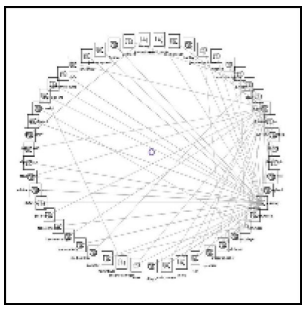


Figure 6-15: Screen shot of communication log visualization by Circle Arrangement.

ARUP provided the communication logs from a particular project which were visualized in the Circle Arrangement, as shown in Figure 6-15. Each participant was visualized as a node, and the strength of the connection between each pair of nodes indicated the number of conversations between the two participants. The project group was able to quickly discover the most and least active participants through ordering the nodes by level of communication.

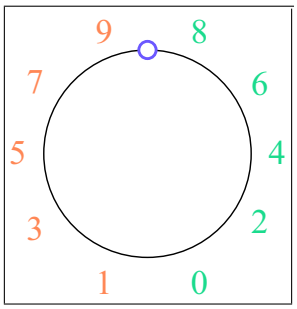
In addition, the Circle Arrangement exposed frequent exchanges between inactive members and various levels of contact between most active members. These trends, detected through the Circle Arrangement, were generally hidden in textual and even numerical data. The group found them useful in facilitating and encouraging collaboration between members.

Resource Management

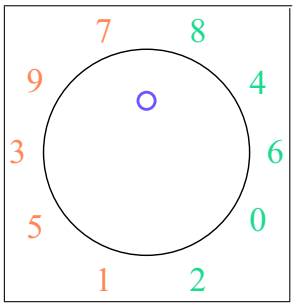
Users suggested applying the Circle Arrangement to help resource management decisions. For example, employees or groups of employees can be entered as information objects, whereas help requests and referrals can be entered as directed connections, pointing from one staff to another. When a large number of outgoing arrows is seen, indicating outgoing help requests or referrals, work load adjustment or training programs can be suggested. When a large number of incoming arrows is found, indicating incoming help requests or referrals, proper support or work load reduction can be arranged.

Implementation

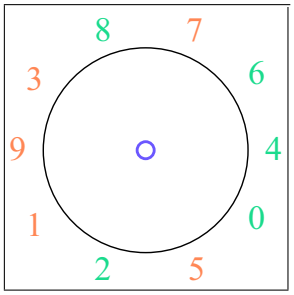
All nodes are sorted in the order of their connectivity. Nodes with the same connectivity are ordered randomly. When the edges are undirected, the connectivity of each node is defined by the total number of edges connected to it. When the edges are directed, the user can choose to count the connectivity as the sum of the incoming edges, the sum of the outgoing edges, or as the total of both types of edges.



(a)



(b)



(c)

Figure 6-16: Node ordering changes as the center of gravity is moved from (a)the rim (b)towards the center of the circle until it reaches (c)the center. They are represented by their index numbers in the sorted connectivity list.

The orderliness of the nodes' placement is proportional to the distance between the center of gravity and the rim. The proportionality constant is equal to the negative inverse of the circle's radius, and a constant offset of 1 is present.

For instance, the distance is equal to 0 when the center of gravity lies on the rim. The orderliness would equal 1, meaning all nodes are fully sorted. The sorted list of nodes is broken into two sub-lists of odd and even indices. The odd indices sub-list is placed to the left of the center of gravity whereas the even sub-list is placed to the right, as illustrated in Figure 6-16(a).

The distance decreases when the center of gravity moves towards the center of the circle. The heavily connected nodes are shuffled towards the least connected nodes. The original spots of the heavily connected nodes are filled in by the neighboring nodes with less connectivity, as illustrated in Figure 6-16(b).

The distance is equal to one when the center of gravity overlaps with the center of the circle. The nodes have been

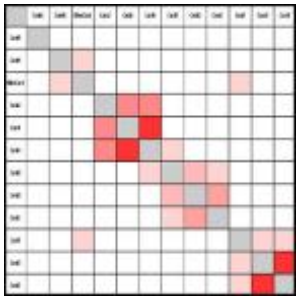
rearranged enough that a random ordering is present, with no predetermined placement for nodes based on their connectivity, as illustrated in Figure 6-16(c). Any negative distance, which happens when the center of gravity is dragged outside the circle, is considered to be zero.

6.5 Association Arrangements

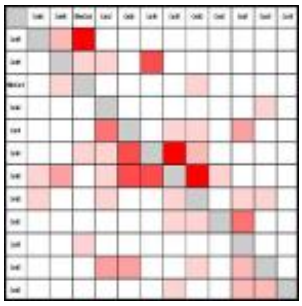
The last category of arrangements, association arrangements, focus on relations between information objects. They present users with various grouping and separation between nodes, when different relevance standards are applied in classifying and ordering nodes. Two association arrangements have been designed and implemented: Matrix Arrangement and Spring Arrangement.

6.5.1 Matrix Arrangement

Design



(a)



(b)

Figure 6-16: (a)Symmetric and (b)assymetric Matrix Arrangement screen shots.

The Matrix Arrangement maps every node to every other node by using all the nodes as both the horizontal and vertical indices. Edges are put into table cells as colored blocks. The darkness of each block indicates the weight of the edge connecting the corresponding vertical index node to the horizontal index node; the heavier the link, the darker the color. Because no edge connects a node to itself, the diagonal cells are colored gray.

When the edges have no specified directionality, the matrix is symmetric across the diagonal, as illustrated in Figure 6-17(a). When the edges are directed, an edge from node A to node B is different from an edge back from B to A, if such edge even exists. Therefore the matrix appears asymmetric, as illustrated in Figure 6-17(b). The Matrix arrangement changes users' perspective from node centered to edge centered, allowing users to easily distinguish dense connections as concentrated color blocks and trace back to associated node groups.

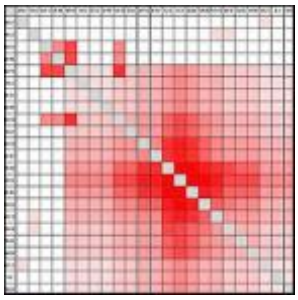


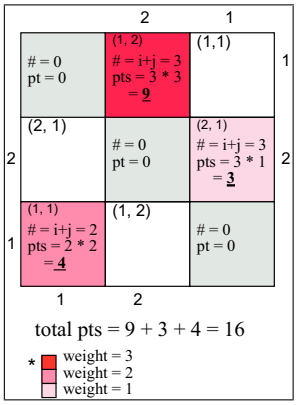
Figure 6-18: Optimized indices ordering screen shot.

The Matrix Arrangement performs a preliminary optimization of node index ordering to maximize the clustering of colored blocks around the gray diagonal, as illustrated in Figure 6-18. The darkest blocks are placed nearest to the diagonal, the second darkest the next nearest, and so on.

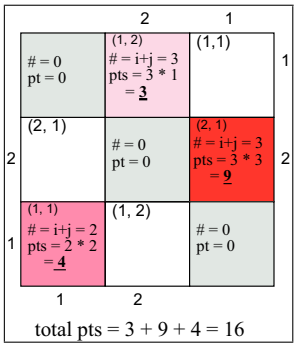
Potential Application

Road Traffic Monitor

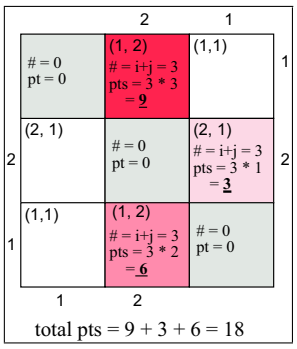
The Matrix Arrangement can monitor traffic condition. Grids with dark colors indicates congested road segments whereas light colored ones indicates uncrowded traffic roads. Busy segments automatically cluster around the diagonal line, separating themselves from the unclogged segments in the corner of the matrix. Because all grids in the same row or column indicates road exiting or entering a specific location, the result can aid users in selecting alternative paths. It could help road planning in the long term. For example, if all roads leading to a particular region, represented by grids indexed by the region are constantly colored dark, construction plans could be proposed to increase the road capacity or build de-tours allowing drivers to take an alternative path to their usual routes of passing through the region.



(a)



(b)



(c)

Figure 6-19: Matrix Arrangement point system. Different sample placements with (a)initial, (b)same, and (c)improved total points.

Implementation

Hill climbing search [5] is used with optional restarts to generate an optimal ordering of the card indices. A point system is designed to implement the Matrix Arrangement.

Each block is assigned with a number proportional to the block's distance from the gray diagonal, which connects the top left and bottom right vertices of the matrix. Blocks with equal distance to the diagonal will be assigned the same number. The total points for a Matrix Arrangement are equal to the sum of each link weight multiplied by its grid number, as illustrated Figure 6-19(a). The program randomly switches the placement of selected pairs of indices, and only preserves the switched configuration if the total points do not decrease, as illustrated in Figures 6-19(b) and (c).

When the Matrix Arrangement first starts up, a limited number of switching attempts is made to provide users with a preliminary arrangement to review in a reasonable amount of time. Users can request, using menu buttons, to continue improving the ordering of the card indices, or to restart with a random ordering of the indices.

6.5.2 Spring Arrangement

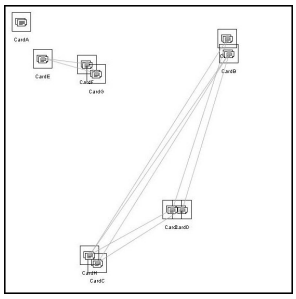


Figure 6-20: Spring Arrangement screen shot.

Design

The Spring Arrangement treats each edge as a spring, with the inverse of the weight on the edge as the natural length of the spring. Two nodes with a strong connection have a naturally short spring between them, whereas two nodes with a weak connection are joined by a long spring. Each spring exerts a pulling force when stretched and a pushing force when compressed.

The Spring Arrangement adjusts nodes' placements to minimize energy resulting from total force needed to keep all springs, edges between nodes, at their present length. Because both pulling and pushing forces result in a positive energy increase, stretched springs will not counterbalance compressed springs; thus, all edges need to be restored as close to their natural lengths as possible. As a result, nodes with strong connections come close to each other while weakly linked nodes maintain a moderate distance.

By relating distance in space to relevance in context, the Spring Arrangement allows users to easily pick out

individual interest groups visually, as well as immediately related information surrounding a specific object.

Implementation

The basic spring energy system is modelled after [6]. The positioning of nodes is tweaked, node by node, to minimize the total energy, until equilibrium is reached and energy no longer reduces despite any adjustment attempts. Invisible edges with very long lengths are added between every cluster of connected nodes. Thus, clusters are clearly separated by the force of clusters pushing away neighboring clusters, as illustrated in Figure 6-20.

Because the energy formula includes the mass of objects, the importance of nodes can be used to determine their weights to further differentiate the significance of relative distance between nodes. Given the same pushing or pulling force, a higher mass in the originating node leads to a higher energy. Consequently, nodes closely related to an important node will gain a higher priority in restoring their distance. Nodes unrelated to an important node will also have a higher chance of restoring their original distance in being pushed away.

Chapter 7

Front End: Options

The front end of the system, described in the beginning of the previous chapter, has two components: arrangements and options. Arrangements are visualization mechanisms, deciding the placement and structure of objects being displayed, explained in the previous chapter. To allow flexibility and increase variation in display, options are also implemented in addition to arrangements.

7.1 Overview

Options are additional features functioning on top of arrangements to control the final outlook of the visualization produced by arrangements. Examples of options are the Age Option, the Weight Option, and the Priority option, highlighting a specific information property; the Highway option, adding extra interpretation on the existing information; and the Convergence option, allowing a few parallel properties to be examined collectively.

Users can turn on one or more options within an arrangement to highlight specific characteristics of the information. As users go from one arrangement to another, the applicable options are made available by enabling the corresponding buttons in the top tool bar, shown in Figure 7-1. The user can then activate individual options by selecting the corresponding buttons. The selected options for the current arrangement will remain active if they are also available in the next arrangement.

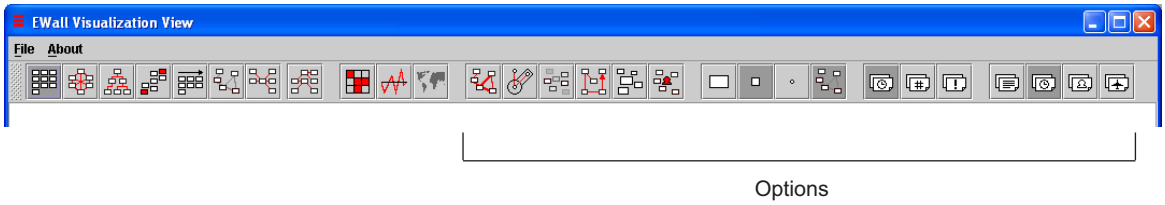


Figure 7-1: The toolbar. The buttons in the bracket correspond to various options.

There are three types of options: size, link, and context options.

7.2 Size Options

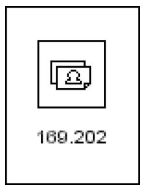
The size options control the display size of the nodes. Only one of the three size options can be turned on at any given time, because each node can only be of one size. Three size options have been designed and implemented: Normal Node Size, Medium Node Size, and Small Node Size.

7.2.1 Normal Node Size

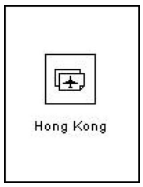
Figure 7-2:
An E-Card.

The Normal Node Size option displays each information object as an "E-Card", implemented by the EWall workspace group, which provides all the detailed information, as illustrated in Figure 7-2. At the top of the E-Card, there is an image associated with the information object. Below the image, there is an icon bar containing three icons: a left icon linking to a webpage or a file, a middle icon indicating the credibility of the card, and a right icon linking to comments made on the card. At the bottom of the E-card, the time stamp, author, and title are displayed.

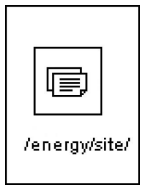
7.2.2 Medium Node Size



(a)



(b)



(c)

Figure 7-3:
(a) A person/IP node; (b) a location node; (c) a content node.

The Medium Node Size option focuses on three common fields found in most data: people, location, and content. Once this option is switched on, each node is visualized with a small image on top and a short piece of text at the bottom, as illustrated in Figure 7-3. The image is one of three equally sized icons representing the three fields separately. The text underneath the image describes the value of the represented field.

The Medium Node Size option is especially useful when combined with the Convergence option, which indicates which of the three fields users would like to concentrate on, as discussed in Section 7.4.7.

7.2.3 Small Node Size

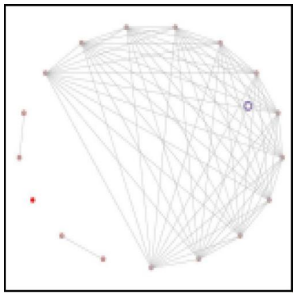


Figure 7-4: Small Node Size option screen shot.

The Small Node Size option allows users to focus solely on connections and overall structure instead of content details, minimizing distraction from the presence of nodes. When the option is selected, all nodes on the screen are shrunk down to the size of a dot, as illustrated in Figure 7-4, to direct users' attention to the connectivity of the edges. This aids users in better understanding the general visual pattern that a given set of data exhibits. In addition, more nodes can be plotted into the same area as each node occupies less screen space.

7.3 Link Options

The link options alter the visibility, directionality, appearance, and placement of the edges. Each option can be turned on and off independently because link options do not interfere with each other. Two link options have been designed and implemented, Highway and Link Visible.

7.3.1 Highway

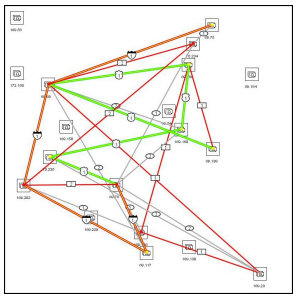
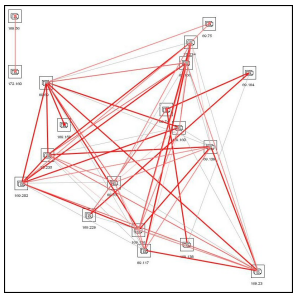


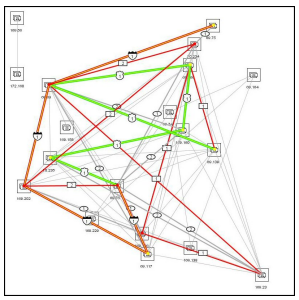
Figure 7-5: Highway option screen shot.

Design

The Highway option computes the most important paths, the highways, and plots them on top of the original edges. Each highway is drawn with thicker and brighter strokes than the basic edges, as illustrated in Figure 7-5and is labelled with a highway symbol like those found in conventional maps. The importance of each edge is defined by its weight, which has been assigned by the back end. The importance of each path is determined by the combined importance of all the edges in that path. Consequently, highways are the paths with the most edges and the heaviest combined weight, just as physical highways are designed to accommodate both heavy and long-distance traffic.



(a)



(b)

Figure 7-6: (a)Basic search graph with (b)highway mapping on top of it.

The Highway option first creates a search graph identical to the basic graph. The option then browses through the search graph for a primary highway, assigns the highway number 1, and removes all edges on the highway from the search graph. To qualify as a highway, a path must have a minimum number of edges, and/or a minimum total weight. The option proceeds to search for more highways and to assign incremental numbers to each highway in the order it is found. The search halts when no additional qualifying highway can be located. As a result, the number 1 highway is the largest highway in the graph, number 2 the second largest, and so on. No highways overlap because the edges on each highway have been removed as each one was identified on the search graph.

A transparent layer displaying only the highways is placed on top of all other display layers to ensure the full visibility of highways above all nodes and basic edges. When the Highway option is turned on, the highway layer is made visible. When this option is turned off, the layer is hidden from the user.

Three implementation approaches have been explored for locating highways.

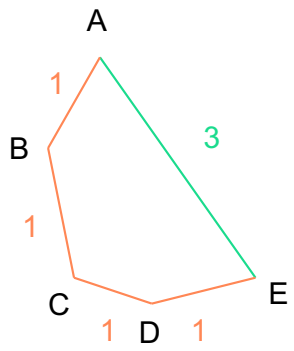
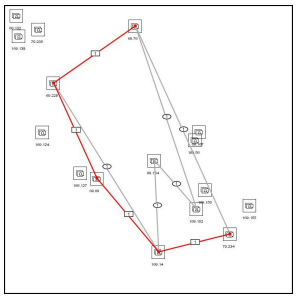


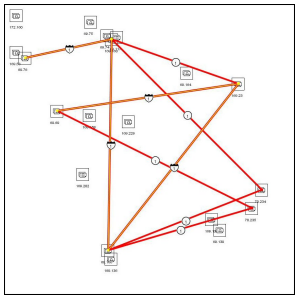
Figure 7-7: The total length for path A-B-C-D-E is 4 versus 3 for path A-E. However, A-E should be the chosen highway because its edge weight is larger than any single edge in the first path.

The first approach counts the length of each edge as its weight, symbolizing the traffic flow. The longest path is computed with a modified version of Floyd-Warshall algorithm, detailed in [5], for discovering all-pairs shortest paths in a graph. The longest possible path between all-pairs of nodes and their length is deduced using dynamic programming and recorded in a matrix. This approach uses greedy search [20] to locate the path with the absolute longest total length. However, when a different path with higher traffic but fewer edges exists, such as the example illustrated in Figure 7-7, this approach fails to compromise path length for higher average traffic flow.

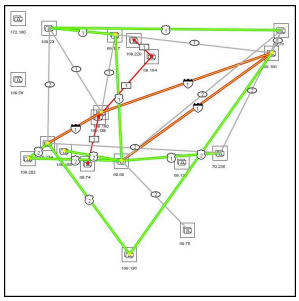
The second approach takes this defect into account by modifying the total length of a path to be its total weight divided by the total number of edges within the path. This metric discounts the actual path length to ensure that most edge segments forming the highway have substantial weights, hence traffic flow. Recursive searches, identical to ones used in the first approach, are carried out until no



(a)



(b)



(c)

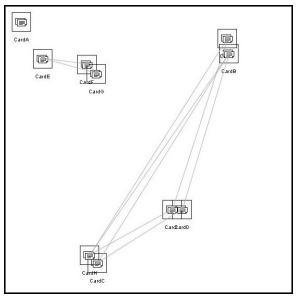
Figure 7-8: Highways: (a)level one and two; (b)level three and four; and (c)level one, two, four, and five.

qualifying highways can be found.

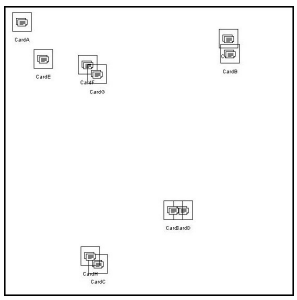
The final approach maintains the metric, which uses total path weight as path length, as described in the first approach. Edges are first categorized by their weights into five groups at equal intervals between the minimum and maximum edge weights. Independent highway searches, identical to ones in the first approach, are performed within each interval. Each search halts when no more qualifying highways can be found within the specific interval.

The highways generated in this manner are analogous to highways in real life. Expressways consisting of relatively low traffic roads are modelled by highways in the lowest weight interval. Similarly, the remaining four intervals correspond to equivalent versions of local freeways, state freeways, interstate freeways, and federal freeways, with incremental traffic flow, as illustrated in Figure 7-8. Five transparent layers are used to display the highways from different intervals in a hierarchical order. The layer with highways formed by the heaviest edges are placed on top; the layer with highways formed by the second heaviest edges are placed underneath it, and so on.

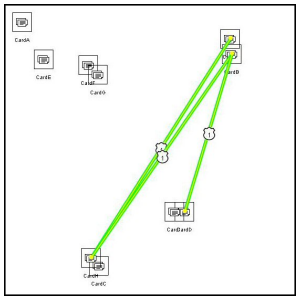
7.3.2 Links Visible



(a)



(b)



(c)

Figure 7-9: Link Visible in Spring Arrangement screen shots: Links in (a)the basic graph are (b)invisible for cluster analysis or (c)highway highlights.

Design

The Links Visible option controls the visibility of the basic edges. The edges are made visible when the option is selected and invisible when the option is deselected, as illustrated in Figure 7-9. This option allows the user to focus solely on the placement of nodes in the area, the content of the nodes, and other visible components in the window.

Implementation

A transparent layer displaying only the edges is placed on top of the layer displaying all nodes to ensure the full visibility of edges above all nodes. When the Link Visible option is turned on, the edge layer is made visible. When this option is turned off, the layer becomes hidden from the user.

7.4 Context Options

The context options visualize additional properties or fields of the information objects through other visual channels, such as brightness, and additional components, such as arrows for edges. Ten context options have been designed and implemented, Sequence, Transparency, Sizing, Age, Weight, Priority, Convergence, Location, People, and Content.

7.4.1 Sequence

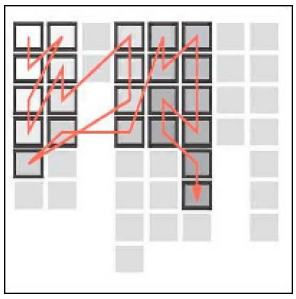


Figure 7-10: Sequence option mock-up.

Design

The Sequence option adds a set of directed edges to the existing ones. This new set of edges connects nodes in chronological order, such that an edge points from a node with an earlier timestamp to the one following it in time, as illustrated in Figure 7-10. This option allows users to trace nodes in the graph in a different way. Also, because all nodes follow one after another, it creates connections among all the nodes in the graph, and makes sparsely connected graphs more meaningful by adding new relations to completely unrelated clusters.

Implementation

The Sequence option sorts all nodes in chronological order beginning with the earliest node in time and ending with the latest. This option then creates an edge pointing from each node to its succeeding node. To display the edges, a transparent layer displaying only the sequence edges is placed on top of the basic edge layer to ensure that sequence edges overlap above basic edges when both are visible to users. When the Sequence option is turned on, the sequence edges layer is made visible. When this option is turned off, the layer becomes hidden.

7.4.2 Transparency

(a)

(b)

Figure 7-11: When the Transparency option is selected, the group of options to the right is enabled; when the Transparency option is deselected, the group is disabled.

Design

The Transparency option acts as a super-option and enables sub-options Age, Weight, or Priority, as illustrated in Figure 7-11. One and only one sub-option can be enabled at a time.

The Transparency option controls the opacity of each node. All nodes are ranked according to a common property, such as age or weight, determined by the enabled

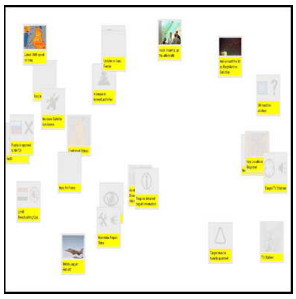


Figure 7-12: Transparency option screen shot.

sub-option. The higher ranked nodes will appear in full color, whereas the lower ranked ones will be transparent, as illustrated in Figure 7-12. Unimportant information fades into the background. This option directs users' attention to the more valued information objects while still presenting the rest in a subtle way.

Implementation

Two implementation approaches have been tested and evaluated for the Transparency option. Both options assign each node a transparency count. Depending on the selection status of the three sub-options, the transparency count is determined by its age, weight, or priority value. This option sorts all nodes according to the selected sub-option value and sets the transparency count of each node as its ranking normalized by the total number of nodes.

The first approach is to fade nodes evenly by putting a blank screen over each node on the window and adjusting the transparency of the screen according to the transparency count. Nodes with lower transparency counts

(a)

(b)

(c)

Figure 7-13: An ECard is (a)fully opaque, (b)uniformly and (b)partially transparent.

are covered by near white screens; nodes with higher transparency counts are covered by clear screens; all nodes in between are covered by screens with intermediate levels of transparency, as illustrated in Figure 7-13(b).

The second approach is to fade nodes unevenly, by fading individual components of the ECard. The transparency counts are used to set the appropriate alpha color channel values of all images and colors, which are used in drawing borders, filling in backgrounds, and writing out characters. Alternatively, individual images, colors, and brush strokes can be faded to various degrees. For example, while the image and characters change from fully visible to nearly invisible, the border can be set to a range between full colored and slightly faded. The text characters can remain fully legible while images and background color fade away, as illustrated in Figure 7-13(c), as visual indication of a node's significance. While both approaches yield this desired effect, the first has the advantage of simplicity in implementations and minimizes display lag time, whereas the second allows maximum flexibility in usage.

7.4.3 Sizing

(a)

(b)

Figure 7-14: When the Sizing option is selected, the group of options to the right is enabled; when the Sizing option is deselected, the group is disabled.

The Sizing option acts as a super-option which enables sub-options Age, Weight, or Priority, as illustrated in Figure 7-14. One and only one sub-option can be enabled at a time.

The Sizing option alters the size of all nodes when they're displayed in normal size (see Section 7.2.1). All nodes are ranked according to a common property, such as age or weight, determined by the enabled sub-option. The higher ranked nodes will appear in larger sizes, whereas the lower ranked ones will be smaller, as illustrated in Figure 7-15. Unimportant information takes up less screen space, allowing users to discount it. This option directs users' attention to the more valued information objects while still presenting the rest in a subtle way.

Figure 7-15: Sizing option screen shot.

7.4.4 Age

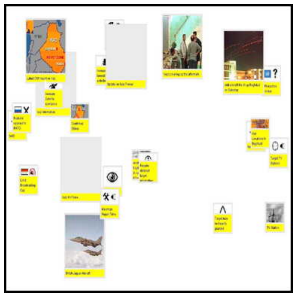


Figure 7-16: Screen shot of the Age option with the Sizing option.

The Age option is a sub-option of the Transparency option and of the Sizing option, detailed in Section 7.4.2 and Section 7.4.3. As illustrated in Figure 7-16, this option directs arrangements to determine the size or opacity, depending on the selection status of the Transparency option and the Sizing option, using the age of information objects, when they are displayed as normal size nodes (see Section 7.2.1).

7.4.5 Weight

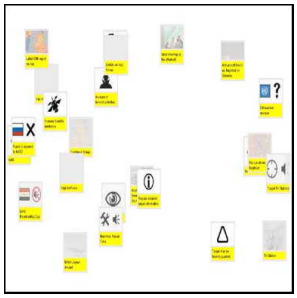


Figure 7-17: Screen shot of the Weight option with the Fading option.

The Weight option is a sub-option of the Transparency option and of the Sizing option, detailed in Section 7.4.2 and Section 7.4.3. As illustrated in Figure 7-17, this option directs arrangements to determine the size or opacity, depending on the selection status of the Transparency option and the Sizing option, using the weight of information objects, when they are displayed as normal size nodes (see Section 7.2.1).

7.4.6 Priority

Figure 7-18: Screen shot of the Priority option with the Sizing option.

The Priority option is a sub-option of the Transparency option and of the Sizing option, detailed in Section 7.4.2 and Section 7.4.3. As illustrated in Figure 7-18, this option directs arrangements to determine the size or opacity, depending on the selection status of the Transparency option and the Sizing option, using the importance of information objects, when they are displayed as normal size nodes (see Section 7.2.1).

7.4.7 Convergence

(a)

(b)

Figure 7-19: When the Convergence option is selected, the group of options to the right is enabled; when the Convergence option is deselected, the group is disabled.

The Convergence option acts as a super-option which enables sub-options Location, People, and Content, as illustrated in Figure 7-19. One and only one sub-option can be enabled at a time. When combined with the medium size option described in Section 7.2.2, each node is clearly distinguished by its icon associated with one of the three fields-location, people, and content. As discussed, the Convergence option can help users discover hidden characteristics in the data.

7.4.8 Location

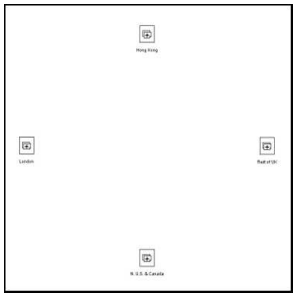


Figure 7-20: Screen shot of the Location option combined with the Circle Arrangement.

The Location option is a sub-option of the Convergence option, detailed in Section 7.4.7. This option directs arrangements to visualize location related information objects as the main nodes, as illustrated in Figure 7-20.

7.4.9 People

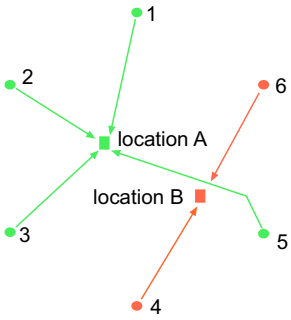
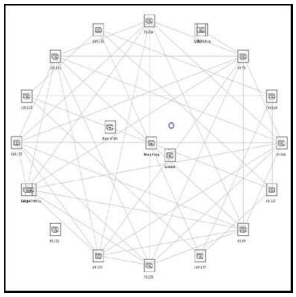


Figure 7-21: Mapping of location nodes' placement. Location node A is mapped in the middle of the center of people nodes 1, 2, 3, and 5, whereas location node B is mapped in the middle of nodes 4 and 6.

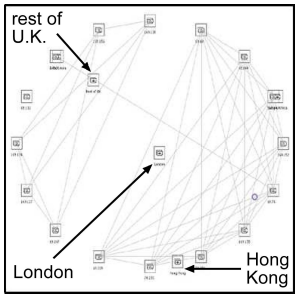
Design

The People option is a sub-option of the Convergence option, detailed in Section 7.4.7. This option directs arrangements to visualize people-related information objects as the main nodes. Location nodes, representing location-related information objects, are then mapped in the center of gravity of all the people from the particular location, as illustrated in Figure 7-21. As users manipulate the placement of people nodes, placements of location nodes are updated in real time, as illustrated in Figure 7-22.

Potential Application



(a)



(b)

Figure 7-22: Screen shots of the People option with the Circle Arrangement. When the people nodes on the rim are reordered, the location nodes' placements are updated accordingly.

Weblogs were mapped in Circle Arrangement with connections between readers who accessed the same pages, indicating similar interests. In Figure 7-22(a), when the center of gravity is at the middle of the circle, people nodes are distributed randomly along the rim and location nodes are mapped closely to the center of the circle. A quick observation could lead to the conclusion that all three locations have an even compilation of people with various levels of similar interests with others in the company.

However, when the center of gravity is moved to the lower right rim, people nodes are ordered by the number of common interests they have with others in the log, as illustrated in Figure 7-22(b). As a result, all three location nodes quickly become separated. London node is at the center, implying a collection of people with balanced common interest levels distributed throughout the sorted range of people nodes. On the other hand, Hong Kong and the rest of U.K. are on the opposite ends of the interest levels, with Hong Kong among people with the most similar interests, and the rest of U.K. among ones with least similar interests.

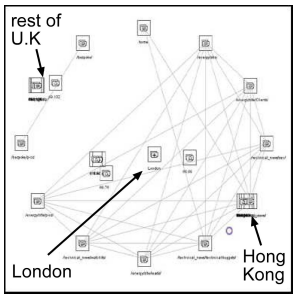
Implementation

The People option calculates the proper placement of each location object using all people objects belonging to the particular location. The x-coordinate value of the location object equals the sum of the x-coordinate values of all those people objects divided by the total number of those people. The y-coordinate value of the location objects is calculated the same way, using the y-coordinate values of all those people.

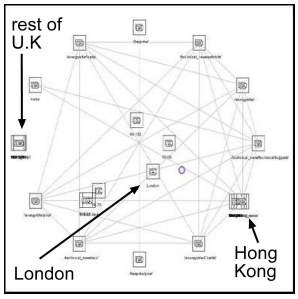
7.4.10 Content

Design

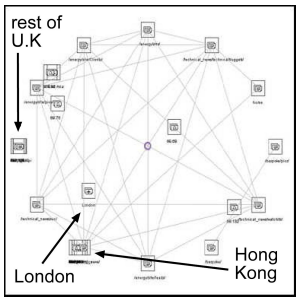
The Content option is a sub-option of the Convergence option, detailed in Section 7.4.7. This option directs the arrangements to visualize content related information objects as the main nodes. As illustrated in Figure 7-0, people who are associated with the content objects are mapped in the middle of the content nodes in the same way the location cards are mapped from the people nodes, identical to the mapping algorithm in Section 7.4.9.



(a)



(b)



(c)

Figure 7-23: Screen shots of the Content option with the Circle Arrangement. When the people nodes on the rim are reordered, the location and content nodes' placements are updated accordingly.

Potential Application

Following the scenario of the People option in Section 7.4.9, website content related nodes are now placed at the rim of the circle with reader related nodes mapped in the center of all accessed websites, and location related nodes are mapped in the center of the all the people belonging to that location.

In Figure 7-23(a), the center of gravity is on the lower right rim and orders all content nodes according to the number of common readers they have. Similar to the final discovery from the People option, London is near the center of the circle, with Hong Kong overlapping one of the most popular websites near the center of gravity, and the rest of the U.K. located between websites of little interest to most people. As the center of gravity is moved towards the center, illustrated in Figure 7-23(b) and (c), all three location nodes remain close their original positions. The discovery from the People option is not only confirmed but narrowed down to an identical conclusion regarding the content interest distribution of people from each location.

Chapter 8

Conclusion

8.1 Results

Though no quantitative user study has been conducted to gauge the success of the HAPPIE system, I have collected anecdotal feedback through user surveys at ARUP. In comparison with previous work in [12], which used the same types of data as HAPPIE but was evaluated as being confusing and unhelpful, I have received collectively positive responses and enthusiastic application suggestions from the same users who tested [12].

Specifically, users appreciated the short learning process due to the simplicity of each visualization. It encouraged them to explore various arrangements and options freely. They were able to gain insights from the visualizations of real data from ARUP, including analysis of communication density (see Section 6.4.1), verification of activity pattern (see Section 6.3.1), and discovery of hidden data characteristics (see Section 7.4.9 and Section 7.4.10).

8.2 Contribution

The main contribution of this work is the proposal of a feasible architecture for the HAPPIE system, which would provide high flexibility as well as extensibility. I have built a prototype demonstrating the success of the proposed design, and have gradually expanded it to include arrangements and options from each category, as well as data processing steps, proving the design is easily expandable. To confirm the adjustability of the design, I have tested various file types and different levels of data processing, in addition to modifications of arrangements and options. Furthermore, I have researched, implemented, and analyzed algorithms for each arrangement and option, selecting the most successful candidates for inclusion in the system. Potential improvements to these algorithms are also discussed. Real data sets were provided by ARUP throughout the development of the prototype. Personnel with extensive knowledge of ARUP monitored the progress closely and provided constant feedback on the effectiveness of each arrangement and option. I then improved the system based on this feedback. I also conducted surveys to gauge the general impression and uncover potential applications of the prototype after presenting the work during a week-long site visit at ARUP, and received positive feedback.

Bibliography

[1] Aire. http://www.ai.mit.edu/projects/aire.

[2] anemone. http://acg.media.mit.edu/people/fry/anemone/.

[3] Ronald Cole, Joseph Mariani, Hans Uszkoreit, Giovanni Varile, Annie Zaenen, and Antonio Zampolli, editors. *Survey of the State of the Art in Human Language Technology*. Cambridge University Press, March 1998.

[4] Converstationmap. http://www.sims.berkeley.edu/

[5] Thomas Cormen, Charles Leiserson, Ronald Rivest, and Clifford Stein. *Introductio to Algorithms*. The MIT Press, Cambridge, MA, 2nd edition, 2001.

[6] Giuseppe Di Battista, Peter Eades, Roberto Tamassia, and Ioannis Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall, 1st edition, July 1998.

[7] Harold Fox. The efacilitator: A meeting capture application and infrastructure for intelligent environments. Master's thesis, Massachusetts Institute of Technology, September 2003.

[8] Michael Friendly and Daniel Denis. Milestones in the history of thematic cartography, statistical graphics, and data visualization -an illustrated chronology of innovations. http://www.math.yorku.ca/SCS/Gallery/milestone/.

[9] Michael Kahan. Computer analysis: Learning and creation of arrangements of information. Master's thesis, Massachusetts Institute of Technology, Cambridge, MA, June 2004.

[10] Paul Keel. Process and relation analysis: Capturing architectural thought. Master's thesis, Massachusetts Institute of Technology, Cambridge, MA, February 1997.

[11] Paul Keel. *Knowledge Trading: Computational Support for Individual and Collaborative Sense-Making Activities*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, June 2004.

[12] Axel Killian. Defining digital space through a visual language. Master's thesis, Massachusetts Institute of Technology, 2000.

[13] M. C. Lin and S. Gottschalk. Collision detection between geometric models: a survey. In *Proceedings of IMA Conference on Mathematics of Surfaces*, pages 37–56, 1998.

[14] Loom. http://www.isi.edus/isd/LOOM/LOOM-HOME.html.

[15] Map.net., 2001. http://www.cybergeography.org/atlas/map_net_2d_b_large.gif.

[16] Thinkmap. http://www.thinkmap.com.

[17] Edward Tufte. *The Visual Display of Quantitative Information*. Graphics Press, Cheshire, CT, 1983.

[18] Websom. http://websom.hut.fi/websom/.

[19] Patrick Winston. The human intelligence enterprise - why i am optimistic. http://www.ai.mit.edu/people/phw/optimism.html.

[20] Patrick Winston. *Aritificial Intelligence*. Addison-Wesley Pub. Co, 3rd edition, January 1992.

[21] Patrick Winston and Sundar Narasimhan. *On to Java: Fast Travel on the Natural Path to Java Essentials and Effective Programming Practices*. Pearson Addison Wesley, 3rd edition, January 2001.