8-14-2007

Generating User-centric Dynamic and Adaptable Knowledge Models for World Wide Web

Li Lei

Notice to Borrowers

The author of this dissertation is:

Name:	*Lei Li*
Address:	*D. Abbott Tuner College of Business*
	4225 University Dr.
	Columbus, GA 31907

The director of this dissertation is:

Name:	*Vijay Vaishnavi, PhD, IEEE fellow.*
Department:	*Department of Computer Information Systems*
Department Address:	*9th Floor, 35 Broad Street, Atlanta, GA 30302-4015, U.S.A.*

Name of User	**Address**	**Date**

GENERATING USER-CENTRIC DYNAMIC AND ADAPTABLE KNOWLEDGE MODELS FOR WORLD WIDE WEB

By

LEI LI

A Dissertation Submitted in Partial Fulfillment of the Requirements for the Degree
of
Doctor of Philosophy
in the Robinson College of Business
of
Georgia State University

GEORGIA STATE UNIVERSITY
ROBINSON COLLEGE OF BUSINESS
2007

ACCEPTANCE

This dissertation was prepared under the direction of the candidate's Dissertation Committee. It has been approved and accepted by all members of that committee, and it has been accepted in partial fulfillment of the requirements for the degree of Doctor in Philosophy in Business Administration in the Robinson College of Business of Georgia State University.

Dean: H. Fenwick Huss
Robinson College of Business

Dissertation Committee:

Chair: Dr. Vijay K. Vaishnavi

Dr. Richard Baskerville

Dr. Mark Keil

Dr. Yanqing Zhang

Dr. David Washburn

Acknowledgement

First of all, I would like to offer sincere thanks my advisor, Dr. Vijay Vaishnavi. Starting from the formation of my research idea, to the development of my research approach and prototype, and to the write-up of my dissertation, Dr. Vaishnavi constantly gave me valuable guidance and advice. Dr. Vaishnavi showed tremendous patience by his timely responses to my countless emails and weekend phone calls, and his constructive reviews of numerous proposal and dissertation drafts. Without his direction, I would not have accomplished all that I have to date. From Dr. Vaishnavi, I not only learned how to conduct research, but also how to be a scholar.

I owe great amount of gratitude to my committee members, Dr. Mark Keil, Dr. Richard Baskerville, Dr. Yanqing Zhang, and Dr. David Washburn. I really appreciate the time and efforts they spent on reading my proposal and dissertation drafts, and providing insightful feedback.

I am very greatful to Mr. Art Vandenberg, my unofficial committee member. I really appreciate his support, guidance, and encouragement on the development of the research prototype. I also want to thank other members in the project team, Jack Zheng, Victor Bolet, Vidya Rangaswamy, Anish Shindore, Nicole Geiger and Milan Pandya, for their help in implementation of the research prototype.

I'd like to thank Georgia State University Brain and Behavior program. Because of the program's generous financial support, I can concentrate on my dissertation work and my family can live a decent life. Thank you very much.

I want to thank my fellow doctoral students, Adriane Randolph, Lan Cao, Peng Xu, Chad Anderson, Jack Zheng, Yide Sheng, Chongwoo Park, Ricardo Checci and Yi Ding. I thank them very much for their help, suggestions, and encouragement during this long and difficult process.

My heartfelt thanks go to my parents. They have been my consistent support and resting harbour in the past 30 plus years. Although I am thousands miles away, I can still strongly feel their care and love in our weekly and hour-long phone calls. They are the best mom and dad in the world!

To my lovely daughter and son: Joy and Andrew. They are my ultimate source of happiness and love. They are the reason that I study hard and work hard. I thank them, my little angels!

Last, not the least, I want to thank my wife, Rong. I thank her for her true love, her selfless support, and from-bottom-of –the-heart understanding! No one besides me can understand how difficult it was to take care of my family, support me to pursue my doctoral study, and get a Ph D of her own. I'm so proud of her!

Table of Contents

List of Figures ix
List of Tables x
Abstract xi
Chapter 1: Introduction 1
1.1 Research Problem & Motivation 1
1.2 Research Questions 3
1.3 Research Scope 6
1.4. Research Methodology 7
1.4.1 Research Paradigms in IS Field 7
1.4.2 Design Science Research Theories 8
1.4.3 Design Science Research Methodology 10
1.5 Dissertation Overview 14
Chapter 2: Suggestion via Literature Review 15
2.1 Web User Adaptation System Framework 15
2.1.1 User Adaptation in Web Information Systems 17
2.2 Web-Based Information Management using Clustering Techniques 21
2.2.1 Clustering Techniques and Self Organizing Maps 23
2.2.2 Web Document Clustering using Self-Organizing Maps 28
2.3 Users in Web Information Clustering Systems 30
2.4 Research Findings 31
Chapter 3: Development - Research Approach 33
3.1 The Proposed Research Approach 33
3.2 User Interface Component 35
3.3 Web-based Resources Identification Component 37
3.4 Model Building Component 37
3.5 Research Findings 39
Chapter 4: Development – Research Prototype 41
4.1 Prototype System Architecture 41
4.2 Prototype Development 43
4.2.1 Implementation of Keyword Extractor 44
4.2.2 Implementation of SOM Clustering Algorithm 46
4.2.3 Implementation of Interactive User Interface 46
4.2.4 Implementation of Genetic Algorithm & Grid 50
4.3 Integration of SOM, Genetic Algorithm and Grid 51
4.3.1 Genetic Algorithm and SOM 51
4.3.2 GA Operations Configuration 53
4.3.3 GA Parameter Selection and Execution 55
4.3.4 Grid and GA Enabled SOM Clustering 56
4.4 Research Findings 59
Chapter 5: Evaluation - Experiment Design & Results 61
5.1 General Experiment Design 61
5.1.1 Evaluation methodology and Data Selection 61
5.1.2 Justification of Using Simulation-based Experiment 62
5.1.3 Experiments Overview 64
5. 2 Experiment Data Selection 65

5.2.1 Data Source Selection 65
5.2.2 Selection of the Experiment Data 68
5.3 Experiment Evaluation Metrics 70
5.4 Experiment Hardware and Software Settings 72
5.5 Research Hypotheses 73
5.6 Experiment 1 Design & Result 74
5.7 Experiment 2 Design and Results 78
5.7.1 Experiment Results and Analysis 83
5.8 Experiment 3 Design and Results 91
5.9 Research Findings 92
Chapter 6: Conclusion 94
6. 1 Summary of Research Efforts 94
6.2. Research Contributions 95
6.2.1 Research Findings in suggestion phase 95
6.2.2 Research Findings in the development phase 96
6.2.3 Research Findings in Evaluation Phase 97
6.2.4. Other contributions 98
6.3 Research Limitations 98
6.4 Future Research 99
Reference 101
Appendix A Training and Testing Dataset 106
Appendix A.1 Training Dataset 106
Appendix A.2 Testing Dataset 107
Appendix B. Semantic Facilitator TM SM V2.0 Web Crawler Source Code 108
Appendix B.1 Web Crawler Source Code – ExtractHtmlLinks.java 108
Appendix B.2 Web Crawler Source Code – GenerateWords.java 114
Appendix B.3 Web Crawler Source Code – URLtokens.java 120
Appendix C Semantic Facilitator TM SM V2.0 Visualization Component Source Code 125
Appendix C.1 Visualization Component – SOMConverter.java 125
Appendix C.2 Visualization Component – SOMVisualizer.java 132
Appendix D Semantic Facilitator TM SM V2.0 GA & Grid Component Source Code 143
Appendix D.1 GA & Grid Component Source Code – gacontroller.c 143
Appendix D.2 GA & Grid Component – CALCULATE_METRICS.java 148
Appendix D.3 GA & Grid Source Code – GAPopulation.c 157
Appendix D.4 GA & Grid Source Code – GAGenome.c 179

List of Figures

Figure 1.1 The General Methodology of Design Research, taken from (Vaishnavi and Kuechler 2004)........ 11
Figure 1.2 Mapping Design Science General Methodology Processes to Research Phases 12
Figure 2.2 Web User Adaptation Systems Framework 16
Figure 2.2 Structure of Adaptive Software Systems , taken from Brusilovsky and Maybury 2002........ 18
Figure 2.3 Dimensions for Adaptable and Adaptive Systems........ 19
Figure 2.4 Comparisons of Adaptable Systems and Adaptive Systems 20
Figure 2.5 Hierarchy of Clustering Algorithms, taken from Jain, Murty et al. 1999 24
Figure 2.6 Kohonen SOM Network Topology 25
Figure 3.1 Dynamic & Adaptable knowledge Model for Web-based Resources........ 33
Figure 3.2 Research Approach for Dynamic and Adaptable Knowledge Models........ 36
Figure 4.1 Semantic Facilitator TM SM V2.0 System Architecture........ 42
Figure 4.2 Semantic Facilitator V2.0 Start Page........ 44
Figure 4.3 Keywords Extractor Program Flow........ 45
Figure 4.4 Semantic Facilitator TM SM System Architecture 46
Figure 4.5 A Sample SOM Map of Semantic Facilitator TM SM V2.0........ 48
Figure 4.6 Pseudo Code for Visualization Component 49
Figure4.9 Semantic Facilitator TM SM V2.0-GA Component Program Flow 57
Figure 4.10 Pseudo Code for Applying Grid to GA........ 58
Figure 5.1 Categorization Perspectives from NDG Website........ 66
Figure 5.2 A Sample Web Page on SfN NDG Web Site. 67
Figure 5.3 Experiment 1 SOM Map 77
Figure 5.4 SOM Map for System Run without User Feedback........ 84
Figure 5.5 SOM Map from System Run with 10% User Feedback........ 85
Figure 5.6 SOM Map from System Run with 20% User Feedback........ 86
Figure 5.7 SOM Map from System Run with 30% User Feedback........ 87
Figure 5.8 Experiment 2 – F-measure Values 89

List of Tables

Table 1.1 Eight Components of an Information Systems Design Theory, taken from (Gregor and Jones 2007) 9
Table 1.2 Components of a Design Theory for This Dissertation10Table 2.1 Comparisons of User Adaptable System and User Adaptive System 20
Table 2.2 Web Document Clustering Using Self-Organizing Map 30
Table 5.1 Selected Database Web Pages from NDG Web Site 69
Table 5.2 Experiment 1 Genetic Algorithm Parameter Values 76
Table 5.3 Experiment 1 SOM Parameter Values Ranges 76
Table 5.4 Experiment 1 Result for Training Dataset 77
Table 5.5 Experiment 1 Results for Testing Dataset 78
Table 5.6 Training Dataset Categorization from NDG Website 79
Table 5.7 Testing Dataset Categorization from NDG Website 80
Table 5.8 User Selected Web Pages for Feedback 82
Table 5.9 Experiment 2 Results 90
Table 5.10 Experiment 3 Results 92

Abstract

GENERATING USER-CENTRIC DYNAMIC AND ADAPTABLE KNOWLEDGE MODELS FOR WORLD WIDE WEB

By

LEI LI

JUNE, 2007

Committee Chair: Dr. Vijay Vaishnavi

Major Department: Computer Information Systems

In the current Internet age, more and more people, organizations, and businesses access the web to share and search for information. A web-based resource is often organized and presented based on its knowledge models (categorization structures). The static and inflexible knowledge models of web-based resources have become a major challenge for web users to successfully use and understand the information on the web.

In this dissertation, I propose a research approach to generate *user-centric dynamic* and *adaptable* knowledge models for web-based resources. The *user-centric* feature means that a knowledge model is created based on a web user specified perspective for a web resource and that the user can provide feedback on the model building process. The *dynamic* feature means the knowledge models are built on the fly. The *adaptable* feature means the web user can have control of the user adaptation process by specifying his or her perspective for the web resource of interest.

In this study, I apply a design science paradigm and follow the General Design Cycle [1] (Vaishnavi and Kuechler 2004) during the course of research. A research prototype, Semantic Facilitator $^{\text{TM SM}}$ V2.0, has been implemented based on the proposed approach. A simulation-based experimentation[2] is used to evaluate the research prototype. The experimental results show that the proposed research approach can effectively and efficiently create knowledge models on the fly based on a web user preferred perspective for the web resource. I found that incorporating user feedback into the modeling building process can greatly improve the quality of the knowledge models. At the end of the dissertation, I discuss the limitations and future directions of this research.

[1] The General Design Cycle is a part of the general methodology of design science (Vaishnavi, V. and W. Kuechler. 2004, last updated January 18, 2006. Design Research in Information Systems. from http://www.isworld.org/Researchdesign/drisISworld.htm . It has 5 sequential and iterative design processes: *awareness of problem, suggestion, development, evaluation, and conclusion*.

[2] A simulated domain expert was used in the experiments. This is due to two considerations. 1). A simulated user enables effective and economical testing of the research approach. 2) Using human expert could confound the evaluation process. This is discussed in detail in the general experiment design section of chapter 5.

Chapter 1: Introduction

In this chapter, I introduce the research problem and research questions for this dissertation. I clearly define the research scope and briefly discuss the design research methodology that is used throughout the research life cycle.

1.1 Research Problem & Motivation

The number of web sites and web pages on the World Wide Web continues to grow exponentially as more and more people, organizations, and businesses rely on the Internet to share and search for information. In this dissertation, such web sites (e.g., web portals) or sets of web pages are referred to as *web-based resources*. These web-based resources are often organized and presented in a hierarchical structure based on certain categorization schemas. For example, the academic units of a university web site are organized using college/department/faculty structure. A university web site may have sub-pages for each college; each college site may have sub-pages for each department; and each department may have sub-pages for each faculty member in the department. In this dissertation, I refer to the categorization structures of a web-based resource as its respective *knowledge model*.

As the metadata for web-based resources, knowledge models greatly assist web users in browsing and searching web resources. However, these knowledge models are often static and inflexible because their creation and maintenance are typically accomplished manually offline. This process requires significant human-mediated attention, which makes it difficult and time-consuming to adapt the knowledge model to the dynamic changing web-based resources. For example, as web pages can be added, revised, deleted, and moved with a high frequency, it is difficult to make the current knowledge model keep up with the changes. More importantly, web users are often diverse in nature and different users may have different perspectives or viewpoints[3] with respect to the knowledge model of a web-based resource; however, web users are restricted to the viewpoint provided by the existing model. The motivation for dynamic knowledge models that adapt to different user perspectives can be further illustrated by the following two use cases.

1) The Southeastern Universities Research Association (SURA, www.sura.org) has a strategic initiative to develop Grid infrastructure for its 62 member universities and is compiling a knowledge base of faculty researchers and their Grid activity. An initial knowledge base of faculty web pages from 9 SURA sites is developed manually by a dedicated researcher browsing each university site, locating relevant pages, adding them to a database, and providing a new web front-end, a task filling many months. Meanwhile the underlying knowledge model as well as the perspective on the knowledge model has continued to change; faculty move, web pages change, and SURA now wants to extend the knowledge base to all 62 SURA sites. A dynamic knowledge model would be much preferred in this type of situation.

2) The Society of Neuroscience (SfN) Neuroscience Database Gateway (NDG) (http://ndg.sfn.org/) is a large web portal that hosts links to 176 neuroscience databases (the number has increased steadily from the beginning of the site and will continue to grow in the future). Those databases are categorized by a panel of neuroscientists and every new database must go through a lengthy process of submission and panel review to

[3] I use perspective and viewpoint interchangeably in this dissertation. The 'perspective/viewpoint' refers to how a web user wants the knowledge model of a web-based resource to be formed.

appear on the NDG website. Although this submission and reviewing process is rigorous, it is not very efficient because it takes a long time to add a neuroscience database to the NDG website. Consequently, the NDG website often can't reflect the latest progress in the neuroscience field.

In addition, the users' viewpoints often cannot be met by the website's current categorization schemes. For example, a neuroscientist who specializes in vision wants to check out available databases in the NDG website, but there is no vision category available. This neuroscientist can either send a request to the NDG website administrator (who may take a long time to process the request) or spend time reading the descriptions of all databases in order to perform the inquiry himself. Either way this can be a long and time-consuming process. The ability to automatically create a knowledge model for the web-based resource (web pages in the NDG website) to match the neuroscientist's specific perspective toward the site would obviously have benefited the researcher.

1.2 Research Questions

As described in the two use cases in section 1.1, dynamic and adaptable knowledge models of the underlying web-based resources would greatly facilitate web users' understanding and use of such web resources. This leads to the general research question of this dissertation:

How can the knowledge models of web-based resources be re-organized or re-structured to easily meet the specific perspective of an individual web user?

The general research question can be further elaborated as:

Given a web-based resource of user interest and a user preferred perspective towards the resource, can an information system be designed to effectively and efficiently generate appropriate knowledge models on the fly for the web-based resource?

The importance of user adaptation in web-based systems is well recognized by the research community. Many studies have been proposed to make the web and web-based systems meet the heterogeneous needs of web users through user adaptation processes. Such adaptation can be adaptable or adaptive. Systems that perform all adaptation steps autonomously on behalf of the user are called *adaptive* systems; systems that allow users to control the initiation, selection and production of adaptation mechanisms are called *adaptable* systems (Kostiainen and Lampinen 2000).

An adaptive and adaptable system can be analyzed based on three dimensions: degree of user interaction, level of adaptation, and nature of adaptation. The interaction between user and system during the adaptation process can be low or high. The adaptation can be at an individual level or at a community level. The nature of adaptation can be simple or complex.

In adaptive systems, user modeling techniques are widely used. This type of adaptation is generally with respect to the individual users, requires little user interaction, and handles adaptation ranging from simple to complex. Adaptive systems are popular because they provide adaptation effects without inconveniencing users. However, this type of system often has difficulty meeting the specific needs of an individual user, especially when the user's adaptation needs can't be modeled by the system.

In adaptable systems, adaptation is initiated by individual users and generally requires considerable user interaction. Although the user needs to spend time interacting with the system, the system can accurately reflects the user's adaptation needs. However, whereas effective adaptation requires the system to be flexible and to have sufficient internal variety (Ashby 1956), in current adaptable systems, such as Yahoo personalized homepage (http://my.yahoo.com), the available adaptation options are usually very limited and are at a community level (they are defined by the system designer). A more flexible adaptable system that can provide individual level and complex adaptation to web users is desired.

Filling in the knowledge gap in web user adaptation research, this dissertation is targeted at creating a flexible adaptable system that supports individual level and in-depth adaptation. The dissertation also investigates how knowledge models can be generated efficiently and how user feedback can be incorporated into the model building process so as to improve the quality of the knowledge models. Based on this discussion, I refine the general research question into the following three questions:

Q1) Given a web-based resource of user interest and a user preferred perspective towards the resource, can an information system be designed to effectively generate a knowledge model on the fly for the web-based resource that is adapted to the user's specific perspectives?

Q2) Can the information system be modified to utilize user feedback on the knowledge model generated (reflecting the user's perspectives) so as to improve the quality of the knowledge model generated?

Q3) Can the information system efficiently generate knowledge models for the web-based resources of user interest?

1.3 Research Scope

As articulated in the research problems and research questions sections, dynamically creating knowledge models that fit web users' specific perspectives is a very challenging and very important problem in web information management research. This dissertation aims to take an initial step in solving the issue. The scope of this research is defined as follows.

1) System development is a major component of this dissertation. In this dissertation, I not only concentrate on proposing an approach for generating flexible and adaptable knowledge models for web-based resources, but also focus on developing and implementing a working research prototype that can demonstrate the feasibility of the research approach.

2) A *user-centric* approach is emphasized in this study. Although user involvement in this research could be very complicated, I only concentrate on the objectiveness aspect of a web user (the results of user involvement in the approach can be objectively and easily measured). In this dissertation, I mainly focus on three aspects of a web user: the perspective that a web user specified for a web-based resource of interest (what is the user preferred perspective?); the feedback that a web user provided to the system (what kind of modifications did the user made to the knowledge model generated by the system?); the correctness of a knowledge model generated by user-system interaction (how correct is the knowledge model generated by the system comparing to a standard knowledge model?). The cognitive impact of the system on a web user (i.e., what has the web user learned from interacting

with the system? Is the web user satisfied with the knowledge model generated? etc.), however, was not studied in this research. Those cognitive aspects would be interesting to investigate in future studies.

3) The knowledge models (categorization structures) of a web-based resource can have multiple levels. This dissertation only concentrates on generating one level knowledge models for the sake of simplicity.

4) The web-based resources are heterogeneous in their structures: some web sites are well structured (i.e., html tags are properly used, the content of the web pages are clearly sectioned, etc.); some are semi-structured; and some are poorly structured or have no structure at all. In this dissertation, I only focus on well-structured and semi-structured web-based resources.

1.4. Research Methodology

In this dissertation, I apply a design science research paradigm to address the research questions stated in section 1.2.

1.4.1 Research Paradigms in IS Field

There are two fundamental research methodologies in Information Systems (IS) research: behavioral science and design science. Behavioral science research, that “seeks to develop and verify theories that explain or predict human or organizational behavior” (Hevner et al. 2004, p. 70), is the dominant research paradigm in the IS field. Design science research, first explicitly defined as a research paradigm in IS research by Weber (Weber 1987) in the 1980’s, has drawn more and more attention from IS researchers

(Walls et al. 1992) (March and Smith 1995) (Hevner, March et al. 2004) (Gregor and Jones 2007) (Kuechler et al. 2007).

The design science paradigm, often labeled as improvement research, “seeks to extend the boundaries of human and organizational capabilities by creating new and innovative artifacts” (Hevner, March et al. 2004, p. 70). Design science research by nature emphasizes problem-solving or performance-improvement. This dissertation aims to develop and evaluate IT artifacts (a research model and a research prototype) to solve the identified organizational problem (the inflexible and static knowledge models of web-based resources). Thus, a design science research approach is an appropriate paradigm for this dissertation.

1.4.2 Design Science Research Theories

In this research, I not only focus on developing a working system, but also take an initial step to develop a design theory for a web user adaptation system. There are a couple of seminal articles that have been published to guide IS researchers to develop design science theories. Walls et al. (Walls, Widmeyer et al. 1992) defined the information systems design theories (ISDT) for developing more effective information systems and presented an example of ISDT in the context of Executive Information Systems. Built on ISDT, Gregor and Jones (Gregor and Jones 2007) presented an anatomy of a design theory, in which eight structural components of design theories are identified (illustrated in table 1.1).

Although they grew from the same root, the eight design theory components by Gregor and Jones (Gregor and Jones 2007) offer a more systematic and useable guideline to IS researchers than the ISDT. In this dissertation, I use the 8 design theory components to guide my research effort. The instantiations of the 8 design theory components in this research is listed in table 1.2.

Table 1.1 Eight Components of an Information Systems Design Theory, taken from (Gregor and Jones 2007)

Component	**Description**
Core components	
1) Purpose and scope	"What the system is for", the set of meta-requirements or goals that specifies the type of artifact to which the theory applies and in conjunction also defines the scope, or boundaries, of the theory.
2) Constructs	Representations of the entities of interest in the theory.
3) Principle of form and function	The abstract "blueprint" or architecture that describes an IS artefact, either product or method/intervention.
4) Artifact mutability	The changes in state of the artifact anticipated in the theory, that is, what degree of artifact change is encompassed by the theory.
5) Testable propositions	Truth statements about the design theory.
6) Justificatory knowledge	The underlying knowledge or theory from the natural or social or design sciences that gives a basis and explanation for the design (kernel theories).
Additional components	
7) Principles of implementation	A description of processes for implementing the theory (either product or method) in specific contexts.
8) Expository instantiation	A physical implementation of the artifact that can assist in representing the theory both as an expository device and for purposes of testing.

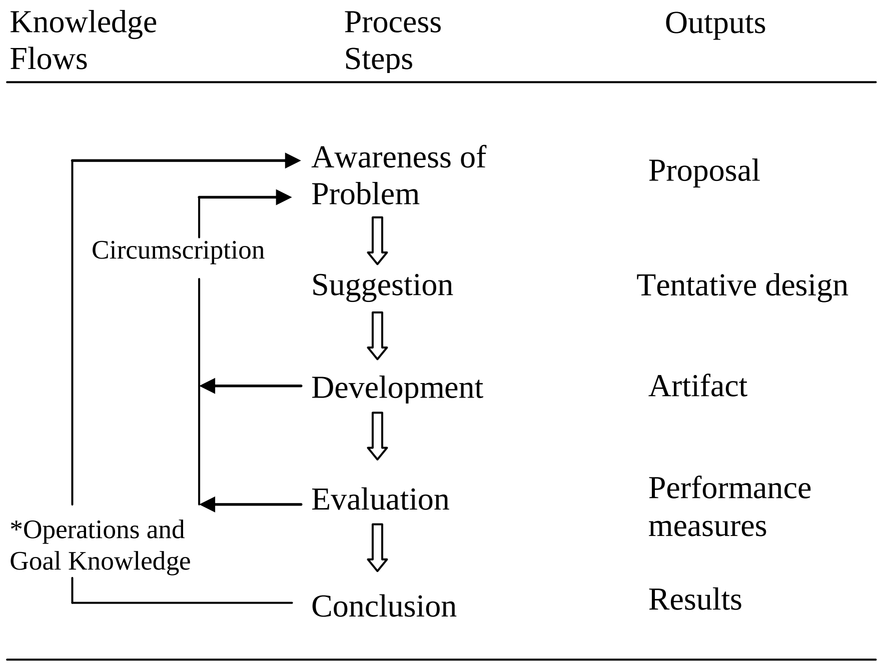
Table 1.2 Components of a Design Theory for This Dissertation

Component Type	Components Examples
1) Purpose and scope	Research motivation, research question and research scope.
2) Constructs	User input (specification and feedback), web-based resources, and the research prototype.
3) Principle of form and function	The proposed research approach (model).
4) Artifact mutability	The research prototype is designed to handle different types of data sources. The "user perspective specification, modelling building, and user review" process is designed to iterative.
5) Testable propositions	The three research hypotheses.
6) Justificatory knowledge	Proper clustering and visualization can generate appropriate knowledge model.
7) Principles of implementation	The user interface design and the integration of Clustering algorithm, genetic algorithm and Grid computing infrastructure.
8) Expository instantiation	The developed research prototype, Semantic Facilitator $^{TM\ SM}$ V2.0.

1.4.3 Design Science Research Methodology

To ensure the validity of this study, it's very important to follow proper guidelines for conducting design science research. Vaishnavi and Kuechler (Vaishnavi and Kuechler 2004) have introduced the general methodology for design research, as illustrated in figure 1.1. In this general methodology, there are 5 sequential and iterative design processes (phases) and 5 corresponding outputs. Those 5 design processes are referred to as the General Design Cycle (GDC).

The first process in GDC is the *awareness of problem* phase, in which a research defines the research problem to be solved. The output of this phase is a formal or informal proposal. In the *suggestion* phase, a solution is envisioned based on a novel configuration of either new or existing elements from the knowledge base. The output of this phase is a tentative design. The tentative design is implemented in the *development* phase and an artifact is created as the output of this phase. Next in the *evaluation* phase, the artifact is evaluated based on the criteria implicitly or explicitly stated in the *awareness of problem* phase. The output of this phase is a set of performance measures. The last phase is the *conclusion* phase, where the results of the research effort are written up and future research is discussed. The *suggestion*, *development*, and *evaluation* phases can be iteratively repeated during the course of the design, as indicated by the *circumscription* arrow in figure 1.1.



***** An operational principle can be defined as "any technique or frame of reference about a class of artifacts or its characteristics that facilitates creation, manipulation and modification of artifactual forms" (Dasgupta 1996; Purao 2002)

Figure 1.1 The General Methodology of Design Research, taken from (Vaishnavi and Kuechler 2004)

The general methodology of design research (GMDR) is simple to understand and provides guidelines for scholars who want to conduct design research in the IS field. In this dissertation, I followed the directions of the general methodology of design science and used the General Design Cycle (GDC) to carry out the research. The mapping of the General Design Cycle processes to different research phases in this dissertation is illustrated in figure 1.2.

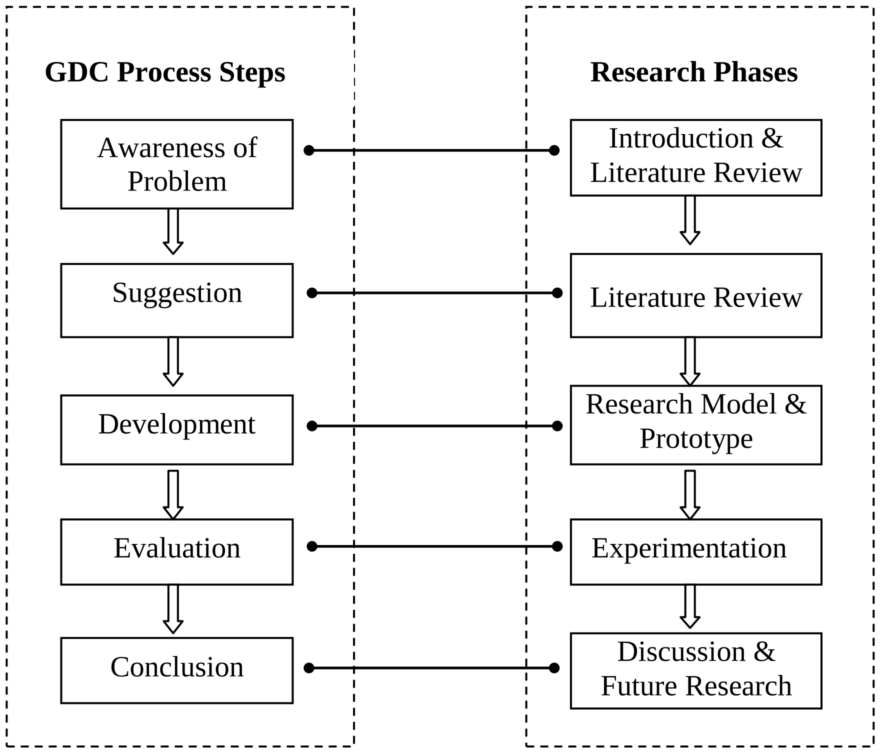


Figure 1.2 Mapping Design Science General Methodology Processes to Research Phases

Awareness of Problem The first step of the General Design Cycle (GDC) is the *awareness of problem* process. In this phase, I carefully evaluated how knowledge models are created in web information system and analyzed different types of web user adaptation systems. The research problem and research questions were identified in this phase.

Suggestion After the research problem was defined, I conducted the *suggestion* phase to generate a tentative design. To leverage existing knowledge, I performed a thorough literature review in related research fields. I developed a research framework to analyze different web user adaptation systems. I carefully examined how knowledge models are created in web information systems, in particular, how the clustering techniques are used in the model building process. I envisioned how the existing clustering approach can be improved by integrating other techniques. Through the *suggestion* step, I obtained a general idea about what the solution to the research question should be.

Development The *development* phase is where the actual design takes place and where creativity is required. Equipped with an identified research problem and a tentative design, I innovatively synthesized existing knowledge to propose a research model for generating dynamic and user-centric knowledge models. This research model is one of the artifacts created by this study. I further designed and implemented a working research prototype, Semantic Facilitator $^{\text{TM SM}}$ V2.0. The research prototype is another artifact created in the *development* phase. Those IT artifacts are the most important research outputs of design research (Hevner, March et al. 2004).

Evaluation After the IT artifact (research prototype) is constructed, the research must evaluate how well the artifact works (Hevner, March et al. 2004). The evaluation methods can be observational, analytical, experimental, testing, and/or descriptive (Hevner, March et al. 2004). In this dissertation, I chose controlled experiment and simulation based experiment as main evaluation methods because of their high internal validity (Whitley 1996). Whereas behavior research focuses on testing research hypotheses based on theory, the design research evaluation is to test the utility of the artifact(s) (how well the artifact works) (Vaishnavi and Kuechler 2004). Thus, the constructed artifact (research prototype) was evaluated according to the research questions identified in the *awareness of problem* phase.

Conclusion The conclusion phase is the “finale of a specific research effort”. In this phase, I summarized the research efforts and discussed the future research directions.

1.5 Dissertation Overview

In chapter 1, I discuss the *awareness of problem* phase by identifying the research problem and the research questions. Chapter 2 describes the *suggestion* phase of the General Design Cycle. I propose a research framework for analyzing web user adaptation systems and review the literature on web-based information management using clustering techniques, and the user’s role in web adaptation systems. In chapter 3, I synthesized the findings from the *awareness of problem* and the *suggestion* phases to propose a research approach for user-centric dynamic and adaptable knowledge models. In chapter 4, I continue on the *development* phase and discuss the research prototype development in detail. In chapter 5, I perform the *evaluation* phase and describe the experimentation design and the experiment results. In chapter 6, I summarize the research efforts of this dissertation and discuss the research contributions. The limitations of the research approach and future directions of research are also discussed in Chapter6.

Chapter 2: Suggestion via Literature Review

In this chapter, I describe the *suggestion* phase of the General Design Cycle. I first introduce a research framework for analyzing web user adaptation system. I then review two types of web user adaptation systems: adaptive web systems such as adaptive hypermedia and adaptive web; and adaptable web systems. I evaluate research progress in building knowledge models in web information systems and discuss in depth one of the enabling techniques for model building, clustering technique. Finally, I analyze the role of the user in the web information adaptation systems and summarize the research findings in this phase.

2.1 Web User Adaptation System Framework

With the proliferation of computers and the Internet, more and more web-based information systems are built to assist people and organizations to utilize information on the web. It becomes increasingly important that web information systems can adapt themselves to different user perspectives and the dynamic web environment. In this dissertation, such information systems are called as web user adaptation systems.

Figure 2.1 illustrates a general framework for web information system adaptation. There are 3 important entities in the framework: web-based resources, web information systems, and web user perspectives.

A web-based resource contains the information/knowledge sources of user interest. A web information system can treat the web-based resource as static or dynamic. However, the web is dynamic in nature. Web site or web pages can be added, removed or modified by web masters freely. A successful web information system should therefore be able to adapt to the dynamic nature of the web environment (as indicated by a unidirectional arrow in Figure 2.1).

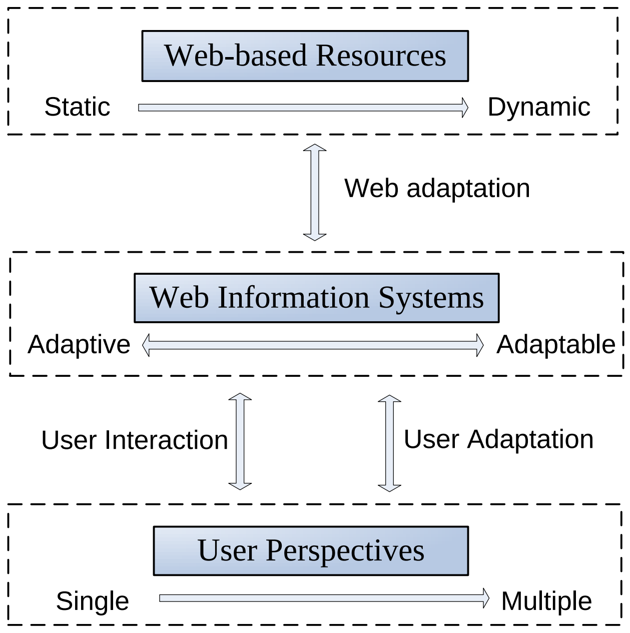


Figure 2.1 Web User Adaptation Systems Framework

Web users are the individuals who use web information systems to explore the web-based resources of their interest. Since web users are greatly diverse in their backgrounds and needs, different users tend to have different perspectives on the same web resource. In this sense, a user's perspective can range from a single perspective to multiple or even evolving perspectives. A good web information system needs to reflect the heterogeneous (multiple) needs of web users (indicated by a unidirectional arrow in Figure 2.1). Another role of a web user is to interact with the information system. By providing feedback, the information system can better understand the user's perspective and be more effective in assisting the user with accomplishing his/her tasks.

Web information systems are computer hardware and software systems that help web users better use or explore the information contained in web resources. The adaptation in web information systems can be divided into two types: adaptive systems and adaptable systems. Systems that perform all adaptation steps autonomously on behalf of the user are called adaptive systems; systems that allow users to control the initiation, selection and production of adaptation mechanisms are called adaptable systems (Kobsa et al. 2001). The adaptation of web systems is indicated by a bi-directional arrow in Figure 2.1.

In summary, there are three dimensions to evaluate a web information system: 1) How does the information system handle a web-based resource? 2) What type of adaptation approach does the system take? 3) How does the system respond to multiple or even evolving user perspectives? A good web information adaptation system should be able to handle dynamic web-based resources and multiple user perspectives. The type of adaptation chosen by an information system depends on the problem domain, user demands etc. This is discussed in detail as follows.

2.1.1 User Adaptation in Web Information Systems

The user adaptation in web information systems can be adaptive or adaptable.

The information systems that perform all adaptation steps autonomously are called user adaptive systems (Kobsa et al. 2001). For instance, if a user bought a book from Amazon (www.amazon.com), the system would automatically generate a list of related books for the user's possible additional purchases. Adaptive hypermedia and adaptive web belong to the category of user adaptive software systems (Schneider-Hufschmidt et al. 1993) (Brusilovsky and Maybury 2002). This area of research and development focuses on tailoring software systems to different users by applying user modeling techniques. Both implicit data (e.g., user click stream activity, server logs) and explicit data (e.g., user profiles) are collected and analyzed to build user models (Brusilovsky and Maybury 2002). Figure 2.2 (taken from Brusilovsky and Maybury 2002) shows the structure of adaptive software systems.

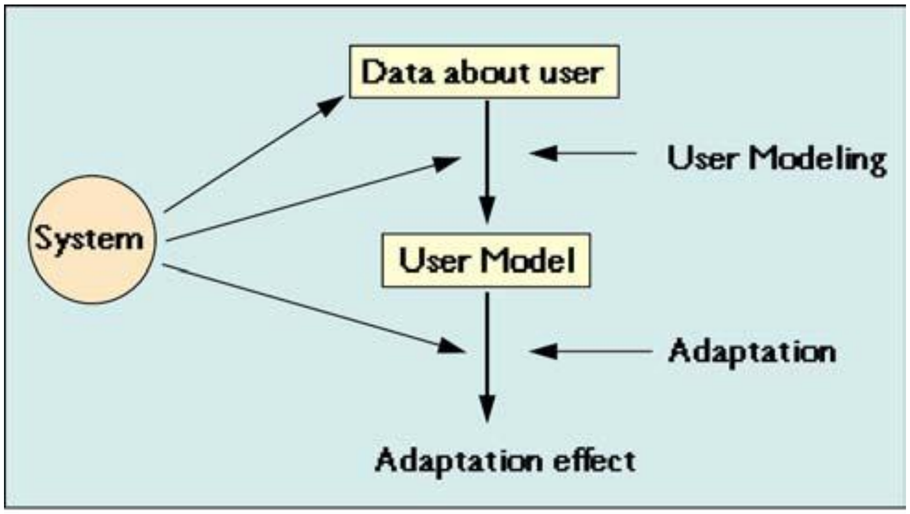


Figure 2.2 Structure of Adaptive Software Systems , taken from Brusilovsky and Maybury 2002

The information systems that allow users to control the initiation, selection, and production of adaptation are called adaptable systems. (Kobsa, Koenemann et al. 2001) For example, My Yahoo (http://my.yahoo.com) lets web users customize their Yahoo homepage based on a set of options.

Adaptive and adaptable systems can be further characterized by three dimensions: degree of user interaction, type of adaptation, and nature of adaptation (illustrated in Figure 2.3).

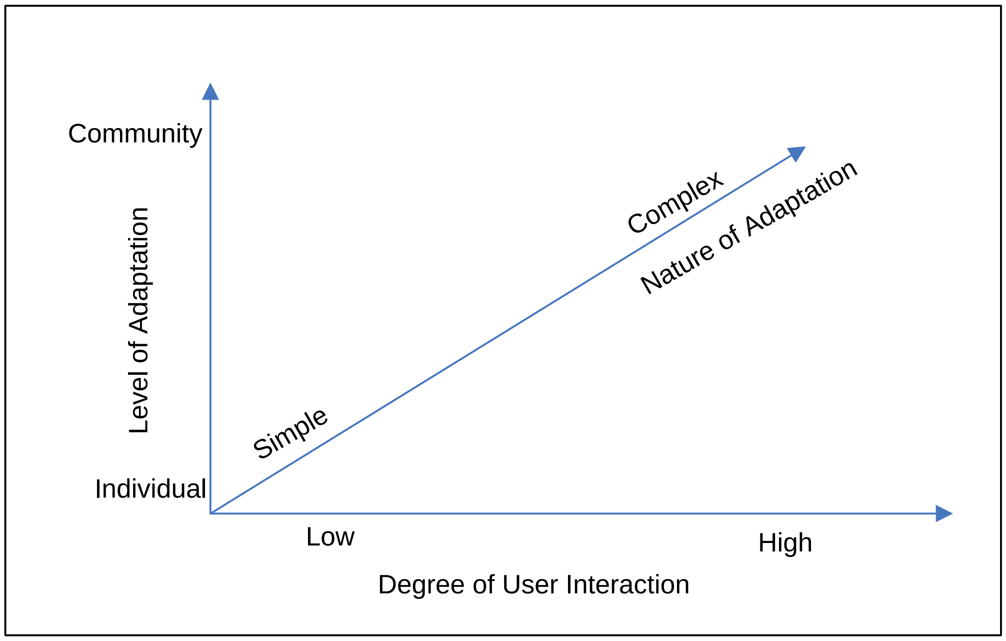


Figure 2.3 Dimensions for Adaptable and Adaptive Systems

The level of adaptation dimension concerns the number of users involved in the adaptation process, ranging from individual level to community level. In other words, the adaptation could be targeted for an individual user or for a group of users. The degree of user interaction dimension measures the degree of interaction between a user and an information system during the adaptation process, ranging from low to high. The nature of adaptation is about the complexity of the adaptation, ranging from simple to complex. In this dissertation, a simple adaptation means that an information system can easily adapt to the user requirements: the adaptation process is generally pre-defined or the adaptation can be performed by following a series of simple steps. A complex adaptation refers to the case where adaptation is done on the fly or requires a complicated computing procedure to adapt to the user requirements. An analysis of user adaptable systems and user adaptive systems, based on the dimensions shown in Figure 2.3, is summarized in Table 2.1 and illustrated in figure 2.4.

Table 2.1 Comparisons of User Adaptable System and User Adaptive System

Type of Adaptation	**Level of Adaptation**	**Degree of User Interaction**	**Nature of Adaptation**	**Examples**
Adaptive Systems	individual level/	Low	simple~ complex	(Brusilovsky and Maybury 2002)
Adaptable Systems	community level	High	simple	Yahoo Personalized Homepage
Adaptable Systems	individual level	High	complex	not available (focus of the dissertation)

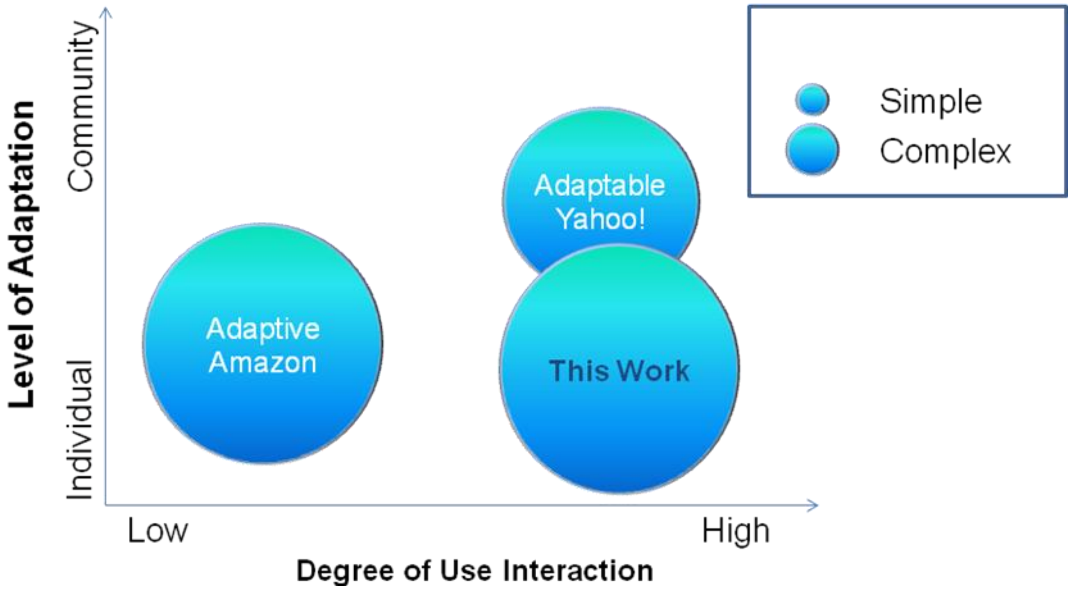


Figure 2.4 Comparisons of Adaptable Systems and Adaptive Systems

In adaptive systems, user modeling techniques are widely used. Such adaptation is generally with respect to the individual users, requires little user interaction, and can provide complex adaptation (e.g., Brusilovsky and Maybury 2002) . Adaptive systems provide adaptation effects without inconveniencing web users. However, such systems often have difficulty in satisfying the specific needs of an individual user, especially when the adaptation needs can't be modeled by the system.

In adaptable systems, adaptation is initiated by individual users and requires considerable user interaction. Although a user needs to spend times interacting with an information system, the system can accurately reflect the user's adaptation needs. However, whereas effective adaptation requires the system to be flexible and to have sufficient internal variety (Ashby 1956), in current adaptable systems, such as Yahoo personal homepage, the adaptation options are usually very limited and determined by the system designer (for a community of users). A more flexible adaptable system is desired to provide individual level and complex adaptation to web users.

As illustrated in Table 1.1, this dissertation proposes an innovative user-centric adaptable approach, providing in-depth and complex adaptation to the individual user with a high level of user interaction. Web users need to define their preferred perspectives on the web resources of their interests and actively provide feedback about the knowledge models generated by the information system.

2.2 Web-Based Information Management using Clustering Techniques

This dissertation aims to create knowledge models for web-based resources. Such knowledge models (hierarchical structures for a web site or a set of web pages) can be considered as metadata of the web resources.

Metadata describes the schema or structure of a particular data source. As "data of data" (Roszkewicz 2004), metadata has been widely proposed as a solution for understanding data sources (Foster and Grossman 2003) to enable greater information sharing (Polydoratou and Nicholas 2001). Metadata also acts as a critical component for discovering relevance of certain data resources and has the potential to bring order to the Internet (Roszkewicz, 2004). Many web information systems have been proposed to create knowledge models/metadata for web resources using clustering techniques.

Commercial search engines such as Vivísimo (Vivisimo 2006) can cluster search results and present the results hierarchically. The Vivísimo Clustering Engine puts documents together (clusters them) based on textual similarity. It uses only the returned title and abstract for each result. The similarity between documents is based only on this raw material (the visible text of the search result). WebClust (http://www.webclust.com) is a Meta search engine. Like Vivísimo, WebClust is based on a technology called "Document clustering" that automatically organizes web documents into meaningful groups. While such Meta search engines have been quite successful, they also have limitations as their clustering algorithms are propriety and the categorization structures (knowledge models) generated by the engines are fixed: users can't customize how search results are clustered.

Many academic studies are also devoted to the information management by clustering search results. Gauch et al. (Gauch et al. 2004) adapted information navigation based on a user profile structured as a weighted concept hierarchy and so provided the users with personalized search and browsing. Ferragina and Gulli (Ferragina and Gulli 2005a) introduced a personalized search engine, called SnakeT, which is able to process search results generated from commodity search engines, and to create, on-the-fly, a hierarchy of labeled folders using the hierarchical clustering technique. The SnakeT engine demonstrated performance similar to the commercial search engine Vivísimo.

Zeng et al. (Zeng et al. 2004) conducted a study on organizing Web search results into clusters. They focused on effectively naming the clusters based on a regression model learned from human labeled training data.

One potential drawback of the above studies is that they do not address the adaptability issue. Some research such as (Gauch, Chafee et al. 2004) and (Ferragina and Gulli 2005a) offer personalized browsing and search, though they lack a mechanism to let users view the search results from different angles. In addition, these approaches don't support much interaction with the user. Although limiting user interaction could be a feature for those systems (providing transparency of function), I argue that a certain level of user interaction is necessary when I try to build a knowledge model customized to a particular user's needs.

In summary, according to the current literature, clustering (as discussed in detailed in the following section) is a good technique for organizing web information. In terms of adaptation, some approaches such as Vivísimo.com and WebClust.com don't provide the adaptation feature; other approaches such as (Gauch, Chafee et al. 2004) and (Ferragina and Gulli 2005a) are adaptive (but not adaptable) to web users. This dissertation adapts clustering along with other techniques to achieve a flexible and effective web information management system.

2.2.1 Clustering Techniques and Self Organizing Maps

Clustering analysis is a well-known approach to structure previously unknown and unclassified datasets (Nürnberger 2001). Clustering is useful when there is little prior information (e.g., statistical models) available about the data, and the decision-maker must make as few assumptions about the data as possible. It is under these restrictions that clustering technology is particularly appropriate for the exploration of interrelationships among the data points to assess their structure (Jain et al. 1999). Different approaches to clustering data can be described with the help of the hierarchy shown in Figure 2.5 (taken from Jain, Murty et al. 1999).

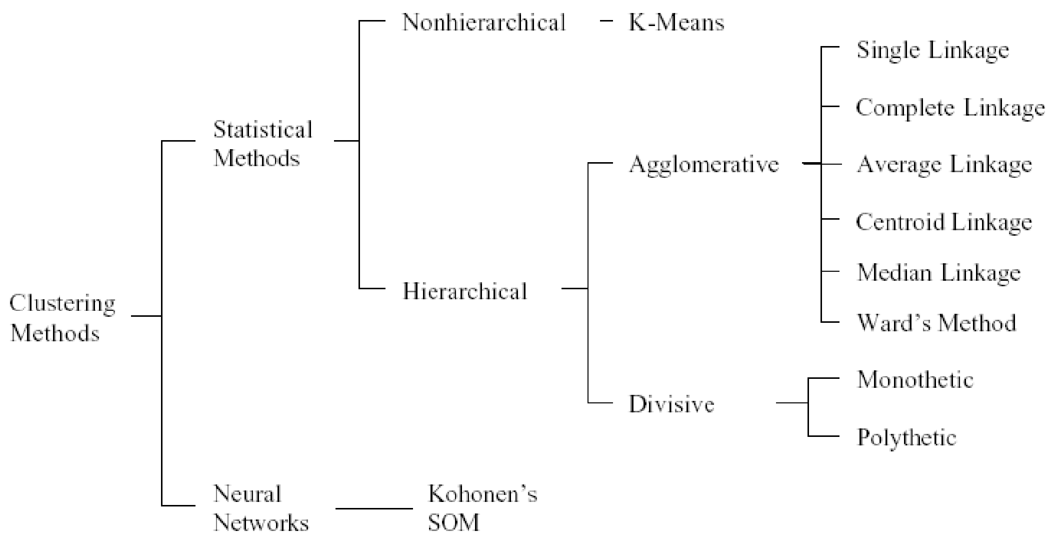


Figure 2.5 Hierarchy of Clustering Algorithms, taken from Jain, Murty et al. 1999

In this dissertation, I adapt a neural network based unsupervised clustering technique - Self-Organizing Maps (SOM) (Kohonen 1995) as the general clustering algorithm. SOM is based on the associative neural properties of the brain. The Kohonen SOM network (illustrated in figure 2.6) contains two layers of vectors: input vectors and mapping vectors usually in the shape of a two-dimensional Grid. The input vector size is equal to the number of unique features associated with the input objects. Each vector of the mapping layer has the same number of features as the input vector. Thus, the input objects and the mapping cells can be represented as vectors that contain the input features.

The mapping vectors are initialized with random numbers. Each actual input is compared with each vector on the mapping Grid. The “winning” mapping vector is defined as that with the smallest distance (e.g. Euclidean Distance) between itself and the input vector. The input thus maps to a given mapping vector. The value of the mapping vector is then adjusted to reduce the distance, and its neighboring vectors may be adjusted proportionally. In this way, the multi-dimensional (in terms of features) input vectors are mapped to a two-dimensional output grid. After all of the input is processed (usually after hundreds or thousands of repeated presentations), the result should be a spatial organization of the input data organized into clusters of similar (neighboring) regions (Roussinov and Chen 1998).

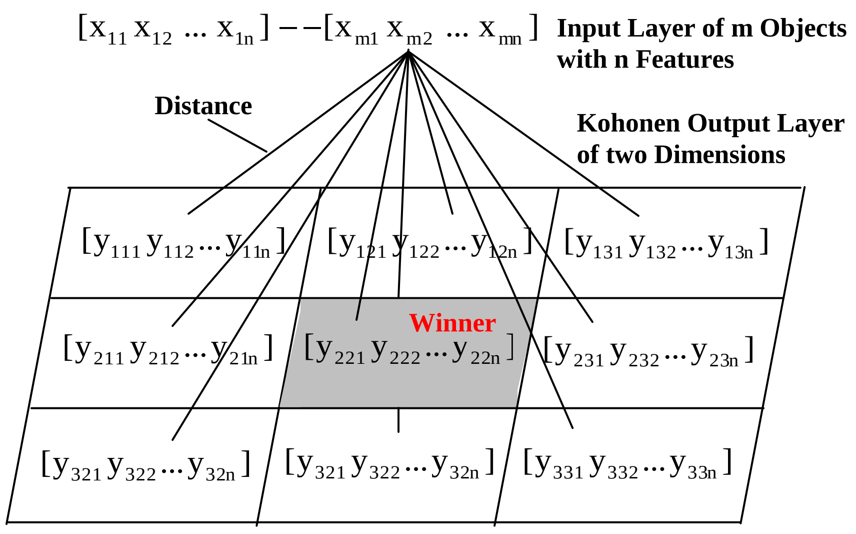

$[x_{11}\ x_{12}\ \dots\ x_{1n}] -- [x_{m1}\ x_{m2}\ \dots\ x_{mn}]$
Input Layer of m Objects with n Features
Distance
Kohonen Output Layer of two Dimensions
$[y_{111}\ y_{112} \cdots y_{11n}]$
$[y_{121}\ y_{122} \cdots y_{12n}]$
$[y_{131}\ y_{132} \cdots y_{13n}]$
Winner
$[y_{211}\ y_{212} \cdots y_{21n}]$
$[y_{221}\ y_{222} \cdots y_{22n}]$
$[y_{231}\ y_{232} \cdots y_{23n}]$
$[y_{321}\ y_{322} \cdots y_{32n}]$
$[y_{321}\ y_{322} \cdots y_{32n}]$
$[y_{331}\ y_{332} \cdots y_{33n}]$

Figure 2.6 Kohonen SOM Network Topology

SOM is very effective for visualization of high-dimensional data. It compresses information while preserving the most important topological and geometric relationships of the primary data elements on the display. It is then possible to visually identify clusters from the map. The main advantage of such a mapping is that it is possible to gain some idea of the structure of the data by observing the map, due to the *topology preserving* nature of SOM.

Mangiameli et al. (Mangiameli et al. 1996) compared SOM and seven hierarchical clustering methods experimentally and found Self-Organizing Maps (SOM) to be superior to all of the hierarchical clustering methods. Zhao and Ram compared K-means, hierarchical clustering, and SOM for clustering relational database attributes. They concluded that the three methods have similar clustering performance and that SOM is better than the other two methods in visualizing clustering results (Zhao and Ram 2004).

Clustering is well suited to our problem domain since it pertains to generating or replicating categorization structures. Rauber and Merkl (Rauber and Merkl 1999) successfully applied the SOM technique to automatically structure a document collection and to create a digital library system. In this dissertation, I adapt SOM as the clustering algorithm.

Effective SOM clustering depends on the parameter values chosen for a certain application domain (Polani and Uthmann 1993). Since users usually are not aware of the structure present in the data (that is why clustering analysis is helpful), not only is it difficult to determine what parameter values to use, but also it is difficult to say when the map has organized into a proper cluster structure (Alahahoo and Halgamuge 2000). There are but few studies conducted on how to select the best or near optimum SOM parameter values for an application domain. Liang et al. (Liang et al. 2006) proposed an approach for searching good SOM parameter values by trying out 320 preset parameter value permutations. In this dissertation, I couple a genetic algorithm with SOM to systematically search for good SOM parameter values. Introduced by John Holland during the 1970s, genetic algorithms make it possible to explore a far greater range of potential solutions to a problem than do conventional methods (Holland 1992). Compared to traditional search algorithms, a genetic algorithm is able to automatically acquire and accumulate the implicit knowledge about the search space during its search process, and self-adaptively control the search process through a random optimization technique. It often yields the globally optimal solution and avoids combinatorial explosion by disregarding certain parts of the search space (Wu et al. 2004). Selection of SOM parameter values is ultimately a search problem. It is thus appropriate to apply a genetic algorithm to find a good set of SOM parameter values from the vast number of possible candidates.

One potential drawback of combining genetic algorithm and SOM is that those two techniques are both computation intensive. It may take hours for the system to produce meaningful results which makes the proposed system impractical to use. In this dissertation, I propose to use Grid computing technology (Foster et al. 2002) to speed up the processing time. A Grid provides a parallel computing infrastructure that can offer tremendous computing power by utilizing numerous networked computers. In a Grid environment, a computing task is divided into many sub tasks and each sub task is assigned to a Grid node (a computer in the Grid network) for execution. The sub tasks in the Grid nodes are executed simultaneously. Then the results of the sub tasks are combined to produce the running result for the original computing task. If the sub task running time is significantly larger than the communication cost, the total running time for the original task can be greatly reduced. A genetic algorithm is perfect for taking advantage of a Grid computing infrastructure since it can be easily divided into small tasks and the results can be combined later.

2.2.2 Web Document Clustering using Self-Organizing Maps

SOM algorithms have been widely used for web documents clustering. Kaski et al. (Kaski et al. 1998) introduced the WEBSOM system based on SOM. WEBSOM organizes textual documents onto a graphical map display and provides an overview of document collections. Using SOM as its core, Rauber and Merkl (Rauber and Merkl 1999) presented a comprehensive solution for a digital library, including document clustering, labeling and visualization. This approach provided macro level personalization by allowing the user to integrate different maps together. Roussinov and Chen (Roussinov and Chen 2001) proposed an approach for interactive information seeking by clustering web query results using a SOM algorithm. A limitation of these studies is that they ignore the heterogeneous nature of the web users to some degree: the SOM clustering results are generally the same to every user even though each user may have a different perspective.

Smith and Ng (Smith and Ng 2003) developed the LOGSOM system that provides personalized web pages clustering by analyzing user navigation patterns using a SOM algorithm. Chau et al. (Chau et al. 2001) and Chen et al. (Chen et al. 2001) proposed personalized web spiders for web search. Users can customize SOM by selecting or deselecting the phases used for clustering and the clustering can be performed multiple times until user satisfaction is reached. A drawback of the above approaches is that SOM parameter values are usually determined ad hoc and such selection can have a big impact on SOM’s performance.

Table 2.2 summarizes the usage of SOM in web information management research. Whereas SOM clustering algorithms have been successfully applied to various domains, there are several aspects that haven't been fully studied in the research community.

1) User personalization of SOM. Most of the studies neglect the heterogeneity of web users. For the same set of data, different users may have different perspectives. SOM clustering should be customizable to reflect the user needs.

2) Selection of SOM parameter values. As I stated in the previous section, the SOM parameter values have a significant impact on the clustering performance. In most studies, such selection is either not addressed or done in an ad hoc manner.

3) User feedback on SOM. SOM is an unsupervised neural network algorithm and the clustering can be done automatically. Most studies use this feature as is, without modification. I argue that giving some guidance (in the form of user feedback) to an unsupervised SOM can produce more accurate and relevant clustering results.

Table 2.2 Web Document Clustering Using Self-Organizing Map

Application Domain	**User Personalization of SOM**	**SOM Parameter Selection**	**User Feedback for SOM**	**User Adaptation**	**Representative Research**
Digital Library	Macro Level. User can integrate different maps based on individual interests.	NA	No	Adaptable, simple adaptation	SOMLib Digital Library (Rauber and Merkl 1999)
Web textual documents	No	NA	No	No	WEBSOM (Mondelli et al. 1998) (Kaski, Honkela et al. 1998)
Web Server Log	No	Ad hoc	No	Adaptive	LOGSOM (Smith and Ng 2003)
Meta search engine	Yes	NA	Yes	Adaptable, simple adaptation	MetaSpider (Chau, Zeng et al. 2001) (Chen, Fan et al. 2001)
Web Query Results	No	NA	No	Adaptive	(Roussinov and Chen 2001)

Note : NA – No information available.

2.3 Users in Web Information Clustering Systems

The user is an important component of any web information clustering system. One of the goals of such systems is to facilitate the use and understanding of the given web-based resource by applying various clustering techniques. Clustering web datasets is ultimately a categorization process. The knowledge models referred to in this dissertation can be considered as a set of categories.

Categories are “groups of distinct abstract or concrete items that the cognitive system treats as equivalent for some purpose” (Markman and Ross 2003). The categories this dissertation deals with are “perceptual categories”-- a collection of concrete similar objects belonging to the same group (Ashby and Maddox 2004). Ashby and Maddox (Ashby and Maddox 2004) identified four different tasks in category learning: rule-based tasks, information-integration tasks, prototype distortion tasks, and weather prediction tasks. Clustering web documents is a combination of information integration and prototype distortion tasks. Success in those types of category learning depends on the quality and timing of the feedback from the categorization system (Ashby and Maddox 2004). Therefore, in order to facilitate user learning and achievement, the clustering system should provide timely and relevant clustering for the users.

2.4 Research Findings

Firstly, the importance of user adaptation in web information systems has been well recognized by the research community. I introduced a framework to evaluate different types of user adaptation systems. Adaptive systems have been very successful by performing adaptation autonomously. The users often have little control over the adaptation process and the adaptation sometime can’t reflect the user’s specific perspective, especially when this perspective can’t be captured by the user modeling process. Adaptable systems give the user the control and can accurately echo user needs, but current adaptable systems often can only offer low-level adaptation. This dissertation seeks a flexible system that can adapt to heterogeneous user perspective. Thus, I propose an adaptable system which can provide complex user adaptation.

Secondly, I also reviewed how knowledge models for web-based resources are created in this chapter. Clustering is one of the enabling techniques for organizing textual dataset. I chose a very successful clustering algorithm, Self-Organizing Maps (SOM), to generate knowledge models for web-based resources. However, SOM performance is sensitive to application domains. In this dissertation, I propose to use an improved SOM clustering algorithm to dynamically and systematically create knowledge models.

Lastly, web users are involved in the knowledge model building process by providing their perspectives and feedbacks. Thus, I proposed that the system needs to be responsive.

Chapter 3: Development - Research Approach

In this chapter, I perform the *development* phase of the General Design Cycle (GDC). I first present an overview of the research approach for generating user-centric dynamic and adaptable knowledge models for web-based resources. Then I discuss the different components of the research approach in detail.

3.1 The Proposed Research Approach

Equipped with the tentative design (research findings section of chapter 2) from the GDC *suggestion* phase, I propose to research and develop an approach for *user-centric, dynamic and adaptable* knowledge models (illustrated in Figure 3.1) for web-based resources of interest.

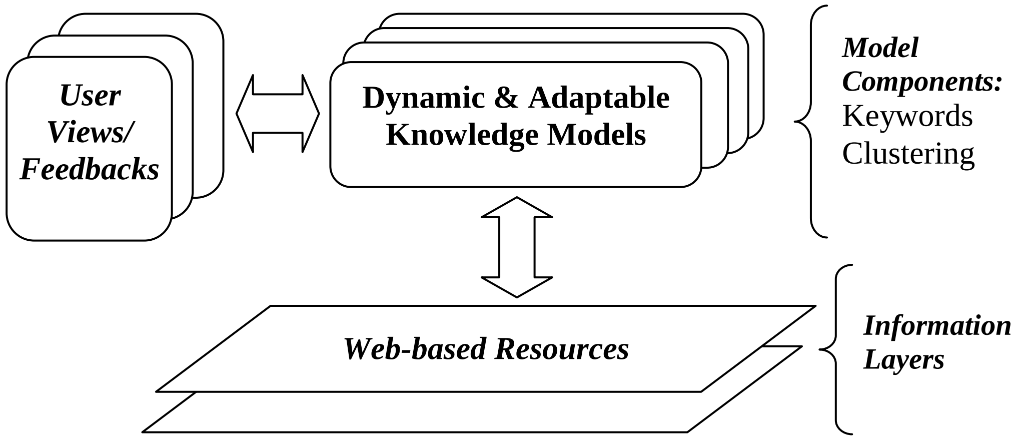


Figure 3.1 Dynamic & Adaptable knowledge Model for Web-based Resources

In the research approach, the *User-centric* feature means knowledge models are created based on user specified perspective and users' feedback can be incorporated into the model building process. For example, a web user may want to organize a set of faculty web pages by the faculty's research interest. The system (research approach) should be able to generate such a knowledge model for the user. The web user can review the created knowledge models and make some adjustments. The *Dynamic* feature means knowledge models are built on the fly so that it can always reflect the newest content of the web-based resources. The *Adaptable* feature means users have control of the adaptation process by specifying their perspective on the web resource of interest.

In the research approach, I argue that by interactive user-centric clustering and visualization, an information system can create a dynamic and adaptable knowledge model for a given web-based resource. Based on a user's preferred perspective, the proposed approach can re-organize the knowledge model (website structure) of user interest if there is an existing model, or create a new one if there is no knowledge model available, thus permitting multiple or even evolving user perspectives of the same web-based resource.

This research approach is based on the proposition that, on the one hand, appropriate clustering and visualization can re-organize knowledge models[4] (web site structures) for web-based resources to help web users identify the patterns of web resources and facilitate their understanding of this dataset. On the other hand, web users' feedback can guide the machine clustering process and generate more relevant clusters for the web users. The research approach emphasizes user-centric, in-depth (complex) adaptation for web-based resources whereas many existing research approaches and studies have been focusing on providing autonomous but inflexible adaptation to web systems.

[4] The dissertation limits itself to generating knowledge models as one-level clusters instead of multi-level clusters.

In the research approach, I adapt Self-Organizing Maps (SOM) (Kohonen et al. 2000) as the clustering algorithm to generate knowledge models. SOM is a widely used unsupervised neural network based technique for document/text clustering (Rauber and Merkl 1999) (Gauch, Chafee et al. 2004) (Ferragina and Gulli 2005b) (Smith and Ng 2003). I then aim to improve the SOM clustering performance by systematically searching optimal SOM parameter values using a genetic algorithm (Holland 1992). I also propose to reduce system processing time by using a Grid computing infrastructure (Foster and Kesselman 1999; Foster, Kesselman et al. 2002).

This research is based on the following assumptions: 1) The web-based resources of user interest are semi-structured or well-structured; 2) Only textual information of the web-based resources is used in the research approach; 3) The web-based resource of user interests has sufficient information to train the SOM algorithm.

The research approach has three components: 1) User interface component; 2) Web resources identification component, and 3) Model building component. The research approach is shown in Figure 3.2.

3.2 User Interface Component

Web users play an important role in the proposed research approach. The purpose of this dissertation is to develop a system that can satisfy varying, individual needs of web users with respect to target web-based resources. A web user should provide a profile or specification input reflecting a desired perspective or user view. A useful user profile specification can be composed of a mixture of 1) a user specification of a web site exemplifying the user's perspective, 2) pre-existing profiles, and/or 3) user selection from existing keywords or category/subject lists. The user profile specification can facilitate the web-based resource identification and model building components, guiding what information to retrieve, and what keywords to select for clustering.

A web user is also responsible for interacting with the system and providing feedback to the clustering results. Based on the user's recommendation, the system will re-cluster the dataset and present it to the user.

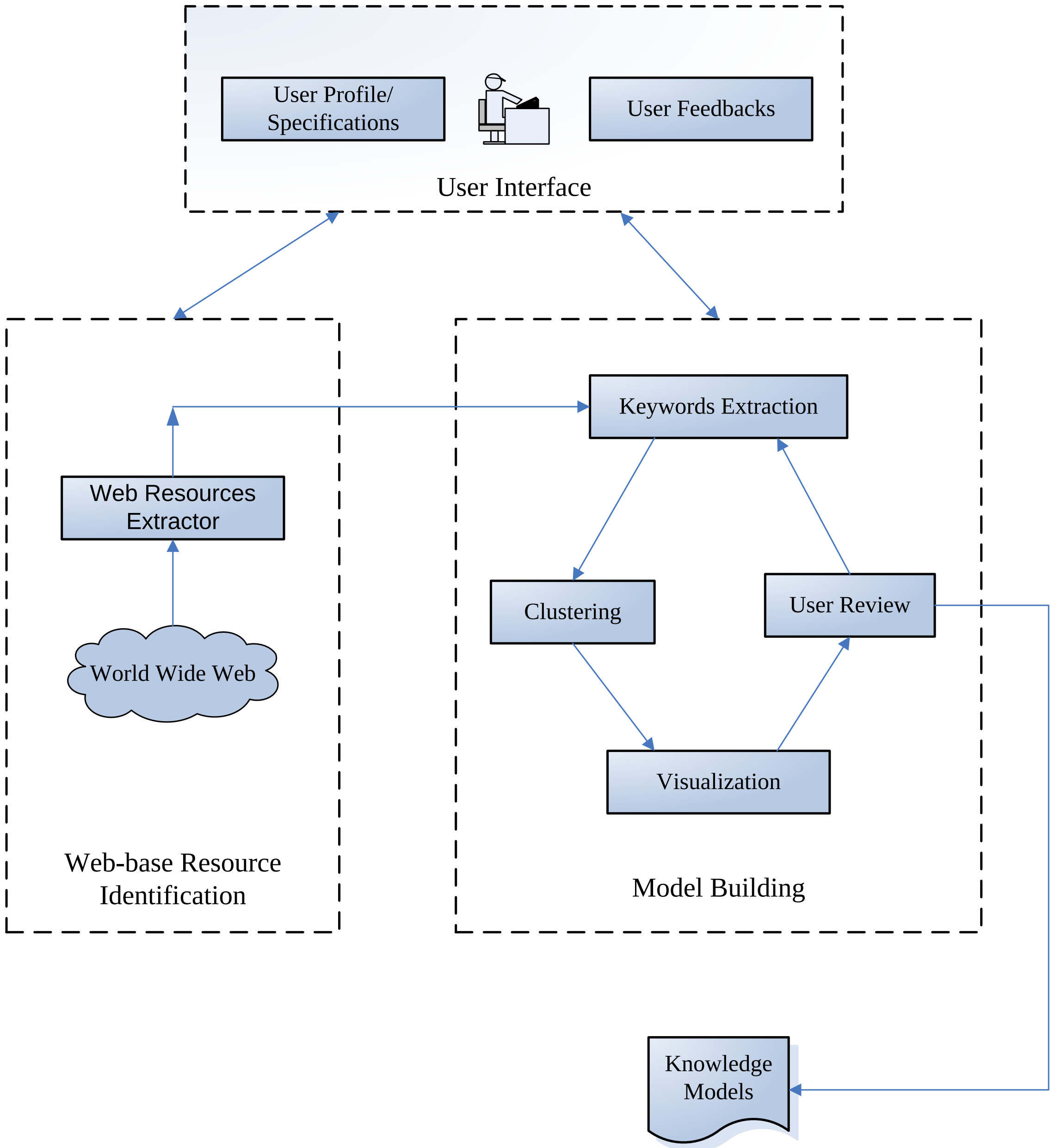


Figure 3.2 Research Approach for Dynamic and Adaptable Knowledge Models

3.3 Web-based Resources Identification Component

The web-based resources of interest are identified by the user's web-based resource specification. The user could provide the URL of a website, e.g. the department site of a university, or the user could perform a web search to retrieve a set of web pages. The contents of user specified web pages are directly extracted from the Internet and cached for further processing.

3.4 Model Building Component

The model building component is the core and most important component of the proposed research approach. It includes four iterative steps: keyword extraction, clustering, labeling, and review. First, keywords are selectively extracted from the web resources. Next the web resources are clustered based on the extracted keywords and the clustering results are visualized in a two dimensional map and presented to the user. Finally, the user reviews the clusters and may make some manual adjustments. Based on user refinement, additional iterations of keyword extraction, clustering, and visualization may be performed to make necessary modifications. The whole process could be repeated several times.

Keyword Extraction. In this step, all the text content from the identified web pages is extracted using the "View Source" feature of a web browser. Those web pages may contain links to other web pages. Theoretically the extractor (e.g., search engine) can go as deep as it is allowed to, although I only extracted information from the first page the crawler accessed for the sake of simplicity.

The extracted web page text is first filtered based on user specification. For example, a database web page from the NDG website may contain a lot of information such as description, species, database URL, project primary investigator, etc. If the user wants to categorize the databases from the perspective of database contents, only the description section of a web page is kept. By doing this, only the information related to the user's specific perspective is extracted for future clustering. This in turns makes the clustering more accurate.

The filtered web page text is then tokenized and a set of keywords are generated. The keyword list is further filtered by removing the common words such as "is," "a," etc. At the end, each web page (or URL) is associated with a set of clean keywords that describe the web page.

Clustering. In this step, the extracted keywords are used as a feature set in the clustering phase. By clustering, the input web pages are organized into related groups based on the feature set. In this dissertation, I apply neural network based Self-Organizing Maps (SOM) as the clustering algorithm. The clustering result of SOM is sensitive to the selection of its parameter values (Kohonen 1995). Liang et al.(Liang, Vaishnavi et al. 2006) researched SOM parameter values for their directory project by trying different permutations of such values. In this dissertation, I apply a genetic algorithm (GA) to discover near-optimal SOM parameter values. A Grid computing infrastructure is used to provide computing power for the system since SOM and GA are both computation-intensive.

One issue in this step is the integration of different techniques. This is discussed in details in the next chapter. The other issue is the evaluation of the knowledge model (clustering result) generated by the system. In the experiments of this dissertation, I selected a website that already organized its web pages from various perspectives following a rigorous process. Those existing categorization schemes (knowledge models) are used to evaluate the knowledge model created by the system.

Visualization Another challenge is the visualization of the clustering. SOM can generate a two dimensional map, but the map isn’t interactive. I adapted a Java based visualization toolkit to provide an interactive user interface. The user can easily review the information about the SOM map and can also be able to drag and drop an object (a node that represents a web page) on the SOM map to better facilitate modification (review) step.

Review The final component is the user evaluation of the produced model. I argue that user participation is critical to generate a user-centric knowledge model especially when the knowledge model needs to be created completely from the start. In this step, a web user reviews a generated model and makes necessary modifications. The system will then rerun the model building process based on user guidance until the model meets the user’s perspective.

3.5 Research Findings

In this chapter, equipped with tentative design (research findings) in the *suggestion* phase, I introduced a research approach for generating user-centric dynamic and adaptable knowledge models for web-based resources. I described in details the functionality of each component in the architecture. In the research approach, clustering technique, specifically Self-Organizing Maps (SOM), was chosen as the main driving engine for creating knowledge models. I proposed to integrate SOM with other enabling techniques such as genetic algorithm and Grid computing infrastructure to improve the effectiveness and efficiency of the approach.

The research approach is unique as a type of web user adaptation system because it focuses on adaptable and flexible user adaptation. Current adaptive systems have been very successful in performing adaptation autonomously but the adaptation has been done on the back end using many server side resources, and the system often can’t accurately react to a user’s specific adaptation needs, especially when such needs can’t be modeled by the user. Existing adaptable systems can accurately reflect user needs, but often offer low-level adaptation. Based on my knowledge, this study is the first research attempt to offer flexible and adaptable user adaptation to web information systems.

The research approach also provides a systematic and efficient way to create a dynamic knowledge model for a web-based resource by integrating clustering technique with a genetic algorithm and Grid technology.

The proposed approach provides a generic approach for knowledge presentation and discovery. In this dissertation, I focus on a web-based dataset, and the research approach fundamentally deals with textual data. The proposed approach could be easily adapted to other application domains, such as database domain and project portfolio management by converting the new datasets into a textual format.

Chapter 4: Development – Research Prototype

In this chapter, I continue on the *development* phase of the General Design Cycle (GDC) and describe the development of the research prototype. In order to demonstrate the feasibility of the proposed research, a research prototype, namely Semantic Facilitator ™ ℠ V2.0, is designed and implemented. In the chapter, I first introduce the system architecture of the research prototype and then discuss the different technologies used in the prototype, i.e. Self Organizing Maps (SOM), genetic algorithm (GA), Grid computing infrastructure, and the Prefuse information visualization toolkit (Prefuse 2007). Lastly, the integration of SOM, GA, and Grid computing infrastructure is presented.

4.1 Prototype System Architecture

Based on the proposed research approach, Semantic Facilitator ™ ℠ V2.0 is designed as a web-based information system that operates in a Microsoft Windows environment. The user can register and access the system remotely, but can also download the software package to install and run it locally. While Semantic Facilitator ™ ℠ V2.0 is primarily developed for handling web-based resources such as web pages, it can also process LDAP directory data or other textual datasets.

The system architecture of Semantic Facilitator ™ ℠ V2.0 is illustrated in figure 4.1. The architecture contains four modules: user interface module, web resource identification module, model building module, and data module.

User Module. The main functionality of the user module is to provide a user interface so that users can define their profile or specific adaptation needs and interact with other modules, e.g., giving feedback to the system by dragging and dropping objects among the clusters. The basic GUI user interface is developed using ASP and Servlet techniques and displayed in the web browser. Another critical issue is the visualization of the SOM clustering results. Instead of using the default two dimensional map provided by SOM packages, Semantic Facilitator $^{TM\ SM}$ V2.0 adapts an information visualization package, Prefuse, to develop an interactive interface for SOM clustering. This still produces a two dimensional map, but each web page is now displayed as a node and the users can drag and drop the nodes among the clusters freely using a computer mouse.

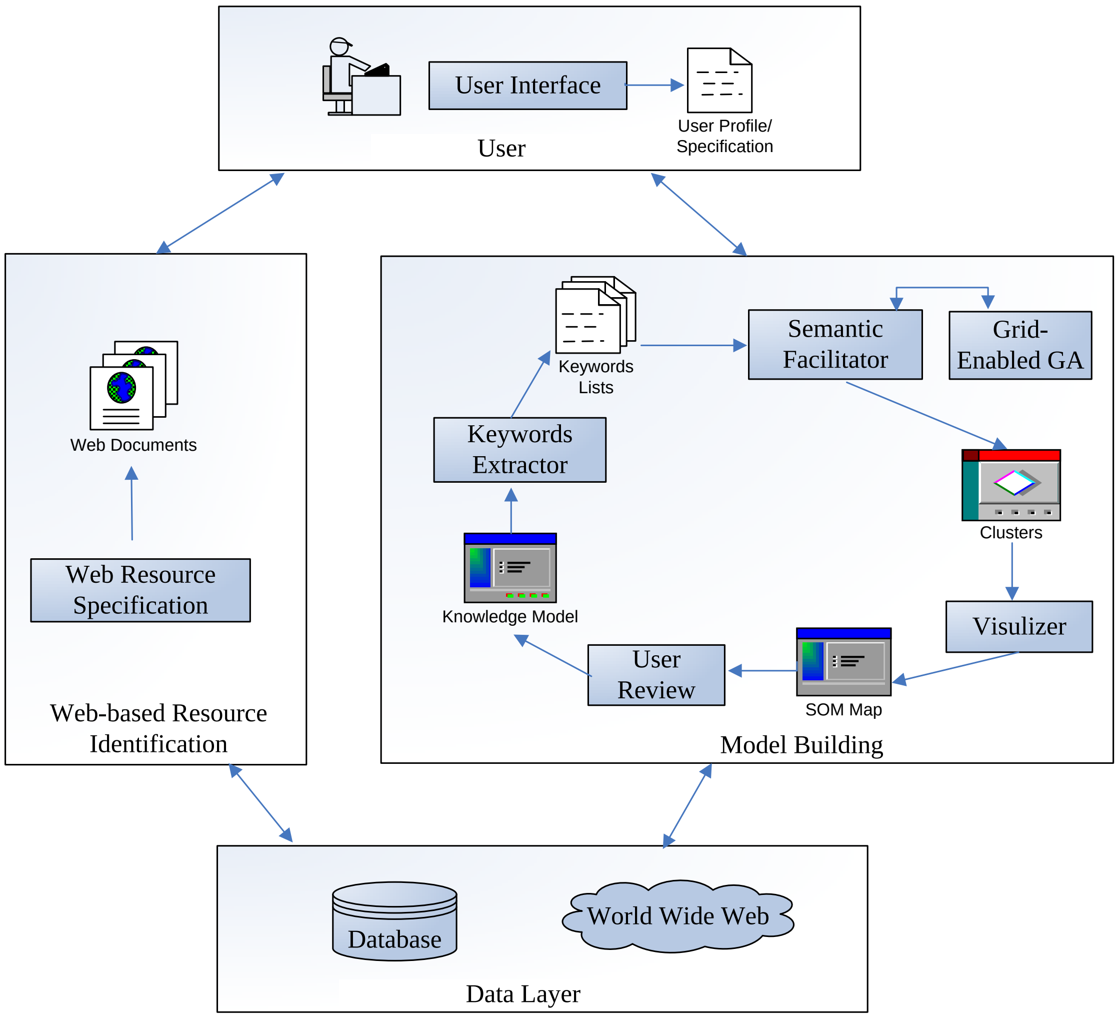


Figure 4.1 Semantic Facilitator $^{TM\ SM}$ V2.0 System Architecture

Web-based Resource Identification Module. In this module, the web users specify the web resources of interest. The web pages (information) are extracted from the Internet using a web crawler and cached in a local file folder. The web crawler was developed in-house using Java programming language.

Model Building Module. In this module, the keyword extractor reads the cached web pages. Commonly used words such as “is,” “a,” etc. are first removed from the extracted information, and then the information is further filtered based on the user profile or specification. Finally, each web page is associated with a list of keywords that are relevant to the user profile or specification. The processed web resources (information) are then clustered using Semantic Facilitator $^{\text{TM SM}}$. Semantic Facilitator $^{\text{TM SM}}$ (Vandenberg et al. 2002) is our prior research prototype that can cluster textual documents using a SOM clustering algorithm. In this dissertation, I integrate a genetic algorithm and Grid computing technique with Semantic Facilitator $^{\text{TM SM}}$ to improve its performance and use a new interactive visualization component to display the generated SOM clusters. The clusters may be further reviewed by the user and the extraction, clustering, labeling, visualization, and reviewing process can be repeated several times.

Data Module. This module contains external data sources such as World Wide Web information. The system also uses a relational database to cache retrieved web resources and extracted keyword lists for more efficient processing.

4.2 Prototype Development

For the research prototype, I used IBM Rational Software Development Platform as the main development environment. The web interfaces were developed using ASP and Servlet technology, and the keyword extractor was developed using Java programming language. The web server is deployed using IBM Web Sphere software. I also adapted available software packages for the implementation of the SOM clustering algorithm, genetic algorithm, Grid computing infrastructure, and visualization component. The beginning page of the prototype is shown in figure 4.2.

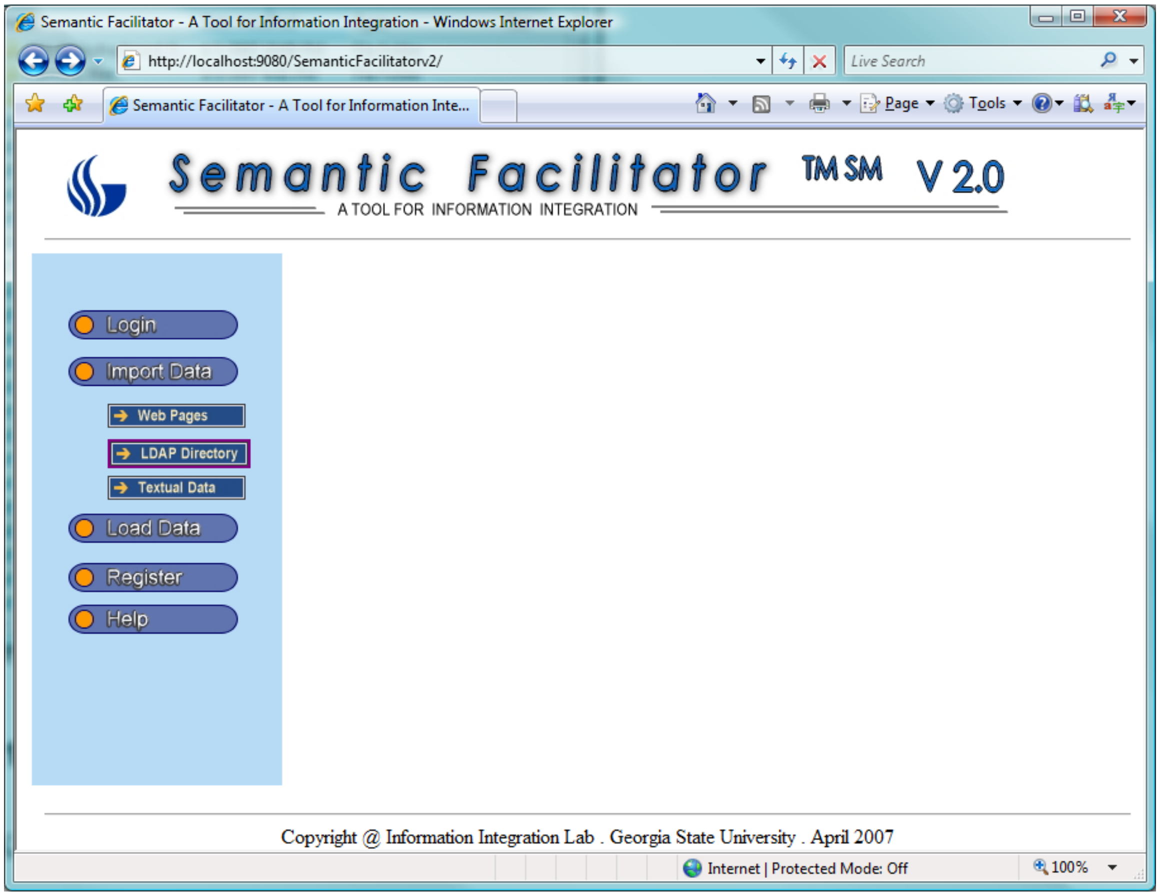

Semantic Facilitator - A Tool for Information Integration - Windows Internet Explorer
http://localhost:9080/SemanticFacilitatorv2/
Live Search
Semantic Facilitator - A Tool for Information Inte...
Page
Tools
Semantic Facilitator TM SM V 2.0
A TOOL FOR INFORMATION INTEGRATION
Login
Import Data
Web Pages
LDAP Directory
Textual Data
Load Data
Register
Help
Copyright @ Information Integration Lab . Georgia State University . April 2007
Internet | Protected Mode: Off
100%

Figure 4.2 Semantic Facilitator V2.0 Start Page

4.2.1 Implementation of Keyword Extractor

The keyword extractor is an important component of the prototype. Written in Java language, the keyword extractor includes a web crawler that extracts textual information from the Web and a keyword processor that processes the textual information and generates a list of keywords that are relevant to the user perspective. Such extraction is done by analyzing the structure of the web pages. For example, in the NDG dataset, if the user perspective is about the description of the databases, only the “description” section on the web page is selected for the clustering. This is part of the reason the prototype only works well with structured or semi-structured web pages. The keywords from each web page are tokenized by comparing them to a universal keyword list (all unique words from all URLs) and a data file that can be used as SOM input data is created.

The program flow of the keyword extractor is shown in figure 4.3. The source code for the keyword extractor is listed in appendix B.

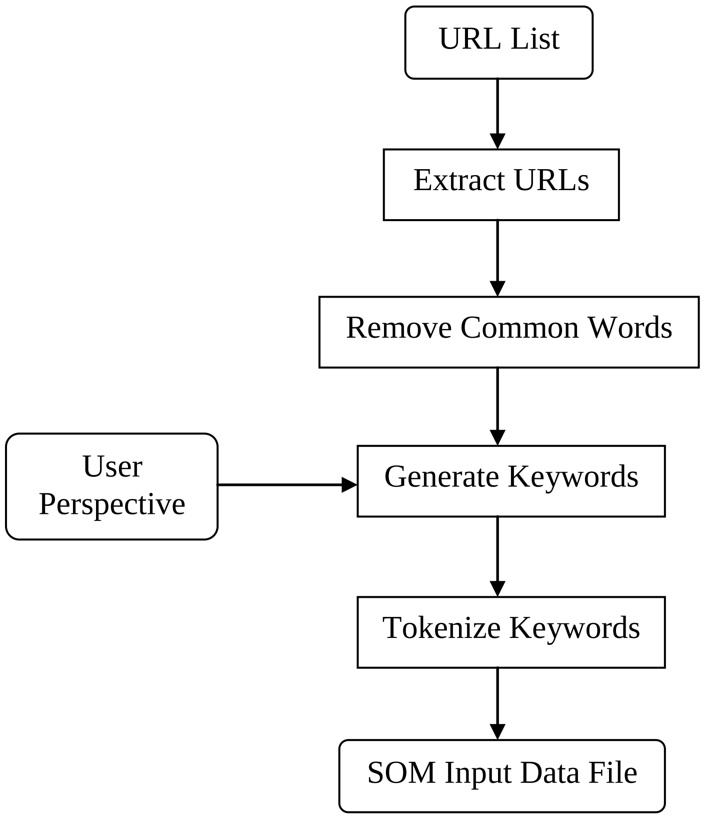


Figure 4.3 Keywords Extractor Program Flow

4.2.2 Implementation of SOM Clustering Algorithm

The SOM clustering algorithm implementation is adapted from the SOM_PAK software package from Helsiniki University of Technology (Kohonen et al. 1995) . SOM_PAK is coded with C programming language. SOM_PAK was also used in our previous prototype Semantic Facilitator $^{TM\ SM}$ (Vandenberg, Liang et al. 2002) (see figure 4.4 for its system architecture) to cluster LDAP directory metadata, and it performed well. This prototype continues to use SOM_PAK package as the SOM clustering engine. However, some modifications such as transforming the input data format and changing the output data format have been made to the original code to make it work with the particular dataset in this dissertation.

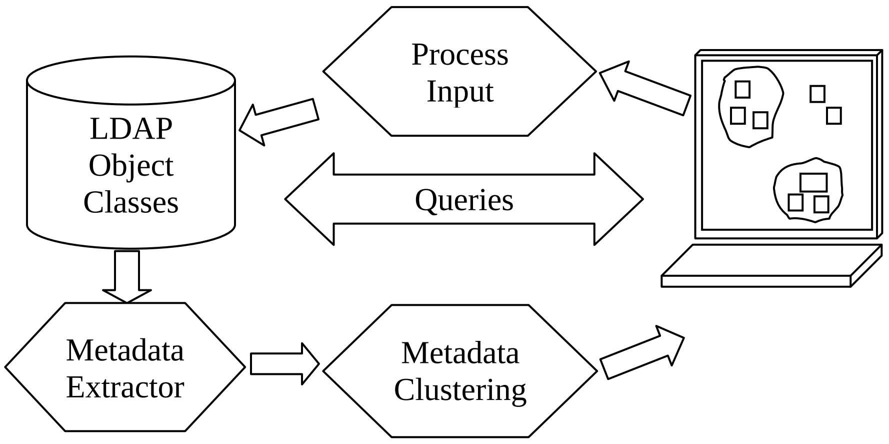


Figure 4.4 Semantic Facilitator $^{TM\ SM}$ System Architecture

4.2.3 Implementation of Interactive User Interface

The proposed research approach emphasizes the importance of an interactive user interface. The SOM visualization component is implemented by adapting an information visualization toolkit, Prefuse, which is written in Java language. Prefuse is “an extensible software framework for helping software developers create interactive information visualization applications using the Java programming language. It can be used to build standalone applications, visual components embedded in larger applications, and web applets. Our intention in using Prefuse was to simplify the processes of representing and efficiently handling data and mapping data to visual representations. I chose Prefuse because it is simple to use and can be easily integrated into the prototype as a block of Java code.

The SOM visualization component has two major programs: SOMConverter.java and SOMVisulizer.java. SOMConverter transforms the SOM map output into a Prefuse-compatible format. SOMVisulizer takes the SOM map output as input and displays it in a two dimensional map. Each web page (or URL) is represented by a node with an icon of a web page. The user interface is highly interactive: if the computer mouse moves over a node, the URL address of that node pops up, and the user can double click on the node to open the URL link in a separate web browser. The user can easily move the URL list from one cluster to another by a dragging and dropping operation, and the user can choose to save the modified map or to go back to the previous map. Also, the labels for each URL node can be displayed or hid with one click. The idea is that the users can easily make changes to the SOM map if they want to.

Figure 4.5 illustrates a sample SOM map (with "Show Label" function turned on). The pseudo code for the visualization component is shown in Figure 4.6. The source code of the visualization component is listed in appendix C.

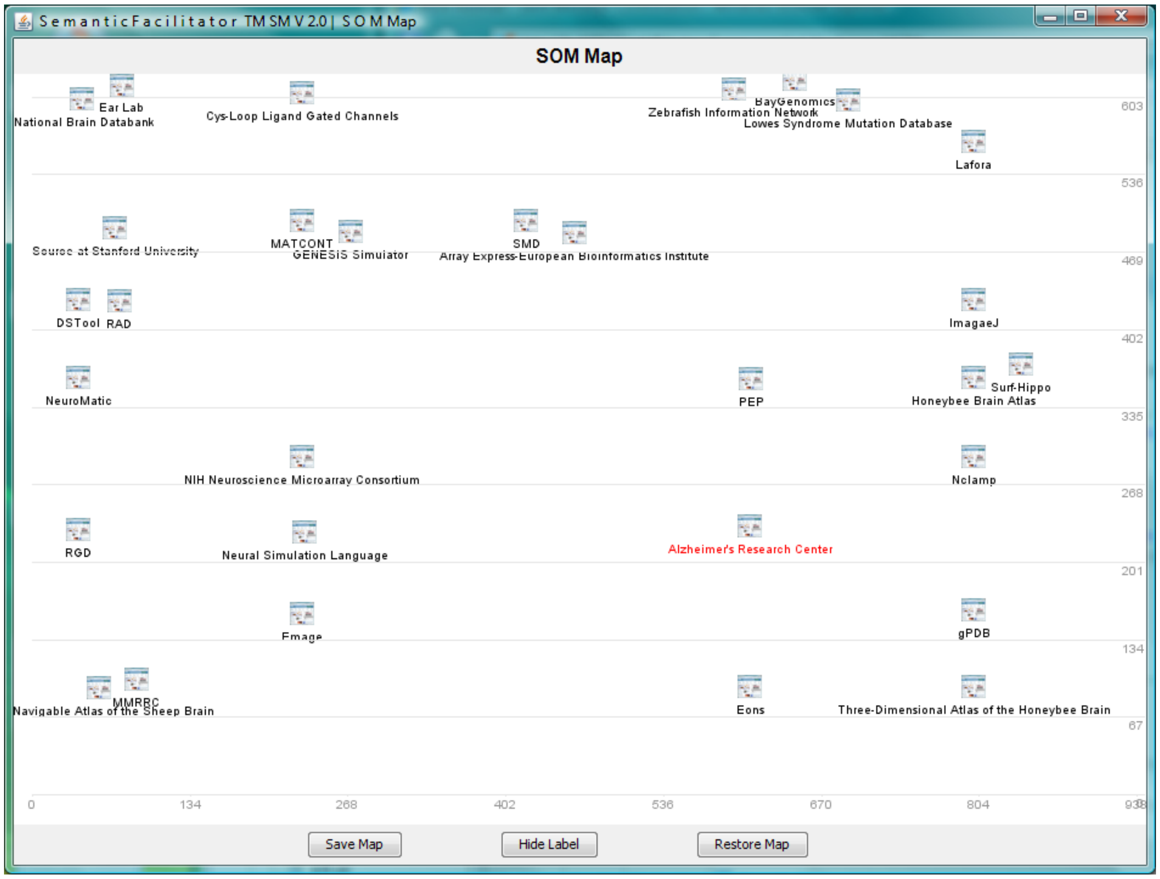

SemanticFacilitator TM SM V 2.0 | SOM Map
SOM Map
National Brain Databank
Ear Lab
Cys-Loop Ligand Gated Channels
Zebrafish Information Network
BayGenomics
Lowes Syndrome Mutation Database
Lafora
Source at Stanford University
MATCONT
GENESIS Simulator
SMD
Array Express-European Bioinformatics Institute
DSTool
RAD
ImagaeJ
NeuroMatic
PEP
Surf-Hippo
Honeybee Brain Atlas
NIH Neuroscience Microarray Consortium
Nclamp
RGD
Neural Simulation Language
Alzheimer's Research Center
Emage
gPDB
Navigable Atlas of the Sheep Brain
MMRRC
Eons
Three-Dimensional Atlas of the Honeybee Brain
603
536
469
402
335
268
201
134
67
0
134
268
402
536
670
804
938
Save Map
Hide Label
Restore Map

Figure 4.5 A Sample SOM Map of Semantic Facilitator $^{\text{TM SM}}$ V2.0

```
// Pseudo code for Visualization Component
// prepare data for visualization by Prefuse
get web browser screen size
get SOM map size
calculate the size of each SOM Cell
for each node (web page) on the map
   calculate the its position on the map
   read URL name
end for
// display the SOM map on the browser screen
read SOM map data file
for each node (web page) on the map
   display the node on the screen
   load the image file for the node
end for
Listen to Mouse Event
If Mouse Over a node
   Display tooltip for that node
If Mouse double-click on the node
   Open the webpage on a separate window
If Mouse Drag & Drop
   Move the selected node to the new location
If "Save Map" button clicked
   Save current map
If "Show label" button clicked
   Show labels for all nodes
If "Restore Map" button clicked
   Discard the changes made the user
    Reload the SOM Map
```

Figure 4.6 Pseudo Code for Visualization Component

4.2.4 Implementation of Genetic Algorithm & Grid

The genetic algorithm and Grid computing infrastructure are also important features of Semantic Facilitator $^{TM\ SM}$ V2.0. SOM parameter values are domain dependent and have a great impact on the SOM's clustering performance. In the prototype, I introduce a genetic algorithm (GA) to systematically search good SOM parameter values. In addition, a Gird computing infrastructure is applied to provide computing power since both SOM and GA are computational intensive. A GA is perfectly suitable for Grid computing because a big task can be easily divided into small tasks with each small task independent of the others. These small tasks can then be assigned to Grid nodes and the results collected and combined to form a composite output.

The genetic algorithm implementation is adapted from GALib code package (GAlib 2005). GAlib is a widely used C++ library of Genetic Algorithm Components. It can be easily integrated with the SOM program which is also based on C code.

The Grid used in this research prototype is a small-scaled in-house Grid implementation that contains 14 nodes (each Grid node is a Dell PC with 933 MHz CPU and 512 MB RAM). Globus Toolkit ^{TM} (GlobusAlliance 2005), built on the Open Grid Services Architecture (OGSA) open source software, is used for a higher level of resource management services (e.g., sharing computational and other resources without sacrificing local autonomy). PBSpro was used to make the Grid system call for submitting tasks to Grid nodes, scheduling the tasks for running, executing tasks, and collecting the results.

Clustering is the core functionality of Semantic Facilitator $^{TM\ SM}$ V2.0. It's critical that the GA and Grid computing infrastructure can be seamlessly integrated with SOM to achieve more effective and more efficient clustering. The next section discusses in detail the implementation of the GA and the integration of the GA and Grid with SOM.

The source codes for the GA and Grid components are listed in appendix D.

4.3 Integration of SOM, Genetic Algorithm and Grid

4.3.1 Genetic Algorithm and SOM

Introduced by John Holland in the 1970s, a genetic algorithm enables computer programs to “evolve” in ways that resemble the natural selection process, characterized by *crossbreeding*, *mutation* and *survival of the fittest*. A GA makes it possible to explore a far greater range of potential solutions to a problem than do conventional programs (Holland 1975). Compared to traditional search algorithms, a genetic algorithm is able to automatically acquire and accumulate the implicit knowledge about the search space during its search process, and self-adaptively control the search process through a random optimization technique. It often yields a globally optimal solution and avoids combinatorial explosion by disregarding certain parts of the search space [36].

The appeal of combining GA and neural network based SOM arises from the expectation that GA might provide a systematic approach for an efficient search for “optimal” SOM parameter values in the large space of network structures. A second motivation is the biological roots that both paradigms share (Polani 1999)]. Polani and Uthmann’s research has successfully applied genetic algorithms to improve the topology of a Kohonen feature map (Polani and Uthmann 1993).

In this research prototype, GA is coupled with SOM to systematically search for good SOM parameter values. There are two important issues that need to be addressed for the coupling to be successful: 1) Fitness function: how can the GA determine whether a given clustering result is good or not? 2) Genetic coding: how are the SOM parameters mapped into the GA's "genomes"?

Fitness Function. This dissertation uses three widely used metrics cluster recall, cluster precision, and F-measure to evaluate the SOM clustering results (the definitions of the metrics are discussed in chapter 5). F-measure is the overall representation of cluster recall and cluster precision. I use F-measure as the indicator of SOM clustering performance.

Genetic Encoding of SOM Parameter Values. The process that maps an algorithm's parameter values to individual "genomes" of a genetic algorithm is called genetic encoding. The robustness of genetic encoding is assured by meeting a minimum of three encoding criteria: completeness, soundness, and non-redundancy (Goldberg 1989). Following the approach used in (Liang, Vaishnavi et al. 2006), four SOM parameter values are varied: *xdim*, *ydim*, *neighborhood size*, and *number of final training iterations*. Therefore, the genetic genome is composed of those four parameter values, satisfying completeness and soundness for genetic encoding since they are values of significant parameters that impact SOM clustering performance. Moreover, *xdim*, *ydim*, *neighborhood size*, and *final training iteration* are independent parameters according to SOM theory, thus assuring non-redundancy. In summary, the genome-encoding is robust.

The value range for each parameter must be determined. On the one hand, varying the parameter values in a wide range ensures a more complete search space. On the other hand, the range cannot be unreasonably large because GA is very computationally intensive. There is some theoretical guidance for selecting SOM parameter value ranges. According to (Kohonen, Kaski et al. 2000), the SOM map ought to be rather more rectangular than square and the initial *neighborhood size* should be nearly the size of the SOM map itself. The effectiveness and efficiency of the SOM map partially depends on how many clusters it forms. With too many clusters on the map, users will spend too much time identifying their desired clusters. With too few clusters on the map, users might get low cluster precision. Humans usually can only maintain seven (plus or minus two) items in short term memory at one time (Miller 1956). Guided by these considerations, *xdim* and *ydim* values range from 3 to 10 and *neighborhood size* values range from 2 to 6. Many researchers (Kiang et al. 1995) (Mangiameli, Chen et al. 1996) suggest typical training iterations of about 20,000, so the *number of final training iterations* ranges from 5,000 to 60,000.

4.3.2 GA Operations Configuration

There are three types of GA operations: selection, crossover, and mutation. Their configurations are discussed as follows.

Selection operation. The selection operation decides which individuals from the current generation will be carried on to the next generation. The fitness (F-measure value) is computed for every individual in the population and the higher the fitness (i.e. its SOM clustering result as compared to the human experts), the better chance an individual has of being selected for the next generation. Replacement probability defines what percentage of the old population will be carried on to the next generation. Low replacement probability can increase the GA's capability to search a new solution space. Therefore, it is set to 0.1 in this implementation so that each generation will have the same population size.

Crossover operation. The crossover operation mimics the gene recombination of biological evolution. Several crossover schemes such as one-point crossover, two-point crossover, and multi-point crossover have been used in genetic algorithms. Since the genome in this implementation just has four elements, a single point crossover was applied. Participants in crossover are probabilistically selected from the population and the crossover points are selected randomly. The elements at the crossover points are simply exchanged between two individuals as illustrated in Figure 4.7.

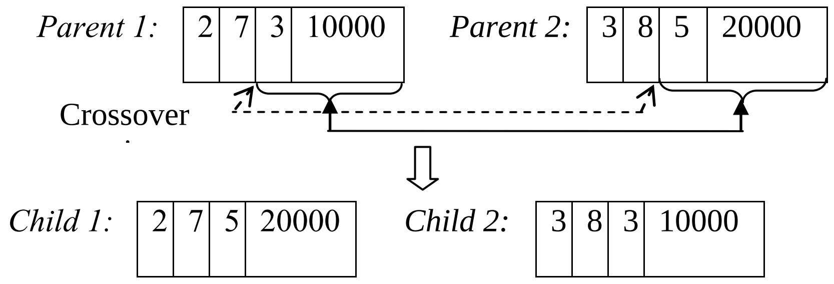


Figure 4.7 Genetic Algorithm Crossover Operation

Crossover is a predominant operation in a genetic algorithm. A high crossover probability is preferable so that the GA extends its search space, exploring more solutions. Therefore, the crossover probability is set to 0.8 in this implementation.

Mutation operation. The mutation operator makes random changes on some individuals of each generation. It allows a generation to jump outside a local optimum and thus maintain the variety of the solution population. To perform the mutation operation, a mutation point is randomly selected and the value at that point is replaced by a random number within the value range of the corresponding parameter value. The mutation operation is illustrated in Figure 4.8.

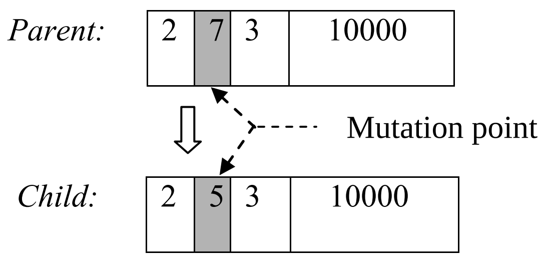


Figure 4.8 Genetic Algorithm Mutation Operation

Mutation can extend the GA's search space. However, frequent use of the mutation operation may make the Genetic Algorithm conduct a random search, so a small mutation probability is preferred. I set it at 0.08 in this implementation.

4.3.3 GA Parameter Selection and Execution

GA parameters include probabilities of genetic operations, population size, generation number, and other details of the run. Probabilities of genetic operations (selection, crossover, mutation) were discussed above. Another important control parameter is the population size. Usually a population size is chosen that will produce a reasonably large number of individuals across all generations before terminating. Population size is set to 120 in this implementation. A convergence indicator (the similarity of the best fitness values – F-measure – between predefined numbers of adjacent generations) is used as the termination condition. The convergence indicator is set to a reasonably large value (0.99) to make sure that the GA becomes steady and has finally discovered a reasonably good result.

The program flow of GA in Semantic Facilitator $^{\text{TM SM}}$ V2.0 is showed in figure 4.9. The GA starts with a randomly generated initial population of genomes (individuals). The GA decodes each genome (converts them into corresponding SOM parameter values), uses the values to run the SOM algorithm, and then calculates the fitness function value (based on F-measure) by comparing the SOM clustering result (knowledge model) of each individual genome to the standard knowledge model taken from web-based resources. When all the genomes in a population are evaluated, GA tests whether the termination condition is met. If so, the best genome from the population is selected and the program ends. Otherwise, the GA performs selection, crossover, or mutation operations to create a new population for the next generation. This process of generating a new population, running SOM, calculating fitness, and evaluating the termination condition is repeated until the termination condition is met. Once the termination condition is met, GA stops running and reports the best SOM parameter values it discovered to a textual file. The Semantic Facilitator $^{TM\ SM}$ V2.0 read this file and run the SOM clustering engine again using the SOM parameter values. The generated SOM map is sent to the visualization component. By doing this, GA and SOM are seamlessly coupled together.

4.3.4 Grid and GA Enabled SOM Clustering

As I discuss earlier, GA and SOM both are computational intensive. I introduce a Grid computing infrastructure to reduce the running time of the prototype to a practical level. Figure 4.10 shows the pseudo code of how GA is integrated with Grid. In this implementation, I not only carefully design how a Grid job is composed, submitted, and checked, but also ensure the error handling of the system (a Grid job may fail or may return a wrong result). The following is a summary of how Grid works with GA.

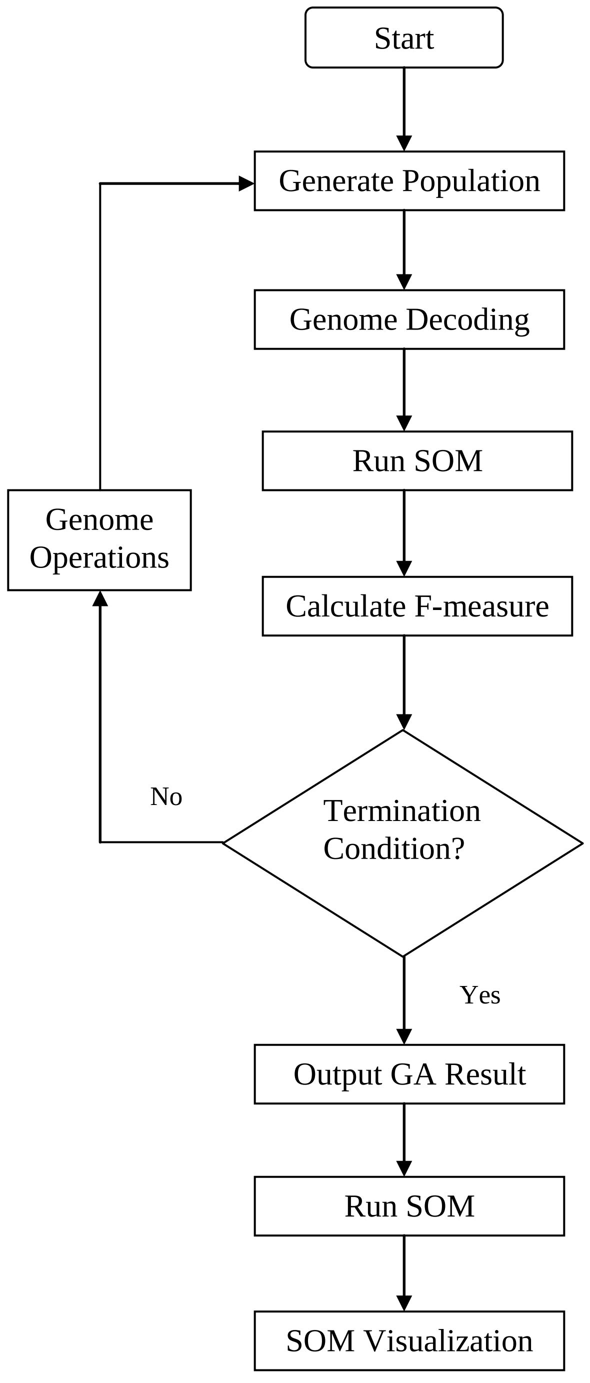


Figure 4.9 Semantic Facilitator TM SM V2.0-GA Component Program Flow

```
// Pseudo code for applying Grid to GA
for each GA generation
    //compose and submit a task to Grid node
    for each genome in a generation
        retrieve & decode genome
        create a task for PBSpro
        submit the task to PBSPro
        PBSPro assign the task to a Grid node
        SOM runs on assigned Grid node
    next genome

    //check status of Grid task and get the running result back
    while not all genomes evaluated
        for each genome in a generation
            if genome not evaluated
                if the task associated with genome finished
                    retrieve the running result of the task
                      if the result falls in correct range
                          mark the genome as evaluated
                      else
                          resubmit the task
                else if the task associated with genome failed
                    resubmit the task
        next genome
     end while

     generate next generation
next generation
```

Figure 4.10 Pseudo Code for Applying Grid to GA

A GA runs one generation at a time and each generation has a population of genomes (the genomes are independent to each other). After decoding, each genome is transformed into set of SOM parameter values and a Grid job file (*.pbs) is created for each genome. This file contains commands for how to run SOM on a Grid node and what output file the Grid should return. The Grid job file is submitted to a Grid node through PBSPro. SOM runs on the Grid node using the parameter values in the task file and the output is stored in the specified file and sent back to PBSPro. At the same time, GA runs an infinite loop to check if all tasks are finished. If a task is finished, the running result (F-measure of corresponding SOM run) is retrieved, and the associated genome is marked as evaluated. If the task isn't finished, the system keeps checking. If a task times out or returns a wrong result (e.g., F-measure needs to be between 0 and 1), the task is resubmitted to the Grid. The loop continues until all the genomes in the generation are evaluated. The GA performs selection, crossover, and mutation operations to create a new generation. The creating Grid job, submitting Grid job and check status of Grid job processes are repeated for the new generation.

4.4 Research Findings

Based on the research approach presented in chapter 3, a research prototype Semantic Facilitator $^{TM\ SM}$ V2.0, was designed and implemented. I first developed system architecture for the prototype. I then discussed the different technologies used in the prototype and their integration. In the implementation, I took advantage of available software packages for adapted technologies, which not only substantially reduced development time, but also improved system stability.

Although the research prototype is still preliminary, it makes contributions in two ways. First, the research prototypes can serve as a good tool for web users to better use and understand the information on the web. For example, a neuroscientist can use the prototype more effectively organize and utilize the database information on the NDG website. Second, the research prototype can act as a work bench for IS researchers to conduct more studies. I can use the prototype to evaluate the proposed research approach. For another example, an IS researcher can study the cognitive impact of the system to web users by setting up experiments and watching users interacting with the system. This would lead to some interesting behavior science type research.

Chapter 5: Evaluation - Experiment Design & Results

In this chapter, I report the work done on the *evaluation* phase of the General Design Cycle. The research prototype, Semantic Facilitator $^{\text{TM SM}}$ V2.0, is evaluated according to the research questions identified in the *awareness of problem* phase, and I chose experimentation as the evaluation method. I discuss the details of the general experiment design, such as data selection, evaluation metrics, and experiment software and hardware environment. Then I introduce the research hypotheses derived from the research questions and three experiments are designed to test those hypotheses. Finally, the detailed designs and results of each experiment are presented.

5.1 General Experiment Design

5.1.1 Evaluation methodology and Data Selection

In this dissertation, I use controlled experiment as the evaluation method. An experiment is "a study in which an intervention is deliberately introduced to observe its effects" (Shadish et al. 2002). An experiment provides a controlled environment where external variance is minimized in order to maximize the chances of determining if the tested approach has any effect and is widely used to evaluate theory and hypotheses (Calder et al. 1981). Experimentation is also commonly used to evaluate design science research efforts (Hevner, March et al. 2004) which makes it an appropriate method to use in this research.

In the experiments, I purposely chose, Neuroscience Database Gateway (NDG) website, as the web-based resource of interest. The contents on NDG website are already categorized from 8 different perspectives by a panel of neuroscientists through a lengthy and rigorous process. Such data selection enables a consequentially simplified and effective experiment design.

5.1.2 Justification of Using Simulation-based Experiment

In the proposed approach, I emphasize the *user-centric* and *adaptable* features, and user-system interaction is required in the knowledge model building process. The experiment design operates with a simulated user to assess the correctness of the proposed approach. This decision is due to two considerations.

1). Using a simulated user can enable me to effectively and economically test the proposed research approach. A human user has two roles in the experiments: one role is acting as an “ordinary user” who uses and interacts with the system (research prototype); the other role is acting as an “expert user” who manually clusters the web-based resource, and their results are used as the standard knowledge model to evaluate the knowledge models generated by the system. The activities of both users can be easily and accurately modeled by a computer simulation.

The “ordinary user” has two actions in interacting with the research prototype: specify the user’s preferred perspective/viewpoint and give feedback to the prototype (change system generated knowledge models). The first action can be simulated by randomly picking up a perspective from the available perspectives in web-based resources of user interest. The second action can be simulated by randomly selecting model components (clusters) in a system generated knowledge model and changing them based on the standard knowledge model (already available on NDG website). The cognitive aspect of a human user that could be very difficult to model addressed in this dissertation. Thus, it is a relatively simple task to model an “ordinary user.”

The simulation of an “expert user” is an even simpler task. The chosen data source (NDG website) already has its content categorized by a group of domain experts. The pre-existing categorization can be considered as the clustering results of a simulated “expert user” and used as standard knowledge model.

In summary, the activities of an “expert user” and an “ordinary user” can be accurately mimicked by a computer simulation. In addition, it would be cost-prohibitive and impractical to invite all neuroscientists in the NDG website panel to participate in the experiment. Thus, using a simulated user in the experiments becomes a reasonable decision because of its effectiveness and economy.

2) Operating with a (real) human user in the experiments would confound the evaluation process. The appropriateness of the system generated knowledge model is the most important indicator of the correctness of the proposed approach. Thus, how to objectively evaluate the computer generated knowledge model becomes a critical issue. If a human user is used in the experiments, the user satisfaction of the final knowledge model could be used as an evaluation metric. However, such evaluation metrics is rather subjective measurement.

An alternative approach is to ask the same or different user to manually create the knowledge model based on the user preferred perspective and use that model as the standard knowledge model. This approach is still problematic. Even if the experiment generates positive results, I can only prove the system is good for this particular user or not for users in general. I could increase the number of users in the experiments, but it would be difficult to manage and complicated to aggregate the experiment results of different users.

For example, even if I could have all the neuroscientists in the NDG website panel in my experiments, it would be still very difficult to objectively evaluate the knowledge models created by the research approach. Prior study (Liang, Vaishnavi et al. 2006) showed each human user tends to have a different understanding of how a dataset should be organized when they are given a same task and same dataset. In other words, I cannot objectively evaluate a knowledge model created by a neuroscientist interacting with the research prototype based on a knowledge model created by aggregative efforts of a group of neuroscientists.

In the above scenario, it seems to be an appropriate solution for the evaluation issue. Because the standard knowledge model is the aggregation of efforts of all neuroscientists, a computer simulated neuroscientist can be designed as a solo owner of the standard knowledge model, thus can act as a perfect representative of the panelists. With the standard knowledge model in "mind," the simulated neuroscientist can interact with the research prototype. The knowledge model generated in this process can be properly and objectively evaluated based on the standard knowledge model.

Based on the above discussion, it's appropriate to use a simulated user in the experiments.

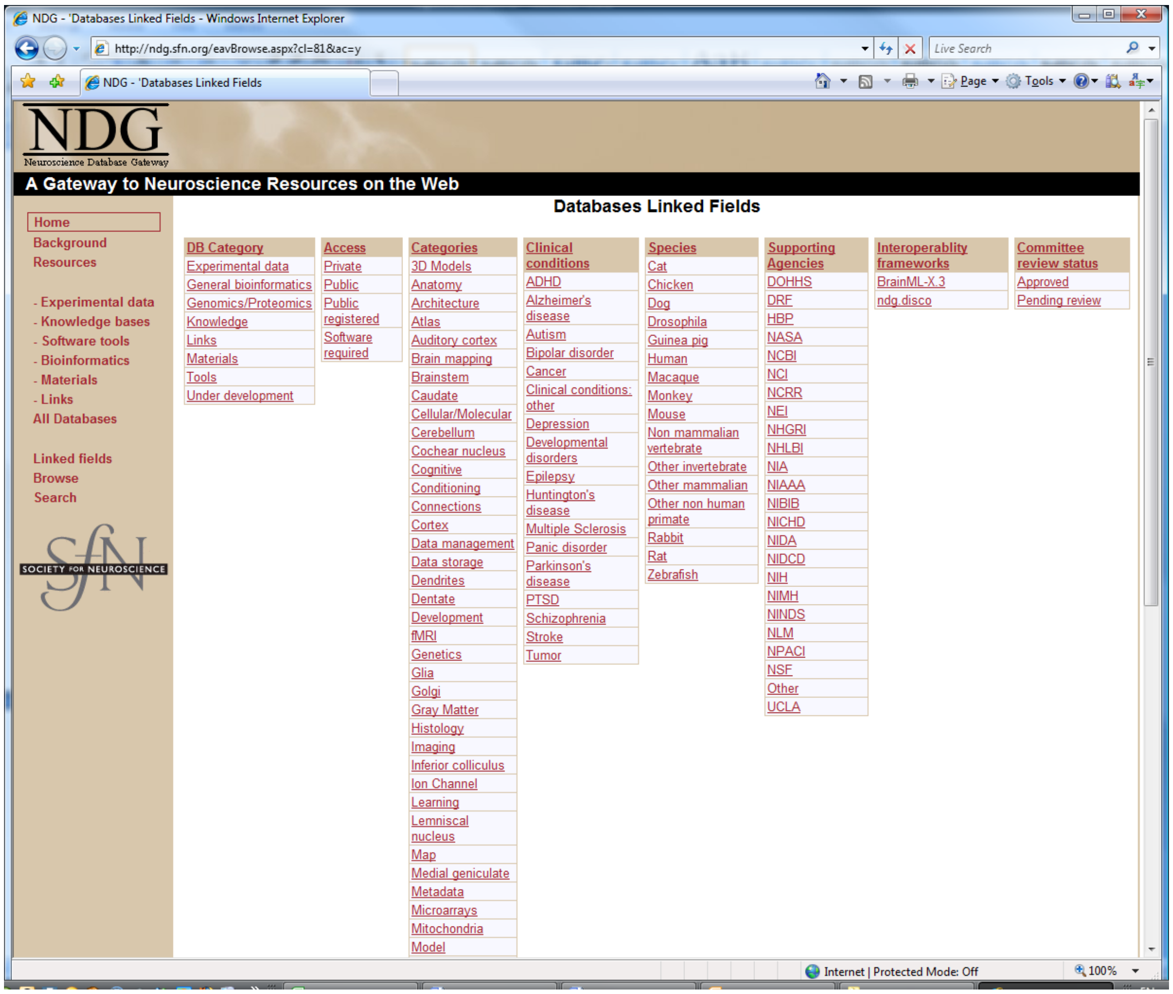
5.1.3 Experiments Overview

In the evaluation phase, three experiments are designed to evaluate the research approach. Experiment 1 studies the effectiveness of a Genetic Algorithm on the clustering performance of Semantic Facilitator $^{TM\ SM}$ V2.0. Experiment 2 studies the impact of user feedback on the system's clustering performance. A simulated user is used in this experiment to interact with Semantic Facilitator $^{TM\ SM}$ V2.0 and provide feedbacks (e.g., modifying the clusters). Experiment 3 addresses the efficiency of the proposed approach. The same task is executed by Semantic Facilitator $^{TM\ SM}$ V2.0 under two different environments: with Grid computing infrastructure and without Grid computing infrastructure. The processing time of each execution is recorded and compared.

5. 2 Experiment Data Selection

5.2.1 Data Source Selection

The raw data of the experiments are drawn from a large web portal, Society for Neuroscience (SfN) Neuroscience Database Gateway (NDG) website. NDG site is "a pilot project developed by the Brain Information Group (BIG)" and " aimed at promoting awareness and facilitating access to relevant neuroscience databases" (http://ndg.sfn.org/). Currently the NDG website contains links to 176 neuroscience databases. Those databases are carefully evaluated and categorized by the NDG sub-committee at the Society for Neuroscience. The databases are categorized based on 8 different viewpoints such as DB category, access, categories, species, etc (see Figure 5.1). NDG has sub pages for each database. Those pages contain the metadata for the databases such as name, database description, and URL, etc (see figure 5.2 for sample NDG page for a neuroscience database).



NDG - 'Databases Linked Fields - Windows Internet Explorer

http://ndg.sfn.org/eavBrowse.aspx?cl=81&ac=y

NDG
Neuroscience Database Gateway

A Gateway to Neuroscience Resources on the Web

Home
Background
Resources

- Experimental data
- Knowledge bases
- Software tools
- Bioinformatics
- Materials
- Links

All Databases

Linked fields
Browse
Search

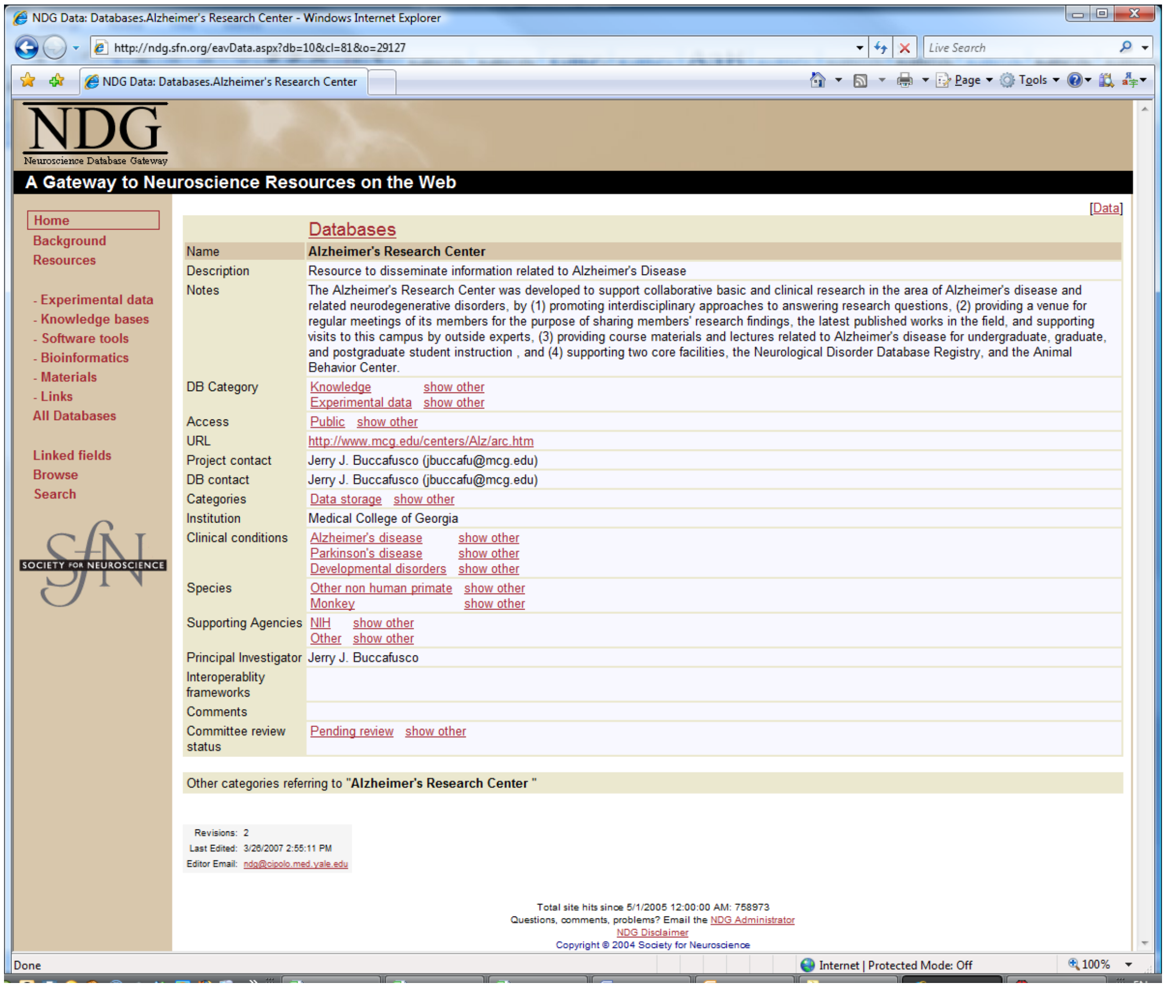
SOCIETY FOR NEUROSCIENCE

Databases Linked Fields

DB Category	Access	Categories	Clinical conditions	Species	Supporting Agencies	Interoperablity frameworks	Committee review status
Experimental data	Private	3D Models	ADHD	Cat	DOHHS	BrainML-X.3	Approved
General bioinformatics	Public	Anatomy	Alzheimer's disease	Chicken	DRF	ndg.disco	Pending review
Genomics/Proteomics	Public registered	Architecture	Autism	Dog	HBP		
Knowledge	Software required	Atlas	Bipolar disorder	Drosophila	NASA		
Links		Auditory cortex	Cancer	Guinea pig	NCBI		
Materials		Brain mapping	Clinical conditions: other	Human	NCI		
Tools		Brainstem	Depression	Macaque	NCRR		
Under development		Caudate	Developmental disorders	Monkey	NEI		
		Cellular/Molecular	Epilepsy	Mouse	NHGRI		
		Cerebellum	Huntington's disease	Non mammalian vertebrate	NHLBI		
		Cochear nucleus	Multiple Sclerosis	Other invertebrate	NIA		
		Cognitive	Panic disorder	Other mammalian	NIAAA		
		Conditioning	Parkinson's disease	Other non human primate	NIBIB		
		Connections	PTSD	Rabbit	NICHD		
		Cortex	Schizophrenia	Rat	NIDA		
		Data management	Stroke	Zebrafish	NIDCD		
		Data storage	Tumor		NIH		
		Dendrites			NIMH		
		Dentate			NINDS		
		Development			NLM		
		fMRI			NPACI		
		Genetics			NSF		
		Glia			Other		
		Golgi			UCLA		
		Gray Matter					
		Histology					
		Imaging					
		Inferior colliculus					
		Ion Channel					
		Learning					
		Lemniscal nucleus					
		Map					
		Medial geniculate					
		Metadata					
		Microarrays					
		Mitochondria					
		Model					

Internet | Protected Mode: Off

Figure 5.1 Categorization Perspectives from NDG Website



NDG Data: Databases.Alzheimer's Research Center - Windows Internet Explorer

http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29127

NDG
Neuroscience Database Gateway

A Gateway to Neuroscience Resources on the Web

Home
Background
Resources

- Experimental data
- Knowledge bases
- Software tools
- Bioinformatics
- Materials
- Links

All Databases

Linked fields
Browse
Search

[Data]

Databases

Name	**Alzheimer's Research Center**
Description	Resource to disseminate information related to Alzheimer's Disease
Notes	The Alzheimer's Research Center was developed to support collaborative basic and clinical research in the area of Alzheimer's disease and related neurodegenerative disorders, by (1) promoting interdisciplinary approaches to answering research questions, (2) providing a venue for regular meetings of its members for the purpose of sharing members' research findings, the latest published works in the field, and supporting visits to this campus by outside experts, (3) providing course materials and lectures related to Alzheimer's disease for undergraduate, graduate, and postgraduate student instruction , and (4) supporting two core facilities, the Neurological Disorder Database Registry, and the Animal Behavior Center.
DB Category	Knowledge show other Experimental data show other
Access	Public show other
URL	http://www.mcg.edu/centers/Alz/arc.htm
Project contact	Jerry J. Buccafusco (jbuccafu@mcg.edu)
DB contact	Jerry J. Buccafusco (jbuccafu@mcg.edu)
Categories	Data storage show other
Institution	Medical College of Georgia
Clinical conditions	Alzheimer's disease show other Parkinson's disease show other Developmental disorders show other
Species	Other non human primate show other Monkey show other
Supporting Agencies	NIH show other Other show other
Principal Investigator	Jerry J. Buccafusco
Interoperablity frameworks	
Comments	
Committee review status	Pending review show other

Other categories referring to "**Alzheimer's Research Center** "

Revisions:	2
Last Edited:	3/26/2007 2:55:11 PM
Editor Email:	ndg@cipolo.med.yale.edu

Total site hits since 5/1/2005 12:00:00 AM: 758973
Questions, comments, problems? Email the NDG Administrator
NDG Disclaimer
Copyright © 2004 Society for Neuroscience

Figure 5.2 A Sample Web Page on SfN NDG Web Site.

The SfN NDG website is chosen as the data sources for our experiments for the following reasons.

- The NDG site is a large and complex web portal which contains links to over 170 databases. This can improve the external validity of the experiments.
- The databases are organized based on 8 different viewpoints by a group of domain experts. The categorization process is very rigorous and done by a group of human experts making the results authoritative. The domain experts' categorizations can be used as standard knowledge models to evaluate the

knowledge models generated by the proposed approach. In addition, multiple viewpoints on the website (about how the databases can be categorized) can be used to simulate different viewpoints from the user.

- The web page that describes the neuroscience database metadata is well structured. The current version of Semantic Facilitator $^{TM\ SM}$ V2.0 can only handle structured or semi-structured web pages well.
- The research result would benefit the neuroscience research community. This is the part of the responsibility of the author as a fellow of the Georgia State University Brain & Behavior program.

5.2.2 Selection of the Experiment Data

There are 8 perspectives for database categorizations at the SfN NDG website (see figure 5.1.). The "categories" perspective is chosen as simulated user preferred perspective. The 176 databases are grouped into 65 categories. 46 databases are carefully selected from the 176 databases to make sure that there is little or no (1 or fewer) category overlap among the selected databases. The chosen databases and their belonging categories are listed in table 5.1. On NDG web site, each database has a corresponding web page describing its metadata. Those web pages form the raw data of the experiments.

Table 5.1 Selected Database Web Pages from NDG Web Site

Category	**Database Web Page Name**
Data storage	Alzheimer's Research Center
Microarrays	Array Express-European Bioinformatics Institute
	NIH Neuroscience Microarray Consortium
	SMD
Cellular/Molecular	BayGenomics
	Ki-Databases
Genetics	Lafora
	Lowes Syndrome Mutation Database
	Source at Stanford University
	Uniprot
	Zebrafish Information Network
Imageing	ImagaeJ
Physiology	Cys-Loop Ligand Gated Channels
Model	Duke/Southhampton Archive of Neuronal Morphology
	MMRRC (Genetics)
Software	IATR
	Nclamp
	EEG Lab (Data Management)
	NeuroMatic (Data Management)
3D Model	Protein Data Bank
	PEP
	FlexProt (Software)
Neuroinformatics	Visiome
	Brede (Ontology)
Cellular/Molecular & Genetics	gPDB
	Hereditary Hearing Loss Homepage
	SynDB
	National Brain Databank (Cortex)
Atlas & Anatomy	HyperBrain Brain Atlas
	The Navigable Atlas of the Sheep Brain
Model & Software	Ear Lab
	Eons
3D Model Anatomy & Genetics	Emage
	FlyTrap
Genetics, Microarray, & Ontology	RAD
	RGD
Simulation & Software	Neural Simulation Language
	DSTool (3D Model)
	GENESIS Simulator (Model)
	MATCONT (Neuroinformatics)
3D Model Anatomy & Atlas	Honeybee Brain Atlas (Map)
	Brain Architecture Center (Brain Map)
	Mouse Brain Atlas (Histology)
	Three-Dimensional Atlas of the Honeybee Brain
Brain Mapping, cellular/Molecular & Software	FMR Lab (FMR Lab)
	Surf-Hippo (Neuroinformatics, Simulation)

The NDG web pages are retrieved from the NDG website and stored in our web server. For example, Alzheimer's Research Center's URL on NDG is http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29127. This neuroscience database page is cached and stored in our web server. The new URL is http://www.cis.gsu.edu/~lli/data/29127.htm. This makes sure the dataset is consistent during the experiment because I do not have control over the NDG web site and therefore the pages on the NDG website may changes during the course of the experiment. In addition, those web pages are still hosted and accessed through the Web. However, this won't affect the validity of the experiments.

In the experiments, the 46 database web pages are randomly divided into 2 groups: a training dataset (about 2/3 of all databases, 30 database pages) and a testing dataset (about 1/3 of all databases, 16 database pages). Randomization is done in the following manner: each database first is assigned a random number ranging from 0 to 1. Then the databases are ranked based on their associated random numbers. The first 30 databases are chosen as the training dataset, and the other 16 databases are used as the testing dataset. The training and testing datasets are listed in Appendix A. For each of the datasets, Semantic Facilitator $^{TM\ SM}$ V2.0 is used to categorize the selected web pages based on the contents of the pages.

5.3 Experiment Evaluation Metrics

In the experiments, I use three widely accepted metrics to evaluate Semantic Facilitator TM SM V2.0's clustering performance: cluster recall (CR), cluster precision (CP) and F-measure. The definitions of CR and CP (adapted from (Roussinov and Chen 1999)) are listed as follows.

$$CR = \frac{total\quad number\quad of\quad correct\quad associations\quad in\quad computer\ partition}{total\quad number\quad of\quad associations\quad in\quad manual\quad partition}$$

$$CP = \frac{total\quad number\quad of\quad correct\quad associations\quad in\quad computer\ partition}{total\quad number\quad of\quad associations\quad in\quad computer\quad partition}$$

Notes: 1). *Manual partition* – clustering partition created by domain expert. In this experiment, it's the pre-existing categorizations on SfN NDG website; 2). *Automatic partition* – clustering partition created by a computer program (Semantic Facilitator $^{TM\ SM}$ V2.0); 3). *Association* – a pair of objects belonging to the same cluster. 4) *Correct associations* – associations that appear in both the computer partition and the manual partition.

The overall clustering performance is measured by *F-measure* (Larsen and Aone 1999) (Stein and Eissen 2002) (Van Rijsbergen 1979). F-measure is a mechanism to provide for an overall estimate of the combined effect of Recall and Precision. F-measure is a standard evaluation metric in the field of information retrieval. The F-measure formula is expressed as:

$$F - Measure = \frac{(BETA\hat{}2 + 1) * CR * CP}{(BETA\hat{}2 * CP) + CR}$$

BETA is the relative importance of Recall vs. Precision such that a BETA value of 0 means that F-measure=Precision; BETA value of ∞ means that F-measure=Recall. (BETA=1 means Recall and Precision are equally weighted; BETA=0.5 means Recall is relatively less important than Precision; BETA=2.0 means Recall is relatively more important than Precision.) In our case, I choose BETA to be 1.0, assigning equal weight to recall and precision. The higher the F-measure value, the better the clustering result.

In the second experiment, user feedback is involved in the clustering process. The user selected clusters that need to be added back to the computer cluster to evaluate the overall clustering performance of the dataset. An adjusted CR, CP and F-measure are used in this process.

$$Adjusted\,CR = \frac{total\ \ number\ of\ \ correct\ \ associations\ \ in\ \ (computer partition + user\ feedback selection)}{total\ \ number\ of\ \ associations\ \ in\ \ (manual\ \ partition + user\ feedback selection)}$$

$$Ajusted\ CP = \frac{total\ \ number\ of\ \ correct\ \ associations\ \ in\ \ (computer partition + user\ feedback selection)}{total\ \ number\ of\ \ associations\ \ in\ \ (computer partition + user\ feedback selection)}$$

$$Adjusted\,f - measure = \frac{(BETA\verb|^|2+1) * AdjustedCR * AdjustedCP}{(BETA\verb|^|2 * AdjustedCP) + AdjustedCR}$$

Note: The clusters selected by user in the feedback process in annotated as “user feedback selection”.

5.4 Experiment Hardware and Software Settings

In the experiments, two sets of computer hardware components are used. One is a standalone PC, the other is a Grid computing infrastructure (a network of 14 PCs). Their configuration is listed as follows.

Stand alone PC: CPU: AMD Athlon 64 X2 5000+ (2.6 GHZ)

RAM: 3.0 GB DDR2 PC 4600

Operating System: Windows Vista Ultimate 32bit.

Grid Computing Infrastructure: A network of 14 PCs.

Each PC equips with Pentium III CPU (933 MHZ), 512 MB RAM.

Operating System: Unix/Linux

In both settings, Semantic Facilitator $^{TM\ SM}$ V2.0 is used as the software.

Note: The PCs in the Grid computing infrastructure are considerably less powerful than the stand alone PC.

5.5 Research Hypotheses

The testing hypotheses are derived from the three research questions (see chapter 1) and the proposed research approach. Hypothesis 1 tests the effectiveness of using a genetic algorithm in conjunction with clustering. Hypothesis 2 studies the improvement in the effectiveness of the approach through user feedback. Hypothesis 3 is concerned about the efficiency of the developed system.

*H1. Given the web-based resource of user interest and user perspective, the information system designed and implemented based on the proposed research approach – an information system that uses a genetic algorithm (I**S-GA**) in conjunction with SOM-clustering – can cluster the web pages (create a knowledge model) effectively. Moreover, the clustering performance is more effective than an identical information system (I**S**) that doesn't use a genetic algorithm.*

*H2. Given the web-based resource of user interest and user perspective, the enhanced information system (I**S-GA**) incorporating user feedback (I**S-GA-FB**) into the clustering process can more effectively cluster the web pages than a system that doesn't incorporate user feedback (I**S-GA**).*

*H3. Given the web-based resource of user interest and user perspective, the enhanced information system (I**S-GA**) that uses a Grid infrastructure (I**S-GA-GRID**) can more efficiently cluster the web pages than the one that doesn't using Grid* *infrastructure (**IS-GA**) and IS-GA using an appropriate Grid infrastructure can be expected to provide reasonably good performance .*

5.6 Experiment 1 Design & Result

Experiment 1 is conducted to test H1 (the effectiveness of the genetic algorithm on improving SOM clustering performance). H1 is listed as follows.

*H1. Given the web-based resource of user interest and user perspective, the information system designed and implemented based on the proposed research approach – an information system that uses a genetic algorithm (**IS-GA**) in conjunction with SOM-clustering – can cluster the web pages (create a knowledge model) effectively. Moreover, the clustering performance is more effective than an identical information system (**IS**) that doesn't use a genetic algorithm.*

In experiment 1, I conducted following activities.

1. Both the training and testing datasets are used in this experiment.
2. There are 8 predefined categorization schemes in the NDG website. The "categories" categorization scheme is randomly chosen by the computer as the user preferred perspective towards the NDG website.
3. Semantic Facilitator $^{TM\ SM}$ V2.0 (no GA component) is first used to organize the training dataset (create a knowledge model) based on the chosen user perspective. For each database web page, only the text in "Descriptions" and "Notes" sections are used in clustering process. Default [5](typical) Self-Organizing Maps (SOM) parameter values are used. The system's clustering (categorization) result is

[5] This default SOM parameter values have been used in another domain (Liang et al. 2006) and it performed reasonably well.

compared to the standard knowledge model (pre-existing categorizations on the NDG web site). Cluster recall, cluster precision and F-measure are calculated.

4. Using the same settings as step 3, a genetic algorithm component is added to search for good SOM parameter values. I then re-run SOM clustering using the parameter value discovered by the genetic algorithm and cluster recall, cluster precision and F-measure are calculated.
5. The performances of Semantic Facilitator $^{TM\,SM}$ V2.0 without the genetic algorithm component and the system with the genetic algorithm component are compared.
6. Using the SOM parameter values discovered in step 2, run Semantic Facilitator ^{TM} ^{SM} V2.0 on the testing dataset. The clustering (categorization) result is compared to standard answers and cluster recall; cluster precision and f measure are calculated.

Table 5.2 lists the genetic algorithm (GA) parameter values used in this experiment. The number of generations and the population size are set to large and the threshold of convergence is set to small to make sure that the GA can explore a sufficiently large pool of SOM parameter values. Percentage of permutation and percentage of crossover are also tuned to fit the dataset used in this experiment and ensure good GA performance. In this experiment, four SOM parameter values are varied by GA: x dimension, y dimension, neighborhood size, final iteration (see table 5.3). Their ranges are set sufficiently large enough so that all good SOM parameter values sets can be included in the search pool of GA approach.

Table 5.2 Experiment 1 Genetic Algorithm Parameter Values

Number of generation	Population size	Percentage of Permutation	Percentage of crossover	Threshold of convergence
50	120	0.14	0.8	0.01

Table 5.3 Experiment 1 SOM Parameter Values Ranges

x dimension	y dimension	Neighborhood size	Final iteration
3 – 10	3 – 10	2 – 6	5000 – 60000

The SOM map generated by Semantic Facilitator $^{TM\ SM}$ V2.0 with the GA component is shown on figure 5.3. The results of the experiment for the training dataset are showed in table 5.4 and the results of the testing dataset are listed in table 5.5. The experimental results show that IS-GA (using a genetic algorithm along with SOM-clustering) provides a reasonably good clustering performance on the dataset of user interest. Only limited information (a part of the textual information on the web pages) is used in computer clustering process and there is no human intervention involved. Comparing the results of similar studies ((Liang, Vaishnavi et al. 2006) - deals with directory metadata, (Li et al. 2006) - deals with a set of faculty web pages), the clustering result of IS-GA is reasonably good (the F-measure of the similar studies is in the range of 0.4-0.5).

The t-tests on table 4 and table 5 showed that the performance measures IS-GA and IS are significantly different. The clustering performance of IS-GA is much better than that of IS (without the GA component). For the training dataset, the cluster recall is 0.4138 vs. 0.1379; the cluster precision is 0.4615 vs. 0.1739, and the F-measure 0.4364 vs. 0.1538. IS-GA also performs well on the testing dataset (F-measure 0.32 vs. 0.1143) [6]. Thus, Hypothesis 1 can be considered to be supported.

In addition, the IS-GA approach provides a systematically way to find a good set of SOM parameter values: SOM parameter values are usually domain-specific and need to be discovered in an ad hoc manner, but now this discovery can be done automatically by the GA component.

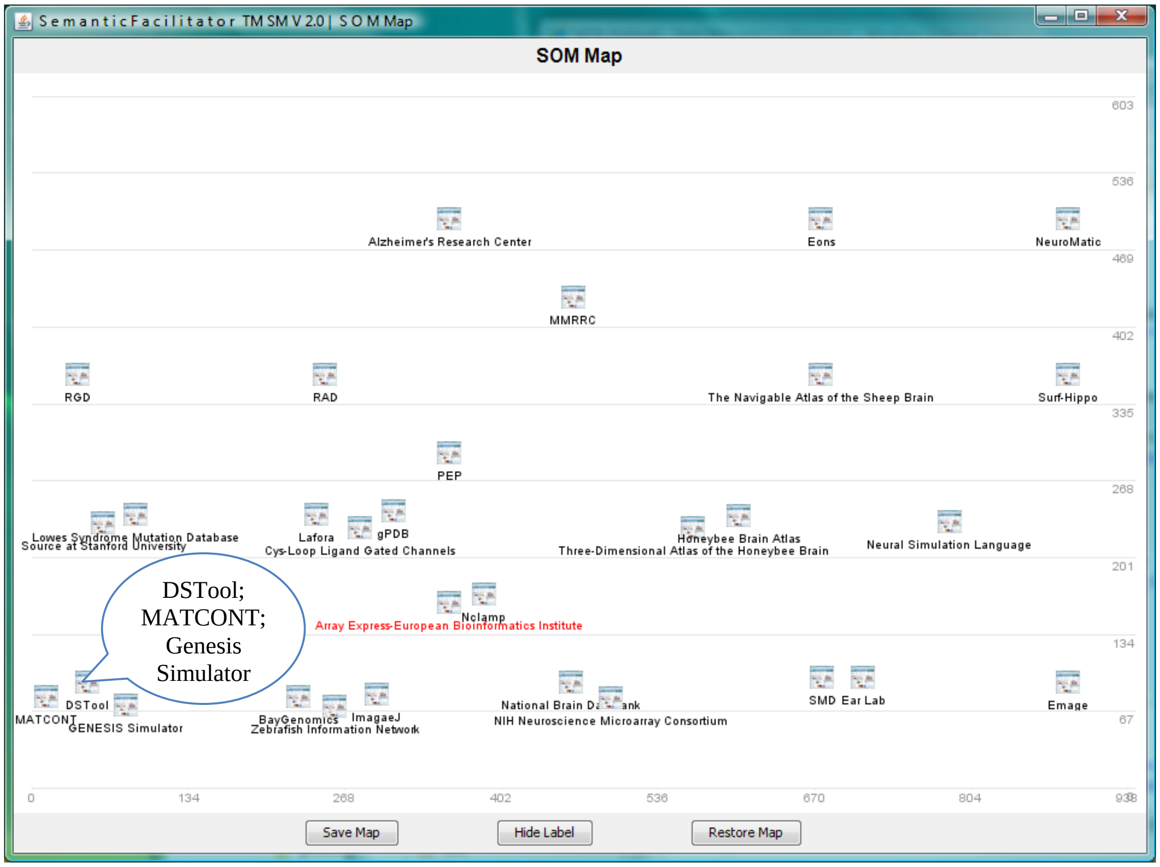

SemanticFacilitator TM SM V 2.0 | SOM Map
SOM Map
Alzheimer's Research Center
Eons
NeuroMatic
MMRRC
RGD
RAD
The Navigable Atlas of the Sheep Brain
Surf-Hippo
PEP
Lowes Syndrome Mutation Database
Source at Stanford University
Lafora
gPDB
Cys-Loop Ligand Gated Channels
Honeybee Brain Atlas
Three-Dimensional Atlas of the Honeybee Brain
Neural Simulation Language
DSTool; MATCONT; Genesis Simulator
Nclamp
Array Express-European Bioinformatics Institute
DSTool
MATCONT
GENESIS Simulator
BayGenomics
ImagaeJ
Zebrafish Information Network
National Brain Databank
NIH Neuroscience Microarray Consortium
SMD
Ear Lab
Emage
603
536
469
402
335
268
201
134
67
0
134
268
402
536
670
804
938
Save Map
Hide Label
Restore Map

Figure 5.3 Experiment 1 SOM Map

Note: The oval callout is manually added to show the database webpage names in a cluster clearly.

Table 5.4 Experiment 1 Result for Training Dataset

[6] The F-measure deteriorates a little bit for testing dataset comparing to F-measure of training dataset. This is probably because testing dataset isn't a good representative of training dataset. However, the IS-GA performance is much higher than the performance IS.

Experiment Settings	SOM Parameter Values				Cluster Recall	Cluster Precision	F-measure
	x dim	y dim	Neighborhood size	Final iteration			
IS (no GA)	7	9	2	10000	0.1379	0.1739	0.1538
IS – GA	5	8	3	48116	0.4138	0.4615	0.4364
T Test of IS (no GA) vs. IS-GA					P (sig.)	0.0000	

Note: 1) x dim – x dimension; y dim – y dimension
2) t-test: two-sample assuming unequal variances; α = 0.05, two-tail.

Table 5.5 Experiment 1 Results for Testing Dataset

Experiment Settings	SOM Parameter Values				Cluster Recall	Cluster Precision	F-measure
	x dim	y dim	Neighborhood size	Final iteration			
IS (no GA)	7	9	2	10000	0.1818	0.0833	0.1143
IS – GA	5	8	3	48116	0.3636	0.2857	0.3200
T Test of IS (no GA) vs. IS-GA					P (sig.)	0.0059	

Note: 1) x dim – x dimension; y dim – y dimension
2) t-test: two-sample assuming unequal variances; α = 0.05, two-tail.

5.7 Experiment 2 Design and Results

In experiment 2, I studied the effect of user feedback on clustering performance of the proposed approach. In this experiment, a simulated domain expert (neuroscientist) instead of a real human user is used to give feedbacks to Semantic Facilitator $^{TM\ SM}$ V 2.0 (the justification is illustrated in section 5.1 experiment general design). Hypothesis 2 is tested in this experiment.

*H2. Given the web-based resource of user interest and user perspective, the enhanced information system (**IS-GA**) incorporating user feedback (**IS-GA-FB**) into the clustering process can more effectively cluster the web pages than a system that doesn't incorporate user feedback (**IS-GA**).*

Table 5.6 Training Dataset Categorization from NDG Website

Database Name	**Category**
Alzheimer’s Research Center	Data storage
Array Express-European Bioinformatics Institute	Microarrays
NIH Neuroscience Microarray Consortium	
SMD	
BayGenomics	Cellular/Molecular
Zebrafish Information Network	Genetics
Lowes Syndrome Mutation Database	
Source at Stanford University	
Lafora	
ImagaeJ	Imaging
Cys-Loop Ligand Gated Channels	Physiology
MMRRC	Model
Nclamp	Software
NeuroMatic	
PEP	3D Model
gPDB	Cellular/Molecular & Genetics
National Brain Databank	
The Navigable Atlas of the Sheep Brain	Atlas & Anatomy
Ear Lab	Model & Software
Eons	
Emage	3D Model Anatomy & Genetics
RGD	Genetics, Microarray, & Ontology
RAD	
DSTool	Simulation & Software
Neural Simulation Language	
MATCONT	
GENESIS Simulator	
Three-Dimensional Atlas of the Honeybee Brain	3D Model Anatomy & Atlas
Honeybee Brain Atlas	
Surf-Hippo	Brain Mapping, cellular/Molecular & Software

Table 5.7 Testing Dataset Categorization from NDG Website

Database	**Category**
Ki-Databases	Cellular/Molecular
Uniprot	Genetics
Duke/Southhampton Archive of Neuronal Morphology	Model
FlyTrap	3D Model Anatomy & Genetics
IATR	Software
EEG Lab	
FlexProt	3D Model
Protein Data Bank	
Visiome	Neuroinformatics
Brede	
Hereditary Hearing Loss Homepage	Cellular/Molecular & Genetics
SynDB	
HyperBrain Brain Atlas	Atlas & Anatomy
Brain Architecture Center	3D Model Anatomy & Atlas
Mouse Brain Atlas	
FMR Lab	Brain Mapping, cellular/Molecular & Software

In experiment 2, I conducted following activities.

1. Both the training and testing datasets are used in this experiment.
2. There are 8 predefined categorization schemes in the NDG website. The "categories" categorization scheme is randomly chosen by the computer as the user preferred perspective towards the NDG website.
3. Semantic Facilitator $^{TM\ SM}$ V2.0 (with genetic algorithm component) is first used to cluster the training dataset based on the chosen user perspective. In each database web page, only the text in "Descriptions" and "Notes" sections are used

in clustering process. The dataset is first clustered without user feedback and an initial SOM map is generated.

4. Genetic algorithm (GA) setting: Since GA is computational intense (it's may take more than 20 hours to run if I try to let GA search the best SOM parameter values for clustering), compared to the GA setting in experiment 1, a relatively small population size and number of generation is used: 100 vs. 120 and 30 vs. 50 (Other GA parameter values remain the same with the ones in experiment 1). The running time of GA reduces to around 10 hours and the results are still reasonably good.
5. A simulated domain expert makes adjustments on the initial clustering. In this experiment setting, the simulated domain expert changes 10%, 20%, or 30% of total database pages in the clustering respectively (the selected database pages are listed in table 5.8). The database pages are carefully chosen so that they don't share clusters with the remaining dataset. The selection is done randomly among all candidate clusters. Based on known clustering answers, the selected database pages are moved to correct clusters and taken out the dataset. Re-run Semantic Facilitator $^{TM\ SM}$ V 2.0 on the remaining dataset.
6. The clustering (categorization) result of the computer is compared to the known clustering answers and cluster recall, cluster precision and f measure are calculated. The chosen clusters in step 2 are added back to the dataset and an adjusted metric (F-measure) are calculated. The equations for calculate adjust F-measure is listed as follows.

7. The performance metrics are compared with metrics generated in step 1 including testing if system performance increases as the amount of user feedback increases.
8. Using the SOM parameter values discovered in step 2, run Semantic Facilitator TM SM V2.0 on testing dataset. The clustering (categorization) result is compared to standard answers and cluster recall, cluster precision and f measure are calculated.

Table 5.8 User Selected Web Pages for Feedback

Percentage of feedback	**Database URLs**	**Category**
10%	http://www.cis.gsu.edu/~lli/data/28948.htm	1
	http://www.cis.gsu.edu/~lli/data/28941.htm	2
	http://www.cis.gsu.edu/~lli/data/29108.htm	2
20%	http://www.cis.gsu.edu/~lli/data/28948.htm	1
	http://www.cis.gsu.edu/~lli/data/28941.htm	2
	http://www.cis.gsu.edu/~lli/data/29108.htm	2
	http://www.cis.gsu.edu/~lli/data/28833.htm	3
	http://www.cis.gsu.edu/~lli/data/28943.htm	4
	http://www.cis.gsu.edu/~lli/data/28942.htm	4
30%	http://www.cis.gsu.edu/~lli/data/28948.htm	1
	http://www.cis.gsu.edu/~lli/data/28941.htm	2
	http://www.cis.gsu.edu/~lli/data/29108.htm	2
	http://www.cis.gsu.edu/~lli/data/28833.htm	3
	http://www.cis.gsu.edu/~lli/data/28943.htm	4
	http://www.cis.gsu.edu/~lli/data/28942.htm	4
	http://www.cis.gsu.edu/~lli/data/28863.htm	5
	http://www.cis.gsu.edu/~lli/data/29011.htm	6
	http://www.cis.gsu.edu/~lli/data/29097.htm	6

Note: The database pages are in the same cluster if their category number is the same. The number itself doesn't contain any special meaning.

5.7.1 Experiment Results and Analysis

Figure 5.4 shows the SOM map generated by the system without any user feedback. A simulated user reviews the clusters (e.g., double click the icon to open the corresponding page and review the description about the page). The user finds that one database web page (ImageJ) should standalone instead of staying together with other database pages in a cluster. The user further finds that two database pages should be put together instead of being put in different clusters (gPDB and National Brain Databank). Those findings are indicated by red over callouts in figure 5.4. The user can make the necessary changes to the map by dragging and dropping the nodes (database web pages) to his/her desired location. The user has now made a 10% modification to the cluster. The Semantic Facilitator $^{TM\ SM}$ V2.0 is re-run on the remaining dataset. The database pages identified by the user in the earlier step are added back to the SOM map and a new map is generated (figure 5.5).

The SOM Map for 20% user feedback and 30% user feedback are shown in figure 5.6 and figure 5.7 respectively. The red oval callouts in the figure 5.5, 5.6, and 5.7 shows the clusters manually identified by the users and demonstrate the database web page names in the selected clusters. From those figures, I can see that the SOM map is different each time when user provides feedback for the same dataset. However, the database pages that user indicated to be together are clustered together.

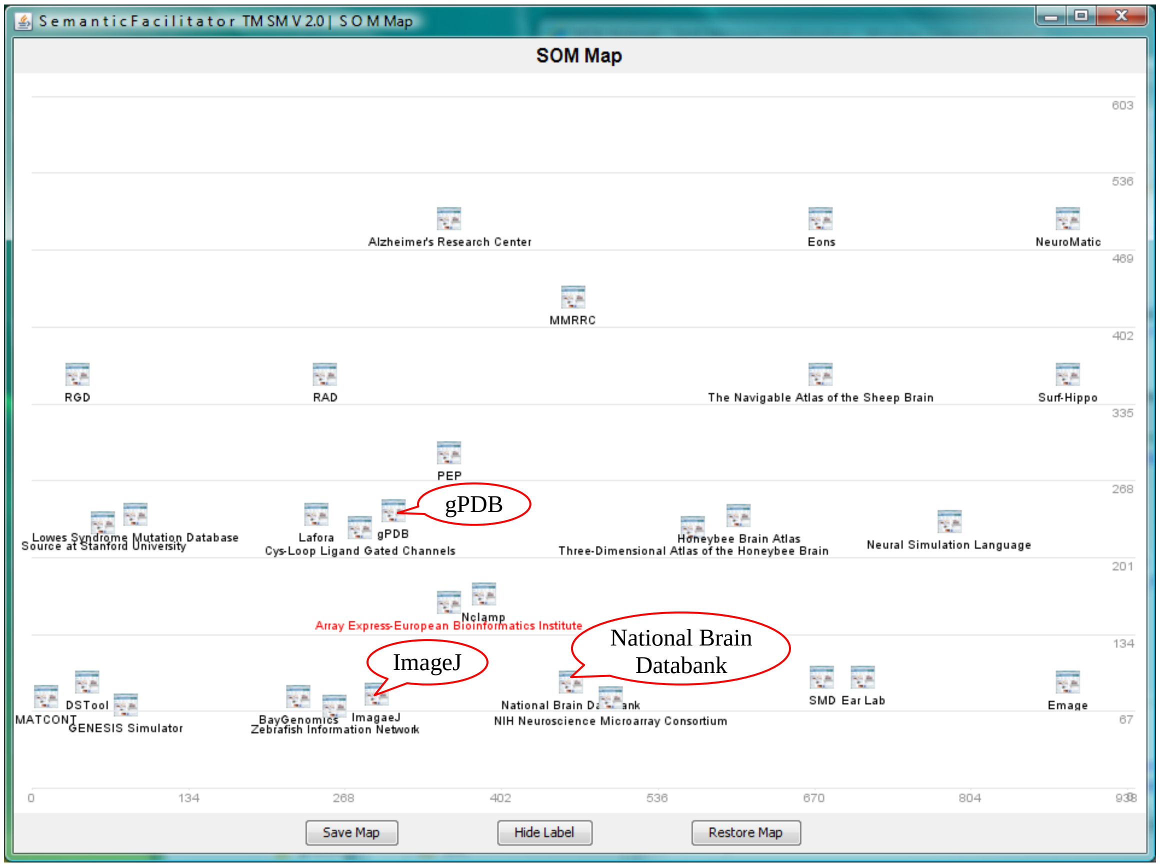

SemanticFacilitator TM SM V 2.0 | SOM Map
SOM Map
Alzheimer's Research Center
Eons
NeuroMatic
MMRRC
RGD
RAD
The Navigable Atlas of the Sheep Brain
Surf-Hippo
PEP
gPDB
Lowes Syndrome Mutation Database
Source at Stanford University
Lafora
gPDB
Cys-Loop Ligand Gated Channels
Honeybee Brain Atlas
Three-Dimensional Atlas of the Honeybee Brain
Neural Simulation Language
Nclamp
Array Express-European Bioinformatics Institute
National Brain Databank
ImageJ
DSTool
MATCONT
GENESIS Simulator
BayGenomics
ImagaeJ
Zebrafish Information Network
National Brain Databank
NIH Neuroscience Microarray Consortium
SMD
Ear Lab
Emage
603
536
469
402
335
268
201
134
67
0
134
268
402
536
670
804
938
Save Map
Hide Label
Restore Map

Figure 5.4 SOM Map for System Run without User Feedback

Note: The Oval callouts are manually added to show the names of the database web pages clearly.

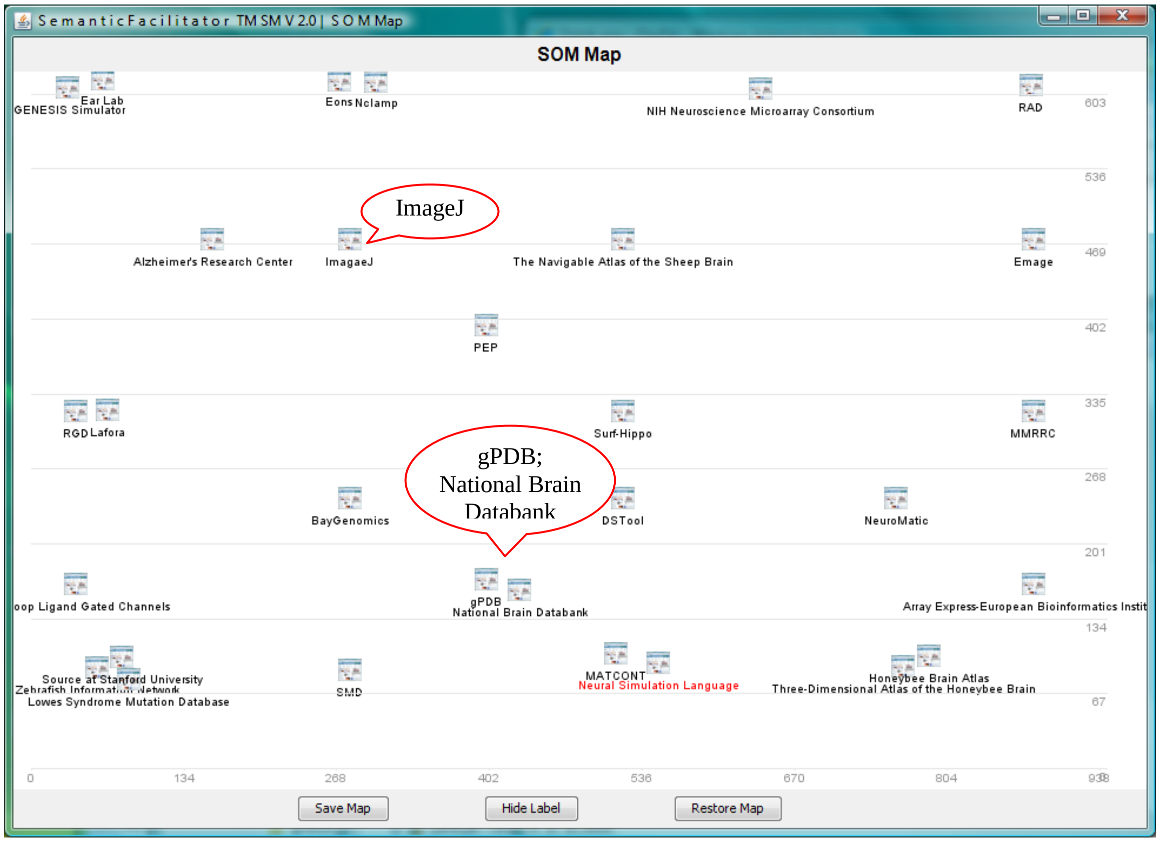

SemanticFacilitator TM SM V 2.0 | SOM Map
SOM Map
GENESIS Simulator
Ear Lab
Eons Nclamp
NIH Neuroscience Microarray Consortium
RAD
ImageJ
Alzheimer's Research Center
ImageJ
The Navigable Atlas of the Sheep Brain
Emage
PEP
RGD Lafora
Surf-Hippo
MMRRC
gPDB; National Brain Databank
BayGenomics
DSTool
NeuroMatic
oop Ligand Gated Channels
gPDB
National Brain Databank
Array Express-European Bioinformatics Instit
Source at Stanford University
Zebrafish Information Network
Lowes Syndrome Mutation Database
SMD
MATCONT
Neural Simulation Language
Honeybee Brain Atlas
Three-Dimensional Atlas of the Honeybee Brain
603
536
469
402
335
268
201
134
67
0
134
268
402
536
670
804
938
Save Map
Hide Label
Restore Map

Figure 5.5 SOM Map from System Run with 10% User Feedback

Note: The red oval callouts are manually added to show the database web pages name in the clusters. Those clusters are formed by user during user feedback process.

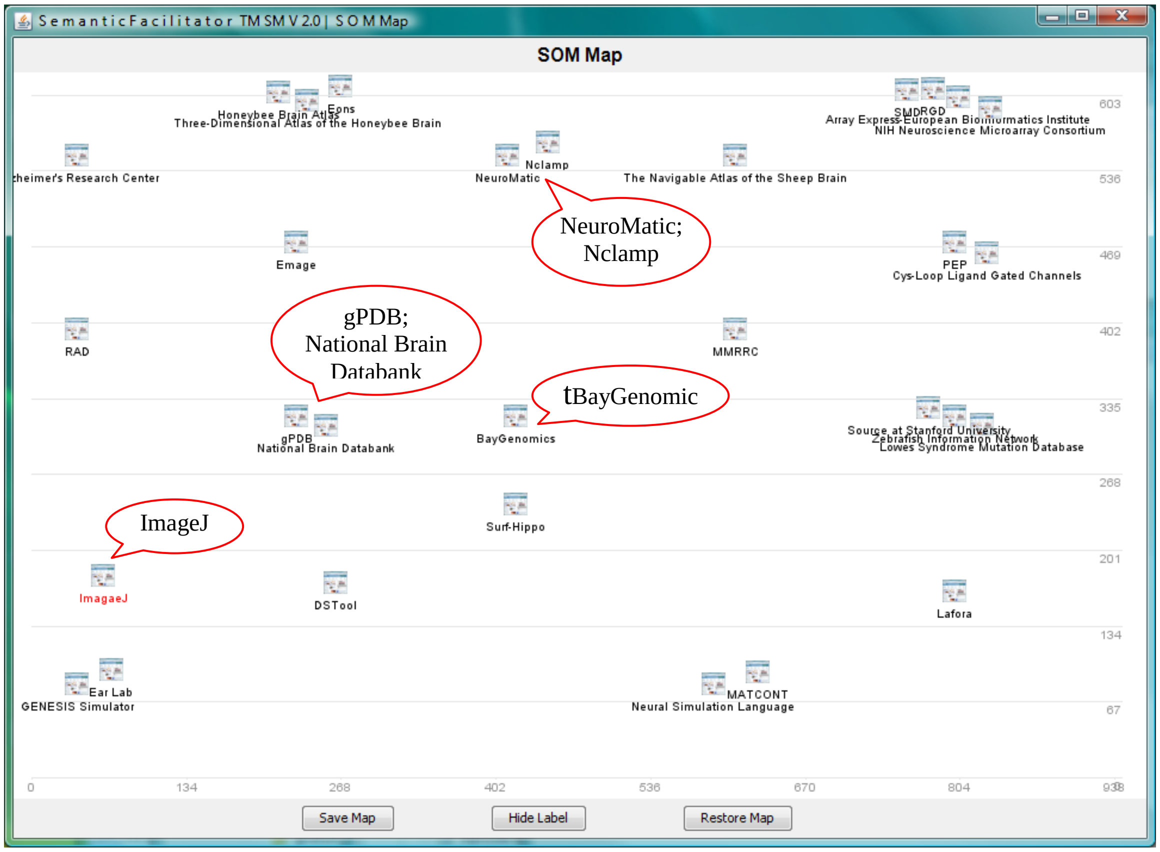

SemanticFacilitator TM SM V 2.0 | SOM Map
SOM Map
Honeybee Brain Atlas
Three-Dimensional Atlas of the Honeybee Brain
Eons
SMD
RGD
Array Express
European Bioinformatics Institute
NIH Neuroscience Microarray Consortium
Alzheimer's Research Center
NeuroMatic
Nclamp
The Navigable Atlas of the Sheep Brain
NeuroMatic; Nclamp
Emage
PEP
Cys-Loop Ligand Gated Channels
RAD
gPDB; National Brain Databank
MMRRC
tBayGenomic
gPDB
National Brain Databank
BayGenomics
Source at Stanford University
Zebrafish Information Network
Lowes Syndrome Mutation Database
ImageJ
Surf-Hippo
ImagaeJ
DSTool
Lafora
Ear Lab
GENESIS Simulator
MATCONT
Neural Simulation Language
603
536
469
402
335
268
201
134
67
0
134
268
402
536
670
804
938
Save Map
Hide Label
Restore Map

Figure 5.6 SOM Map from System Run with 20% User Feedback

Note: The red oval callouts are manually added to show the database web pages name in the clusters. Those clusters are formed by user during user feedback process.

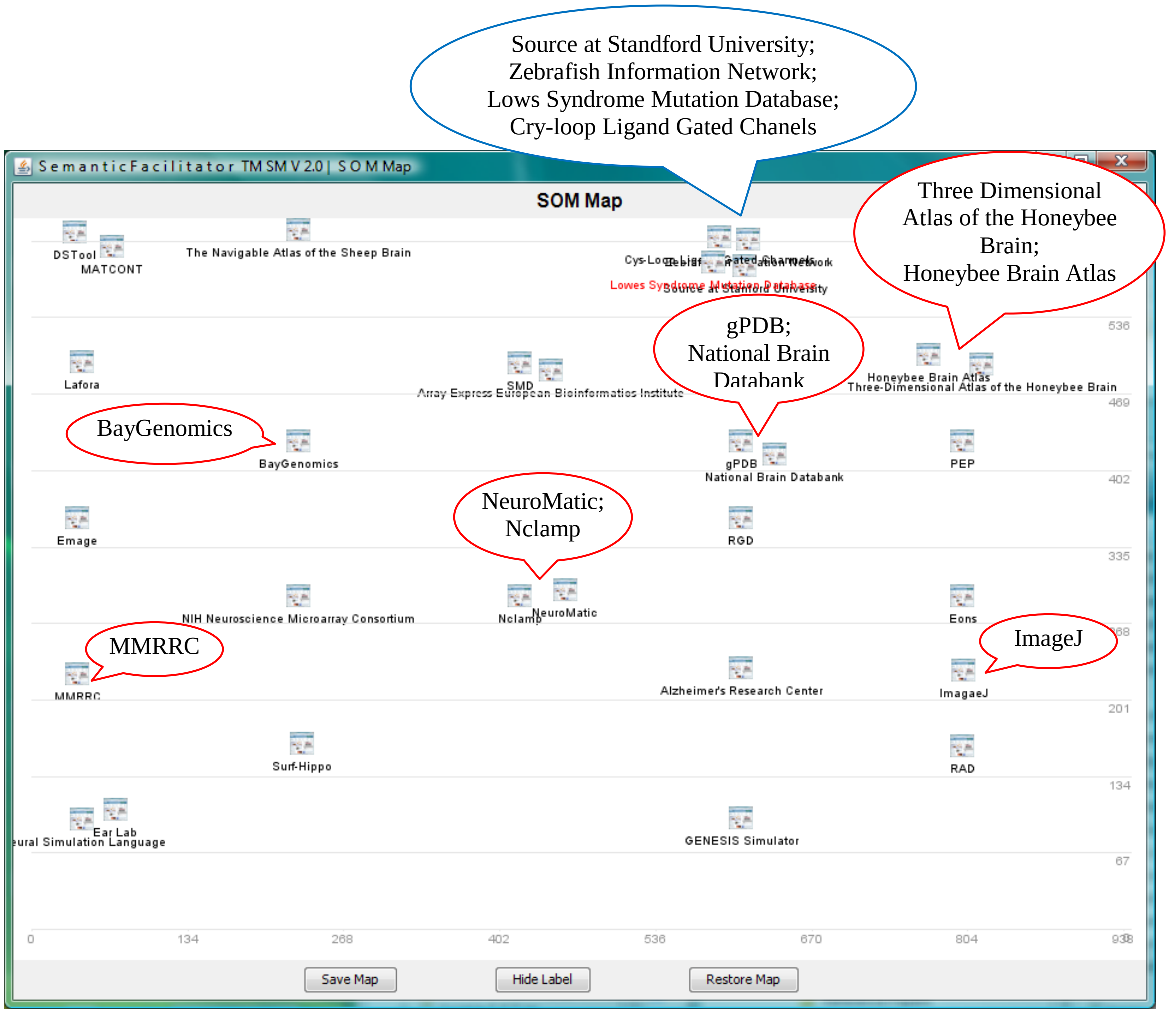

SemanticFacilitator TM SM V 2.0 | SOM Map
SOM Map
Source at Standford University; Zebrafish Information Network; Lows Syndrome Mutation Database; Cry-loop Ligand Gated Chanels
Three Dimensional Atlas of the Honeybee Brain; Honeybee Brain Atlas
gPDB; National Brain Databank
BayGenomics
NeuroMatic; Nclamp
MMRRC
ImageJ
DSTool
MATCONT
The Navigable Atlas of the Sheep Brain
Lafora
SMD
Array Express European Bioinformatics Institute
Honeybee Brain Atlas
Three-Dimensional Atlas of the Honeybee Brain
BayGenomics
gPDB
National Brain Databank
PEP
Emage
RGD
NIH Neuroscience Microarray Consortium
Nclamp
NeuroMatic
Eons
MMRRC
Alzheimer's Research Center
ImageJ
Surf-Hippo
RAD
Ear Lab
Neural Simulation Language
GENESIS Simulator
Save Map
Hide Label
Restore Map

Figure 5.7 SOM Map from System Run with 30% User Feedback

Note: The red oval callouts are manually added to show the database web pages name in the clusters. Those clusters are formed by user during user feedback process. The blue oval callout is manually added to show the web page names in a crowded cluster.

The results of experiment 2 are summarized in table 5.9 and figure 5.8. Table 5.9 and figure 5.8 show that the clustering performance of Semantic Facilitator $^{TM\ SM}$ V2.0 is significantly improved with the user feedback. The metrics measures (cluster recall, cluster precision, F-measure and adjusted F-measure) of IS-GA-FB (system using user feedback) are significantly different from IS-GA (system without using user feedback). The two systems differentiate at the 0.05 level for the training dataset and the testing dataset. The IS-GA-FB (10% feedback) performs much better than IS GA (without feedback), e.g., adjusted F-measure for system with 10% feedback and without feedback is 0.5660 vs. 0.4074. This is a 40% increase.

Although the t-tests show there is no significant difference in evaluation metrics among 10% vs. 20% and 20% vs. 30%, the overall trend is that the metrics are getting better. If I can consider adjusted F-measure, when user feedback increases (from 10% to 20% to 30%), the clustering performance (adjusted F-measure) of the system also increases (from 0.566 to 0.6071to 0.6415). So, it seems that the more feedback the user provided to the system, the better clustering performance the system produces.

However, it's not realistic to expect web users to provide a lot of feedback to the system since this would require a significant commitment. This could greatly reduce the usefulness of the research approach so a moderate feedback percentage would be preferred in this case.

For the testing dataset, the F-measure values reduce a little bit as the amount of the user feedback increases. But such value reduction isn't statistically significant as illustrated in table 5.9. The deterioration of system performance probably because of the insufficient system training (There are less dataset for training when user feedback increases). However, system's performance with user feedback is still much better than the ones generated from the system without user feedback. From this standpoint, I can conclude that it's possible to use the SOM parameter values discovered in one dataset (training dataset) to a different dataset in a similar domain (testing dataset) and they still performs well.

For the testing dataset, user feedback seems to have negative impact on the system performance.

In summary, the user feedbacks can greatly help the system's clustering performance. Thus, hypothesis 2 is supported in this experiment.

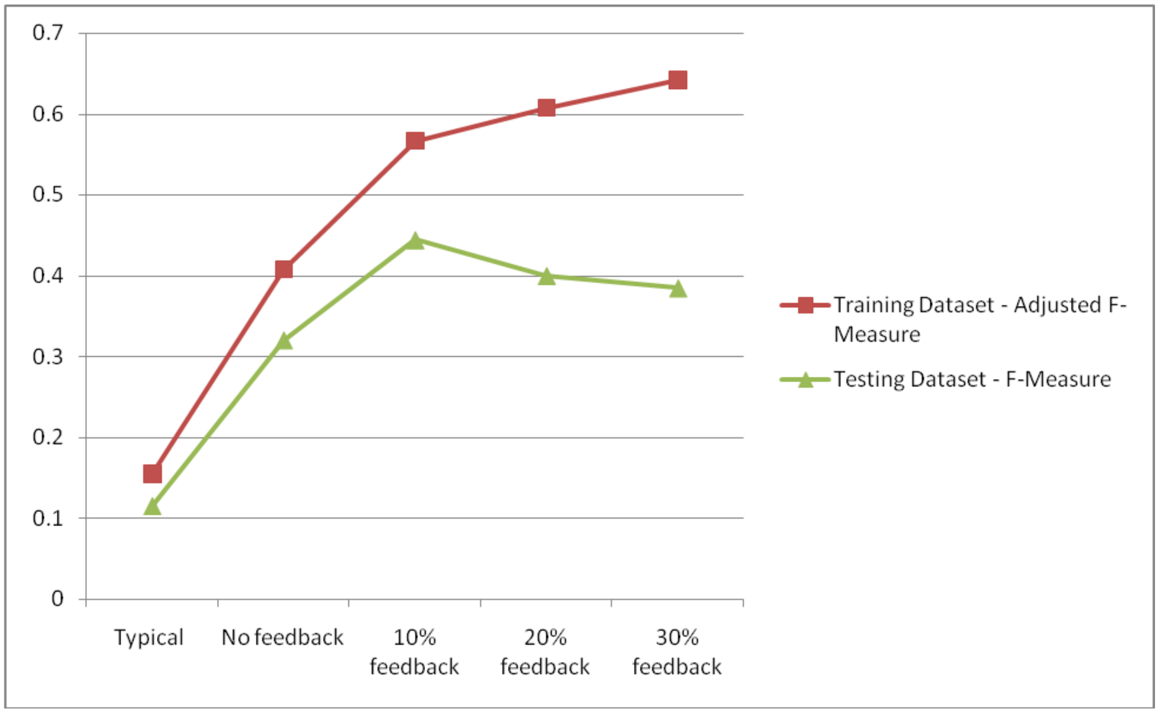


Figure 5.8 Experiment 2 – F-measure Values

Table 5.9 Experiment 2 Results

Training Dataset								Testing Dataset		
Use Feedback	x dim	y dim	Neighborhood size	Final iteration	Adjusted Cluster Recall	Adjusted Cluster Precision	Adjusted F-measure	Cluster Recall	Cluster Precision	F-measure
Typical SOM parameter values*	7	9	2	10000	0.1379	0.1739	0.1538	0.1818	0.0833	0.1143
No feedback*	9	9	2	13491	0.3793	0.4400	0.4074	0.3636	0.2857	0.3200
10% feedback	8	8	4	25804	0.5172	0.6250	0.5660	0.5455	0.3750	0.4444
20% feedback	5	8	4	30971	0.5862	0.6296	0.6071	0.4545	0.3571	0.4000
30% feedback	5	9	4	31135	0.5862	0.7083	0.6415	0.4544	0.3333	0.3846
t-test pairs					**Training Dataset p value (sig.)**			**Testing Dataset p value (sig.)**		
Typical SOM parameter values vs. No feedback					0.0062			0.0059		
No feedback vs. 10% feedback					0.0206			0.0938		
10% feedback vs. 20% feedback					0.3377			0.4358		
20% feedback vs. 30% feedback					0.4201			0.7855		

Note: 1) x dim – x dimension; y dim – y dimension

2) t-test: two-sample assuming unequal variances; α = 0.05, two-tail.

3)* There are no user feedback involved in “Typical SOM parameter values” and “No feedback” settings. So, the adjusted metrics in those settings (recall, precision and F-measure) are the same as the original metrics.

5.8 Experiment 3 Design and Results

In experiment 3, I mainly focus on the efficiency of the proposed approach. Grid computing infrastructure is introduced to reduce the running time of the proposed approach.

*H3. Given the web-based resource of user interest and user perspective, the enhanced information system (I**S-GA**) that uses a Grid infrastructure (I**S-GA-GRID**) can more efficiently cluster the web pages than the one that doesn't use a Grid infrastructure (I**S-GA**) and IS-GA using an appropriate Grid infrastructure can be expected to provide reasonably good performance* .

I conducted the following activities in the experiment.

1. Only the training dataset is used in this experiment since the efficiency of the prototype is the major concern.
2. There are 8 predefined categorization schemes in NDG website. The "categories" categorization scheme is randomly chosen by the computer as the user preferred perspective towards the NDG website.
3. Semantic Facilitator $^{\text{TM SM}}$ is used to cluster the training dataset based on the chosen user perspective. In each database web page, only the text in "Descriptions" and "Notes" sections are used in clustering process. GA is used to search good SOM parameter values. A SOM map is generated.
4. The clustering (categorization) result is compared to standard answers. Cluster recall, cluster precision and f measure are calculated and the running time is recorded.

5. Repeat steps 1-3 on Grid computing infrastructure and record the new turnaround time.
6. Compare the turnaround time of both runs.

The result of the experiment is shown in table 5.10.

Table 5.10 Experiment 3 Result

Setting	**Processing time**
IS-GA	10 hours, 40 minutes
IS-GA-Grid	3 hours 32 minutes

Table 5.10 shows the processing time of the IS-GA-Grid (system with Grid computing infrastructure) is much less than the processing time of the IS-GA (system without Grid computing infrastructure). So, hypothesis 3 is supported.

The Grid infrastructure used in this experiment is a small scaled laboratory implementation (a network of 13 old computers, each computer is considerably less powerful than the standalone PC used in this experiment). The processing time of the research prototype can be greatly reduced if a more powerful Grid implementation is used.

5.9 Research Findings

In the chapter, I designed and conducted 3 experiments to evaluate the correctness of the proposed approach. The experimental results show that the proposed research approach can effectively and efficiently organize web-based resources (create knowledge models) based on user preferred perspective. The user feedback can greatly improve the quality of the knowledge models (clustering performance) of the proposed approach. The more feedback the user provides, the better the overall clustering performance (adjusted f-measure) becomes. There are two important findings in this phase.

1) Simulation-based experiment is an alternative way of evaluating computer clustering performance for the research community. The web information management community commonly uses domain experts to evaluate the clustering results. This could be a difficult task in many cases because it’s often hard to gain access to domain experts. When domain experts are available, they usually tend to have different opinions on whether a cluster is good or not which makes getting a consensus from a group of experts difficult. In this dissertation, I purposely choose a web site that is already organized by a panel of experts and use a computer simulated expert as the perfect representative of the group of panelists. This approach is not only cost-effective, but also allows me to evaluate the computer generated clustering more objectively.

2) In the experiments, I divided the dataset into a training dataset and a testing dataset. The experiments results showed that when applying a trained system to the testing dataset, the system still performs well even though the metrics deteriorate a little bit. This is probably because the training dataset isn’t a very good representative of the application domain. If a set of good training dataset could be identified or I can have sufficient data to training data, the trained system can be used to organize any new data in the same application domain. In this case, it’s possible to develop a fully automated system (without human intervention in model building process) for dynamic and adaptable knowledge models generation. This would be a breakthrough in web information management area. I took a first step in this direction.

Chapter 6: Conclusion

In this chapter, I first summarize the research efforts. Then I discuss the contributions in the course of the research phases. I conclude the dissertation by analyzing limitations and discussing future directions for the research.

6. 1 Summary of Research Efforts

In the current Internet age, more and more people, organizations, and businesses rely on the web to share and search for information. Static and inflexible knowledge models (categorization structures) of web-based resources (a website or a set of web pages) have become a major challenge for web users to successfully use and understand the information on the web.

In this dissertation, I introduced a research approach to generate *user-centric dynamic* and *adaptable* knowledge models for web-based resources. The u*ser-centric* feature means knowledge models are created based on user specified perspective and users' feedback can be incorporated into the model building process. The *dynamic* feature means knowledge models are built on the fly. The *Adaptable* feature means users have control of the adaptation process by specifying their perspective on the web resource of interest.

By innovatively integrating clustering, visualization and enabling techniques such as genetic algorithm and Grid computing infrastructure, the proposed approach can create appropriate knowledge models for the web-based resources of interest while adapting to different user perspectives. A research prototype, Semantic Facilitator $^{TM\ SM}$ V2.0, is developed and three experiments are conducted to evaluate the research approach.

The experimental results show that the proposed research approach can effectively and efficiently organize web-based resources (create knowledge models) based on user preferred perspective. The user feedback can greatly improve the quality of the knowledge models (clustering performance) of the proposed approach. The more feedback the user provides, the better the overall clustering performance (adjusted f-measure) becomes.

During the course of the research, I followed the general methodology of design science research; five design phases from the General Design Cycle, *awareness of problem*, *suggestion*, *development*, *evaluation*, and *conclusion* were executed sequentially and iteratively. The *suggestion*, *development and evaluation* are the key design phases. The research findings of those phases are discussed in details in the following section.

6.2. Research Contributions

6.2.1 Research Findings in suggestion phase

In this phase, following a thorough literature review and critical analysis on web user adaptation systems, I proposed a web user adaptation systems framework and three dimensions for evaluating different adaptation systems. The framework and evaluation dimensions not only provide a solid foundation for designing a tentative solution in this particular research, but also can be a useful tool for the scholars who work in related areas to shape their research. In addition, my analysis on web information management systems provides the research community a fairly complete overview on how knowledge models are built on web systems.

6.2.2 Research Findings in the development phase

The *Development* phase is one of the most important phases in the research process because artifacts are the most salient feature of design science research. Two IT artifacts are created in this phase: a research model/approach and a research prototype, Semantic Facilitator $^{TM\ SM}$ V2.0. Those two IT artifacts are the main contributions of this dissertation.

The proposed research contributes to the web information management community in the following ways.

First, as a type of web user adaptation system, the proposed research approach uniquely focuses on providing user-centric and in-depth user adaptation. Current adaptive systems have been very successful in performing adaptation autonomously but the adaptation has been done on the back end using many server side resources, and the system often can’t accurately react to user’s specific adaptation needs especially when such needs can’t be modeled by the user. Existing adaptable systems can accurately reflect user needs, but often offer low-level adaptation. Based on my knowledge, this study is the first one to bring user-centric in-depth adaptation to web-based resources. Thus, the proposed approach fills the knowledge gap in web user adaptation systems.

Second, the research approach also provides a systematic and efficient way to create a dynamic knowledge model for a web-based resource by integrating clustering technique with a genetic algorithm and Grid technology.

Third, the proposed approach provides a generic approach for knowledge presentation and discovery. In this dissertation, I focus on a web-based dataset, and the research approach fundamentally deals with textual data. The proposed approach could be easily adapted to other application domains, such as database domain, project portfolio management, by converting the new datasets into a textual format.

The developed research prototype, Semantic Facilitator $^{TM\ SM}$ V2.0, can benefit both web users and IS research field.

On the one hand, the research prototypes provide a good tool for web users to better use and understand the information on the web. For example, a neuroscientist can use the prototype to more effectively organize and utilize the database information on the NDG website. On the other hand, the research prototype can act as a work bench for IS researchers to conduct more studies. For example, I used the prototype to evaluate the proposed research approach. For another example, an IS researcher can study the cognitive impact of the system to web users by setting up experiments and watching users interacting with the system. This would lead to some interesting behavior science type of research.

6.2.3 Research Findings in Evaluation Phase

The web information management community commonly uses domain experts to evaluate the clustering results. This could be a difficult task in many cases because it’s often hard to gain access to domain experts. When domain experts are available, they usually tend to have different opinions on whether a cluster is good or not, which makes getting a consensus from a group of experts difficult. In this dissertation, I purposely chose a web site that was already organized by a panel of experts and used a computer simulated expert as the perfect representative of the group of panelists. This approach is not only cost-effective, but also allows me to evaluate the computer generated clustering more objectively. This approach provides an alternative way of evaluating computer clustering performance for the research community.

In this dissertation, I divided the dataset into a training dataset and a testing dataset. The experiment showed that when applying a trained system to the testing dataset, the system still performs well even though the metrics deteriorate a little bit. This is probably because the training dataset isn't a very good representative of the application domain. If a set of good training dataset could be identified or I can have sufficient data to training data, the trained system can be used to organize any new data in the same application domain. In this case, it's possible to develop a fully automated system (without human intervention in model building process) for dynamic and adaptable knowledge models generation. This would be a breakthrough in web information management area. This dissertation took a first step in this direction.

6.2.4. Other contributions

This dissertation demonstrated how a design science research can be done in the IS field by following the directions in the general design science methodology. Thus, the dissertation can be used as an exemplar for IS researchers who are interested in conducting design science research.

6.3 Research Limitations

It is a challenge to develop an approach for generating in-depth adaptable knowledge models on the fly for web-based resources. In this dissertation, I made several assumptions which could limit the applicability of the research. The limitations of this dissertation are listed as follows.

1) The proposed approach can only create one level knowledge models. However, many websites have multi-level knowledge models.

2) The proposed approach can only handle semi-structured and structured web resources whereas many websites on the Internet are poorly-structured or have no structure at all.

3) I only used textual information on a web page in the model building process and assumed the textual content of the web page contains sufficient information for building a knowledge model. This limit the types of web-based resources can be used in the research.

6.4 Future Research

This dissertation can be extended in following directions.

1) Further refine the implementation of the research prototype. While the current prototype implemented all core functionalities and performs well, the prototype can be improved in many different directions: a labeling component (naming the clusters on the map) needs to be developed for the prototype; an annotation (user can annotates on the SOM map) function would be nice feature to have; and the prototype needs to be further tested to be bug-free.

2) Further test the proposed approach by conducting additional experiments on different types of web resources. The dataset used in this dissertation is well-structured web pages. It would improve the generalizability of the research approach if I test it on semi-structured or even less structured web-based resources.

3) This dissertation focuses on generating one-level knowledge model for web-based resources. However, many web knowledge models often have several levels. I am investigating a research approach that can dynamically create multiple level knowledge models for the web-based resources. One of the possible enabling technique is hierarchical SOM (Rauber and Merkl 1999).

4) Utilizing the deep web. The current approach only uses the first level textual information available on the web pages. The clustering performance of the system could be greatly improved by searching the relevant information on the deep web (following the links from the web pages to other web pages, articles, or databases).

5) Study the cognitive aspect of user-system interaction. In this dissertation, I test the correctness of the proposed research approach using a simulated user. The developed research prototype is proved to perform as designed. It would be very interesting to study the impact of the system on a human user. For example, will the user learn something by interacting with the system? Will the user be satisfied with the final knowledge model? Will the user find the system useful or easy to use? Those are appealing questions to be asked in future studies.

Reference

Alahahoo, D. and S. K. Halgamuge, 2000. Dynamic self-organizing maps with controlled growth for knowledge discovery. IEEE Transactions on Neural Networks 11(3), 601-614.

Ashby, F. G. and W. T. Maddox, 2004. Human Category Learning. Annual Review of Psychology 56(1), 149-178.

Ashby, W. R., 1956. An Introduction to Cybernetics. London, Chapman and Hall.

Brusilovsky, P. and M. T. Maybury, 2002. From adaptive hypermedia to the adaptive web. Communication of ACM 45(5), 30-33.

Calder, B. J., L. Phillips and A. Tybout, 1981. Designing Research for Application. Journal of Consumer Research 8, 197-207.

Chau, M., D. Zeng and H. Chen, 2001. Personalized spiders for web search and analysis. Proceedings of the 1st ACM/IEEE-CS joint conference on Digital Libraries, Roanoke, Virginia, ACM Press.

Chen, H., H. Fan, M. Chau and D. Zeng, 2001. MetaSpider: Meta-Searching and Categorization on the Web. Journal of the American Society for Information Science and Technology 52(13), 1134.

Dasgupta, S., 1996. Technology and Creativity. New York, Oxford University Press.

Ferragina, P. and A. Gulli, 2005a. A Personalized Search Engine Based on Web-snippet Hierarchical Clustering. Proceedings of 14th International World Wide Web Conference, Chiba, Japan.

Ferragina, P. and A. Gulli, 2005b. A Personalized Search Engine Based on Web-snippet Hierarchical Clustering. 14th International World Wide Web Conference, Chiba, Japan.

Foster, I. and R. L. Grossman, 2003. Data Integration in a Bandwideth-rich World. Communications of the ACM 6(11), 50-57.

Foster, I. and C. Kesselman, 1999. The Grid: Blueprint for a New Computing Infrastructure. San Francisco, CA, Morgan and Kaufmann.

Foster, I., C. Kesselman, J. M. Nick and S. Tuecke, 2002. The physiology of grid: An Open Grid Services Architecture for Distributed Systems Integration. Open Grid Service Infrastructure WG, Global Grid Forum.

GAlib. 2005. GAlib, A C++ Library of Genetic Algorithm Components 2.4.6. from http://lancet.mit.edu/ga/.

Gauch, S., J. Chafee and A. Pretschner, 2004. Ontology-based personalized search and browsing. Web Intelligence and Agent Systems 1(3-4), 219--234.

GlobusAlliance. 2005. The Globus Alliance. Retrieved Dec 14th, 2005, from http://www.globus.org.

Goldberg, D. E., 1989. Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley.

Gregor, s. and D. Jones, 2007. The Anatomy of a Design Theory. Journal of the Association for Information Systems Forthcoming.

Hevner, A., S. March, J. Park and S. Ram, 2004. Design Science in Information Systems Research. MIS Quarterly 28(1), 75-105.

Holland, J. H., 1975. Adaptation in Natural and Artificial Systems, University of Michigan Press.

Holland, J. H., 1992. Genetic Algorithms. Scientific American, 66-72.

Jain, A. K., M. N. Murty and P. J. Flynn, 1999. Data Clustering: A Review. ACM Computing Surveys 31(3), 264-323.

Kaski, S., T. Honkela, K. Lagus and T. Kohonen, 1998. WEBSOM - Self-Organizing Maps of Document Collections. Neurocomputing 21(1-3), 101- 117.

Kiang, M. Y., U. R. Kulkarni and K. Y. Tam, 1995. Self-organizing map networks as an interactive clustering tool-An application to group technology. Decision Support System 15, 351-374.

Kobsa, A., J. Koenemann and W. Pohl, 2001. Personalized Hypermedia Presentation Techniques for Improving Online Customer Relationships. The Knowledge Engineering Review 16(2), 111-155.

Kohonen, T., 1995. Self-Organizing Maps. Berlin, Springer-Verlag.

Kohonen, T., J. Hynninen, J. Kangas and J. Laaksonen. 1995. SOM_PAK: The self-organizing map package, version 3.1. Retrieved April 7, 1995.

Kohonen, T., S. Kaski, K. Lagus, J. Salojärvi, J. Honkela, A. Paatero and V. Saarela, 2000. Self Organization of a Massive Document Collections, Special Issue on Neural Networks for Data Mining and Knowledge Discovery. IEEE Transactions on Neural Networks 11(3), 574-585.

Kostiainen, T. and J. Lampinen, 2000. Maximum likelihood optimization of Self-Organizing Map parameters. Proceedings of SCI'2000, 4th World Multiconference on Systemics, Cybernetics and Informatics, Orlando, USA.

Kuechler, B., V. Vaishnavi and W. Kuechler, 2007. Design Science Research in IS: A Work in Progress. 2nd International Conference on Design Science Research in Information Systems & Technology. Pasadena, CA.

Larsen, B. and A. Aone, 1999. Fast and Effective Text Mining Using Linear-time Document Clustering. Proc. of the Fifth ACM SIGKDD Int'l Conference on Knowledge Discovery and Data Mining.

Li, L., S. Metikurke, A. Vandenberg and V. K. Vaishnavi, 2006. Generating Dynamic and Adaptive Knowledge Models for Web-Based Resources. International Conference on Design Science, Claremont, CA.

Liang, J., V. K. Vaishnavi and A. Vandenberg, 2006. Clustering of LDAP Directory Schemas to Facilitate Information Resources Interoperability Across Organizations. IEEE Transactions on System, Man, and Cybernetics, Part A 36(4), 631-642.

Mangiameli, P., S. K. Chen and D. West, 1996. A comparison of SOM Neural Network and Hierarchical Clustering Methods. European Journal of Operational Research 93(2), 402-417.

March, S. and G. Smith, 1995. Design and Natural Science Research on Information Technology. Decision Support Systems 15, 251 - 266.

Markman, A. B. and B. H. Ross, 2003. Category Use and Category Learning. Psychological Bulletin 129(4), 592-613.

Miller, G. A., 1956. The magical number of seven, plus or minus two. Some limits on our capacity for information processing. Psychological Review 63(2), 81-97.

Mondelli, M., F. Giannini, F. Reale, S. Kaski, T. Honkela, K. Lagus and T. Kohonen, 1998. WEBSOM - Self-organizing maps of document collections. Neurocomputing 21(1), 101-117.

Nürnberger, A., 2001. Clustering of document collections using a growing self-organizing map. Proceedings of BISC International Workshop on Fuzzy Logic and the Internet.

Polani, D., 1999. On the Optimization of Self-Organizing Maps by Genetic Algorithm. Proceedings of the Workshop on Self-Organizing Maps (WSOM '99), Elsevier.

Polani, D. and T. Uthmann, 1993. Training Kohonen Feature Maps in different Topologies: an Analysis using Genetic Algorithms. Proceedings of the Fifth International Conference on Genetic Algorithms, San Mateo, CA.

Polydoratou, P. and D. Nicholas, 2001. Familiarity with and Use of Metadata Formats and Metadata Registries amongst Those Working in Diverse Professional Communities within the Information Sector. Aslib Proceedings 53(8), 309-324.

Prefuse, 2007. Prefuse, An Information Integration Toolkit.

Purao, S., 2002. Design Research in the Technology of Information Systems: Truth or Dare. GSU Department of CIS Working Paper.

Rauber, A. and D. Merkl, 1999. SOMLib: A Digital Library System Based on Neural Networks. Proceedings of the 4th ACM Conference on Digital Libraries, Berkeley, CA.

Roszkewicz, R., 2004. Metadata in Context. The Seybold Report 4(8).

Roussinov, D. G. and H. Chen, 1998. A Scalable Self-Organizing Map Algorithm for Textual Classification: A Neural Network Approach to Automatic Thesaurus Generation. Communication and Cognition in Artificial Intelligence Journal 15(1-2), 81-111.

Roussinov, D. G. and H. Chen, 1999. Document clustering for electric meetings: an experimental comparison of two techniques. Decision Support Systems 27, 67-79.

Roussinov, D. G. and H. Chen, 2001. Information Navigation on the Web by Clustering and Summarizing Query Results. Information Processing and Management 37, 789-816.

Schneider-Hufschmidt, M., T. Kühme and U. Malinowski, 1993. Adaptive User Interfaces: Principles and Practice. New York, USA, Elsevier Science Inc.

Shadish, W. R., C. T.D. and C. D.T., 2002. Experimental and Quasi-Experimental Designs for Generalized Causal Inference. Boston, New York, Houghton Mifflin Company.

Smith, K. A. and A. Ng, 2003. Web Page Clustering Using a Self-organizing Map of User Navigation Patterns. Decision Support Systems 35(2), 245-256.

Stein, B. and S. M. Z. Eissen, 2002. Document Categorization with Major CLUST. 12th Annual Workshop On Information Technologies And Systems (WITS'02), Barcelona, Spain.

Vaishnavi, V. and W. Kuechler. 2004, last updated January 18, 2006. Design Research in Information Systems. from http://www.isworld.org/Researchdesign/drisISworld.htm

Van Rijsbergen, C., 1979. Information Retrieval. Butterworth, London.

Vandenberg, A., J. Liang, B. Bolet, H. Kou, V. K. Vaishnavi and D. Kuechler, 2002. Research Prototype: Semantic FacilitatorTM(SM) for LDAP Directory Services. Proceedings of the 12th Annual Workshop on Information Technologies and Systems.

Vivisimo. 2006. How the Vivísimo Clustering Engine Works. Retrieved 02/15/2006, from http://vivisimo.com/docs/howitworks.pdf.

Walls, J. G., G. R. Widmeyer and O. A. El Sawy, 1992. Building an information system design theory for vigilant EIS. Information Systems Research 3(1), 36-59.

Weber, R., 1987. Toward a Theory of Artifacts: A Paradigmatic Base for Information Systems Research. Journal of Information Systems 1(2), 3-19.

Whitley, B. E., 1996. Behavioral Science: Theory, Research, and Application. Principles of Research in Behavioral Science, 1-30.

Wu, Q., S. S. Iyengar, N. S. V. Rao, J. Barhen, V. K. Vaishnavi, H. Qi and K. Chakrabarty, 2004. On computing the route of a mobile agent for data fusion in a distributed sensor network. IEEE Transactions on Knowledge and Data Engineering 16, 740-753.

Zeng, H.-J., Q.-C. He, Z. Chen, W.-Y. Ma and J. Ma, 2004. Learning to cluster web search results Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval, Sheffield, United Kingdom, ACM Press.

Zhao, H. and S. Ram, 2004. Clustering Schema Elements for Semantic Integration of Heterogeneous Data Sources. Journal of Database Management 15(4), 88-106.

Appendix A Training and Testing Dataset

Appendix A.1 Training Dataset

Database Name	**URL**
Alzheimer’s Research Center	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29127
Array Express-European Bioinformatics Institute	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29089
NIH Neuroscience Microarray Consortium	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29088
SMD	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29091
BayGenomics	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=28833
Zebrafish Information Network	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29093
Lowes Syndrome Mutation Database	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29112
Source at Stanford University	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29094
Lafora	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29114
ImagaeJ	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=28948
Cys-Loop Ligand Gated Channels	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29100
MMRRC	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=28863
Nclamp	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=28942
NeuroMatic	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=28943
PEP	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29079
gPDB	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29108
National Brain Databank	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=28941
The Navigable Atlas of the Sheep Brain	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29008
Ear Lab	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29017
Eons	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29132
Emage	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29052
RGD	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29074
RAD	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29072
DSTool	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29123
Neural Simulation Language	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29124
MATCONT	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29125
GENESIS Simulator	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29013
Three-Dimensional Atlas of the Honeybee Brain	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29011
Honeybee Brain Atlas	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29097
Surf-Hippo	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29120

Appendix A.2 Testing Dataset

Database	**URL**
Ki-Databases	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29003
Uniprot	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29076
Duke/Southhampton Archive of Neuronal Morphology	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29104
IATR	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=28799
EEG Lab	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=28949
FlexProt	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29116
Protein Data Bank	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29077
Visiome	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=28869
Brede	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=28917
Hereditary Hearing Loss Homepage	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=28946
SynDB	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29065
HyperBrain Brain Atlas	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29024
FlyTrap	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=28897
Brain Architecture Center	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29105
Mouse Brain Atlas	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=29007
FMR Lab	http://ndg.sfn.org/eavData.aspx?db=10&cl=81&o=28950

Appendix B. Semantic Facilitator $^{TM\ SM}$ V2.0 Web Crawler Source Code

Appendix B.1 Web Crawler Source Code – ExtractHtmlLinks.java

```
/*********************************************************************
 * This is part of Semantic Facilitator (TM SM) V2.0 software package.          *
 * Copyright (c) 2007 Georgia State University, Information Integration Lab     *
 *                                                                               *
 * ExtractHtmlLinks.java                                                         *
 * -- This program read the URLs from a file, extract all the textual           *
 *    information and store web pages in cache directory                         *
 *                                                                               *
 * Command Line: java ExtractHtmlLinks [xml file contains URLs]                  *
 * Output: Web pages are cached in urlcache sub directory                        *
 * Programmer: Lei Li and Seema Metikurke                                        *
 * Email: lli@cis.gsu.edu                                                        *
 * Version: 2.0                                                                  *
 * Date: Mar 25th, 2006                                                          *
 *                                                                               *
  *********************************************************************/

import java.io.*;
import javax.xml.parsers.*;
import org.xml.sax.*;
import org.w3c.dom.*;
import java.net.*;

public class ExtractHtmlLinks
{
        static int counter = 0;
        static String FilePath = "urlscache/";
        public static String getText(Node node) {

                // We need to retrieve the text from elements, entity
                // references, CDATA sections, and text nodes; but not
                // comments or processing instructions
                int type = node.getNodeType();
                if (type == Node.COMMENT_NODE
                 || type == Node.PROCESSING_INSTRUCTION_NODE) {
                  return "";
                }

                StringBuffer text = new StringBuffer();

                String value = node.getNodeValue();
                if (value != null) text.append(value);
                if (node.hasChildNodes()) {
                  NodeList children = node.getChildNodes();
                  for (int i = 0; i < children.getLength(); i++) {
                        Node child = children.item(i);
                        text.append(getText(child));
                  }
                }

                return text.toString();

        }
// Extract the text from html file
        public static String extracthtmltags(String uri_str){
                String content = "";
                try{
                        URL str_url = new URL(uri_str);
                        // try opening the URL
                        URLConnection url_connection = str_url.openConnection();
                        BufferedReader br = new BufferedReader ( new
      InputStreamReader(url_connection.getInputStream()));
                        String line = "";

                        while ((line = br.readLine()) != null){
                                //dout.writeBytes(line);
                                content += line;
                        }//end of while
                        br.close();
                        //dout.close();
                }//end of try

                catch(IOException e){
                        System.out.println("Connection refused");
                }
                int index = 0;
                //Just collect Notes section of the page
                index = content.indexOf(">Notes</td>");
                content= content.substring(index+11);
                index = content.indexOf(">DB Category</td>");
                content = content.substring(0, index+1);
                content.replaceAll("DB Category", "  ");
                content.replaceAll("'", "");

                int string_begin_script = 0;
                int tag_begin_script = 0;
                int tag_end_script = 0;
        String without_script_text = "";
        content = content.replaceAll("&nbsp;"," ");
        content = content.replaceAll("&amp;","&");

        int string_length = content.length();

        while(string_begin_script < string_length){

          tag_begin_script = content.indexOf("<script",string_begin_script);

          if(tag_begin_script != -1){
                if(tag_begin_script > string_begin_script){
                        without_script_text +=
content.substring(string_begin_script,tag_begin_script);
                }
                tag_end_script = content.indexOf("/script>",tag_begin_script);

                if(tag_end_script == -1){
                System.out.println("Error: no closing script bracket");
                }
                else{
                        string_begin_script = tag_end_script + 8;
                }
          }else
          {
                without_script_text +=
content.substring(string_begin_script,string_length);
                string_begin_script = string_length;
          }
        }

        int string_begin_style = 0;
        int tag_begin_style = 0;
        int tag_end_style = 0;
        String without_style_text = "";
        int without_style_text_length = without_script_text.length();

        while(string_begin_style < without_style_text_length){

        tag_begin_style =
without_script_text.indexOf("<style",string_begin_style);

        if(tag_begin_style != -1){
                if(tag_begin_style > string_begin_style){
                        without_style_text +=
without_script_text.substring(string_begin_style,tag_begin_style);
```

```
                    }
                    tag_end_style =
without_script_text.indexOf("/style>",tag_begin_style);

                    if(tag_end_style == -1){
                    System.out.println("Error: no closing style bracket");
                    }
                    else{
                            string_begin_style = tag_end_style + 7;
                    }
            }
            else
            {
                    without_style_text +=
without_script_text.substring(string_begin_style,without_style_text_length);
                    string_begin_style = without_style_text_length;
            }
            }

            int without_script_style_length = without_style_text.length();
            int string_begin = 0;
            int tag_begin = 0;
            int tag_end = 0;
            String text = "";
            while(string_begin < without_script_style_length){
                    tag_begin = without_style_text.indexOf("<",string_begin);
                    if(tag_begin != -1){
                            if(tag_begin > string_begin){
                                    text +=
without_style_text.substring(string_begin+1,tag_begin);
                                    //text.replaceAll(" ","\r\n");
                                    text += " ";
                            }
                            tag_end = without_style_text.indexOf(">",tag_begin);
                            if(tag_end == -1){
                                    System.out.println("Error: no closing angular
bracket");
                            }
                            else{
                                    string_begin = tag_end;
                            }
                    }
                    else
                    {
                            text +=
without_style_text.substring(string_begin+1,without_script_style_length);
```

```
                        string_begin = without_script_style_length;
                }
        }

return text;


}
 public static void extracturllistxmldoc(Document doc) throws IOException {
        String content = "";
        try{

        int j = 0;
        NodeList urls = doc.getElementsByTagName("url");
        File outputfile = new File ("textdata.txt");
        //System.out.println("url length is -> "+urls.getLength());
        DataOutputStream dataout =  new DataOutputStream(new
BufferedOutputStream(new FileOutputStream(outputfile)));
        for (int i = 0; i < urls.getLength(); i++) {

          Element url = (Element) urls.item(i);
          String uri_str = getText(url);

          System.out.println("url processed: "+uri_str);
          //int ii = i + 1;

          try{
                File op_file;
                String op_file_name = FilePath+"urlcache"+(i+1)+".txt";

                //System.out.println("output file name is " + op_file_name);

                op_file = new File(op_file_name);
                DataOutputStream dout =  new DataOutputStream(new
BufferedOutputStream(new FileOutputStream(op_file)));
                String tempstr = extracthtmltags(uri_str);
                dout.writeBytes(tempstr);
                dataout.writeBytes(tempstr);
                dataout.writeBytes("\r\n");
                dout.close();


                }//end of try

                catch(Exception e){
                        //System.out.println("\n \n catch was :->"+e+"\n");
```

```
                    }

                }         // for ends
                 dataout.close();
                }
                catch(Exception e){
                }
    }

        public static void main(String args[])
        {

        // parsing xml begins
         try{

           // Create a handler to handle the SAX events generated during parsing
           // Create a builder
                DocumentBuilder builder =
        DocumentBuilderFactory.newInstance().newDocumentBuilder();

                // Use the builder to parse the file
                Document doc = builder.parse(new File(args[0]));
                extracturllistxmldoc(doc);

                } catch (SAXException e) {
                        // A parsing error occurred; the xml input is not valid.
                        // This exception can still be thrown, even if an error handler is
        installed.
                } catch (ParserConfigurationException e) {
                } catch (IOException e) {
                }
        }
}
```

Appendix B.2 Web Crawler Source Code – GenerateWords.java

```
/*********************************************************************
 * This is part of Semantic Facilitator (TM SM) V2.0 software package.              *
 * Copyright (c) 2007 Georgia State University, Information Integration Lab         *
 *                                                                                  *
 * GenerateWords.java                                                               *
 * -- This program read from cached URLs files, generate keywords list for          *
 *    and create all words file                                                     *
 * Command Line: java GenerateWords commonwords.txt [xml file contains URLs]        *
 * Output: keywordsofURL.txt, allwords.txt                                          *
 * Programmer: Lei Li and Seema Metikurke                                           *
 * Email: lli@cis.gsu.edu                                                           *
 * Version: 2.0                                                                     *
 * Last updated: Mar 15th, 2007                                                     *
 *                                                                                  *
  *********************************************************************/
import java.io.*;
import java.net.*;
import java.util.*;
import javax.xml.parsers.*;
import org.xml.sax.*;
import org.w3c.dom.*;

public class GenerateWords {
	String ip_name1;
	String ip_name2;
	String op_name1;
	String op_name2;

	File ip_file2;
	File op_file1;
	File op_file2;

	DataInputStream ip_stream2;
	DataOutputStream op_stream1;
	DataOutputStream op_stream2;
	static Vector commonwordsvec;

	//int total_lines;

	public GenerateWords(String i_name1,String i_name2,String o_name1, String
	o_name2){
		try{
			ip_name1 = i_name1;
			ip_name2 = i_name2;
			op_name1 = o_name1;
			op_name2 = o_name2;

			ip_file2 = new File(ip_name2); //taking input file textdata.txt
			op_file1 = new File(op_name1); //a file contain a list of keywords
for each url
			op_file2 = new File(op_name2); //generating output file
allwords.txt

			ip_stream2 = new DataInputStream(new BufferedInputStream(new
FileInputStream(ip_file2)));
			op_stream1 = new DataOutputStream(new
BufferedOutputStream(new FileOutputStream(op_file1)));
			op_stream2 = new DataOutputStream(new
BufferedOutputStream(new FileOutputStream(op_file2)));
		}
		catch(IOException e){
			System.out.println("IO Problems");
		}
}

public static void getCommonWords(String commonwordfile){
		commonwordsvec = new Vector();
		try{
			File ip_file1 = new File(commonwordfile); //taking input
commonwords.txt
			DataInputStream ip_stream1 = new DataInputStream(new
BufferedInputStream(new FileInputStream(ip_file1)));
			String nextword ="";
			nextword = ip_stream1.readLine();
			while(nextword != null){
				commonwordsvec.addElement(nextword);
				nextword = ip_stream1.readLine();

			}
			ip_stream1.close();
		}
		catch(IOException e){
			System.out.println("IO Problems");
		}
		System.out.println("Common words vector Size: " +
commonwordsvec.size());
		//for(int i=0; i<commonwordsvec.size();i++)
		//	System.out.println(commonwordsvec.elementAt(i));
}
```

```
public static Vector vectorConvertor(String ip_filename){ //generate words in
trial.txt
        Vector wordsvectorperurl = new Vector();

        StringTokenizer token_string;
        String next_line;
        String nexttokentxt;
        try{
                File ip_file = new File(ip_filename);
     DataInputStream ip_stream1 = new DataInputStream(new
BufferedInputStream(new FileInputStream(ip_file)));
                next_line = ip_stream1.readLine();
                while(next_line != null){
                        token_string = new StringTokenizer(next_line," ,\t.:;");
                                while(token_string.hasMoreTokens()){
                                nexttokentxt = token_string.nextToken();
                                nexttokentxt.replace('(', ' ');
                                nexttokentxt.replace(')', ' ');
                                nexttokentxt.trim();
                                wordsvectorperurl.addElement(nexttokentxt);
                                }//end while
                        next_line = ip_stream1.readLine();
                        }//end while
                ip_stream1.close();

        }//end try
        catch(IOException e){
                System.out.println(e.getMessage());
                System.out.println("IO Problems");
        }
        return wordsvectorperurl;

}

public static Vector takeUniquewords(Vector allwordsvec){

        Hashtable uniquetextdatawordsht = new Hashtable();

        for(int j=0; j < allwordsvec.size();j++){
                String extractedtxt = allwordsvec.elementAt(j).toString();
                if(!uniquetextdatawordsht.containsKey(extractedtxt)){
                        uniquetextdatawordsht.put(extractedtxt,extractedtxt);
                }
        }

        allwordsvec.removeAllElements();
```

```
            Enumeration uniquetextdatawordskeys = uniquetextdatawordsht.keys();
                        while(uniquetextdatawordskeys.hasMoreElements()){
                                    String uwordstr =
    (String)uniquetextdatawordskeys.nextElement();
                                    allwordsvec.addElement(uwordstr);
                        } //end while

                        return allwordsvec;

    }

    public static Vector removeCommonwords(Vector wordsvec){
            String cwstr = "";

            for(int i=0; i<commonwordsvec.size();i++){

                        Object cwobj = commonwordsvec.elementAt(i);
                        cwstr = cwstr.valueOf(cwobj);

                        for(int j=0;j<wordsvec.size();j++){
                                    String etwstr = wordsvec.elementAt(j).toString();
                                    Object etwobj = wordsvec.elementAt(j);

                           etwstr = etwstr.trim();
                                    cwstr = cwstr.trim();

                                    if(etwstr.compareToIgnoreCase(cwstr) == 0){
                                                wordsvec.removeElement(etwobj);
                                    }

                        }
            }
 return wordsvec;

    }
    public static String getText(Node node) {

                        // We need to retrieve the text from elements, entity
                        // references, CDATA sections, and text nodes; but not
                        // comments or processing instructions
                        int type = node.getNodeType();
                        if (type == Node.COMMENT_NODE
                         || type == Node.PROCESSING_INSTRUCTION_NODE) {
                           return "";
                        }
                        StringBuffer text = new StringBuffer();

                        String value = node.getNodeValue();
                        if (value != null) text.append(value);
                        if (node.hasChildNodes()) {
                          NodeList children = node.getChildNodes();
                          for (int i = 0; i < children.getLength(); i++) {
                                Node child = children.item(i);
                                text.append(getText(child));
                          }
                        }

                        return text.toString();

                }

public static void main(String[] args) {
      Vector extractedtxtwordsvec  =new Vector();
      //GenerateWords newgw = new GenerateWords(args[1].trim(),args[2].trim(),
      args[3].trim(),args[4].trim());
      getCommonWords(args[0]);
   try{
                // Create a handler to handle the SAX events generated during parsing
                // Create a builder
        DocumentBuilder builder =
      DocumentBuilderFactory.newInstance().newDocumentBuilder();

                        // Use the builder to parse the file
                Document doc = builder.parse(new File(args[1]));
                NodeList urls = doc.getElementsByTagName("url");

     File allwordsfile = new File (args[3]);
     File keywordlistfile = new File (args[2]);
                        //System.out.println("url length is -> "+urls.getLength());
     DataOutputStream allwords_out =  new DataOutputStream(new
      BufferedOutputStream(new FileOutputStream(allwordsfile)));
                DataOutputStream keywordlist_out =  new DataOutputStream(new
      BufferedOutputStream(new FileOutputStream(keywordlistfile)));
                for (int i = 0; i < urls.getLength(); i++) {

                          Element url = (Element) urls.item(i);
                          String uri_str = getText(url);
                          File ip_file;
                          String ip_filename = "urlscache/urlcache"+(i+1)+".txt";
       Vector wordsperurl = new Vector();
```

```
            wordsperurl = vectorConvertor(ip_filename);
            wordsperurl = removeCommonwords(wordsperurl);
            wordsperurl = takeUniquewords(wordsperurl);
            keywordlist_out.writeBytes(uri_str);
            for (int j=0; j< wordsperurl.size(); j++){
                                keywordlist_out.writeBytes("\t");
                                String tempstr = (String)wordsperurl.elementAt(j);
               if (tempstr.indexOf('(')>=0)tempstr=tempstr.replace('(',' ');
                                if (tempstr.indexOf(')')>=0)tempstr=tempstr.replace(')',' ');
                                tempstr.trim();
                                keywordlist_out.writeBytes(tempstr);
                                extractedtxtwordsvec.addElement(tempstr);
                              }
                              keywordlist_out.writeBytes("\r\n");
                        }
                        //extractedtxtwordsvec = removeCommonwords(extractedtxtwordsvec);
                        extractedtxtwordsvec = takeUniquewords(extractedtxtwordsvec);
           for (int j=0; j< extractedtxtwordsvec.size(); j++){

           allwords_out.writeBytes((String)extractedtxtwordsvec.elementAt(j));
                              allwords_out.writeBytes("\r\n");
                             }
                             keywordlist_out.close();
                             allwords_out.close();

                  } catch (SAXException e) {
                             // A parsing error occurred; the xml input is not valid.
                             // This exception can still be thrown, even if an error handler is
           installed.
                  } catch (ParserConfigurationException e) {
                  } catch (IOException e) {       }

      }//end main
     }
```

Appendix B.3 Web Crawler Source Code – URLtokens.java

```
/***********************************************************************
 * This is part of Semantic Facilitator (TM SM) V2.0 software package.            *
 * Copyright (c) 2007 Georgia State University, Information Integration Lab      *
 *                                                                                *
 * URLtokens.java                                                                 *
 * -- This program tokenizes the keywords list of each web pages and             *
 *    generates the data file for SOM clustering                                 *
 *                                                                                *
 * Command Line: java URLtokens keywordsOfURL.txt allwords.txt names.txt         *
 *                       kohonen_data.txt 30                                      *
 * Output: Web pages are cached in urlcache sub directory                        *
 * Programmer: Lei Li and Seema Metikurke                                        *
 * Email: lli@cis.gsu.edu                                                         *
 * Version: 2.0                                                                   *
 * Last updated:  Mar 25th, 2007                                                  *
 *                                                                                *
 ***********************************************************************/
import java.lang.String;
import java.util.*;
import javax.swing.*;
import java.io.*;
import java.awt.*;
import java.text.NumberFormat;

public class URLtokens
{

	public static void main(String[] args)
	{
		String tempstr;
		Vector keywords=new Vector();
		TextReader file1;

		//file1 is the processed input file sampletokenset.txt
		file1=new TextReader(args[0]);
		String nextrecordstr;
		int readtimes=0;

		PrintWriter allwords=null;

		try
		{
			int total_lines = new Integer(args[4].trim()).intValue();
		   //This vector holds a sequence of 530 records
			Vector[] entries=new Vector[total_lines];
```

```
            //allwords is arg[1], it has all the parsed keywords.
            allwords=new PrintWriter(new FileOutputStream(args[1]));
            int i=0;

            while(file1.ready() && i < total_lines)
            {
                        //This vector holds one record each out of 530 records
                        Vector one_record = new Vector();

                        nextrecordstr=file1.readLine();

                        StringTokenizer st = new StringTokenizer(nextrecordstr);
                        while (st.hasMoreTokens()) {
            one_record.addElement(st.nextToken());
             }

            entries[i] = one_record;

                        readtimes++;
                        i++;

            }

            int sz = entries.length;
            allwords.println("readtimes is "+readtimes);
            allwords.println("entries length is "+sz);

            for(int j = 0; j< total_lines; j++){
                        allwords.println(entries[j].toString());
            }

            //Save the features
save_keywords(keywords, entries);

            //Write all the keywords to allwords file.
            print_keywords(keywords, allwords);

            //args[2] is names.txt file and args[3] is kohonen_data.txt file

            prepare_output(keywords, entries, args[2], args[3]);
            allwords.close();

}catch(IOException e)
{
```

```
                System.out.println("IO Error :"+e);
                System.exit(1);
        }

}


//This function is used to find needed features for clustering.

public static void save_keywords(Vector keywords, Vector[] entries)
{
        int size = entries.length;

        String tempstr;

        for(int i=0; i<size; i++)
        {
                Vector one_record = new Vector();
                one_record = entries[i];
                int one_record_size = one_record.size();
                for(int j = 0; j < one_record_size; j++){
                        tempstr=one_record.elementAt(j).toString();
                        if (!keywords.contains(tempstr))
                                keywords.add(tempstr);
                }
        }

}

//This function will write all the keyword into the allwords file.
public static void print_keywords(Vector keywords, PrintWriter allwords)
{
        int size=keywords.size();
        int i;

        for(i=0; i<size; i++)
         {
                allwords.println((String)keywords.elementAt(i));
                //System.out.println("Added" + (String)keywords.elementAt(i));
        }
                System.out.println("Added" +i);

        allwords.println("\nHere are "+size+" keywords;");

}
```

```
   public static void        prepare_output(Vector keywords, Vector[] entries, String
   str1, String str2)
   {
NumberFormat nf = NumberFormat.getInstance();
            nf.setMaximumFractionDigits(3);
            try
            {

                        PrintWriter classname=new PrintWriter(new
   FileOutputStream(str1));
                        PrintWriter kohonen_out=new PrintWriter(new
   FileOutputStream(str2));

                        int entry_num=entries.length;
                        System.out.println("entry number is "+entry_num);

                        String tempstr;
                        Vector one_record = new Vector();

                        //Here the keywords are extracted from sampletokenset.txt file
                        int vector_size=keywords.size();
                        System.out.println("Vector size is "+vector_size);

                        double[] codebook=new double[vector_size];
                        kohonen_out.println(vector_size);

                        for(int i=0; i<entry_num; i++)
                        {

                                    for(int p=0; p<vector_size; p++)
                                                codebook[p]=0.0;

                                    one_record= entries[i];
                                    int one_record_size = one_record.size();
                                    for(int l=0; l<one_record_size; l++)
                                    {
                                                tempstr=one_record.elementAt(l).toString();
                                                Distance d=new Distance();
                                                for(int index=0; index<vector_size; index++)
                                                {
                                                            double
   simivalue=d.Similarity((String)keywords.elementAt(index), tempstr);
                                                            if((simivalue>codebook[index]))
                                                                        codebook[index]=simivalue;
                                                }
                                    }
```

```
                                for(int index=0; index<vector_size-1; index++)
                                {
                                        kohonen_out.print(nf.format(codebook[index]));

                                        kohonen_out.print(' ');

                                }

                                kohonen_out.print(nf.format(codebook[vector_size-1]));
                                kohonen_out.print(' ');
                                kohonen_out.println(one_record.elementAt(0));
                        }

                        kohonen_out.close();

                        //Preparing names.txt.....
                        for(int i=0; i<entry_num; i++)
                        {
          int j=i+1;
                                one_record=entries[i];
                                classname.print(j);
                                classname.print(' ');
                                classname.println(one_record.elementAt(0).toString());

                        }

                        classname.close();
                }
                catch(IOException e)
                {
                        System.out.println("IO ERROR : "+e);
                        System.exit(1);
                }


        //end of prepare_output()

}
}
```

Appendix C Semantic Facilitator $^{TM\ SM}$ V2.0 Visualization Component Source Code

Appendix C.1 Visualization Component – SOMConverter.java

```
/*********************************************************************
 * This is part of Semantic Facilitator (TM SM) V2.0 software package.         *
 * Copyright (c) 2007 Georgia State University, Information Integration Lab     *
 *                                                                              *
 * SOMConverter.java                                                            *
 * -- The program convert SOM clustering result to a format that can be visualized   *
 *    by Prefuse visualization package                                          *
 * Input file:  somoutput.txt                                                   *
 * Output file: clusteringresult.txt                                            *
 * Command Line: java SOMConverter                                              *
 *                                                                              *
 * Programmer: Lei Li                                                           *
 * Email: lli@cis.gsu.edu                                                       *
 * Version: 2.0                                                                 *
 * Date: Mar 20, 2007                                                           *
 *                                                                              *
 *********************************************************************/
import java.io.*;
import java.lang.Math;

public class SOMConverter{

        final static int screenwidth = 944;//1024-80 (80 is the size of the border)
        final static int screenheight = 608;//768-160 (160 is the size of the border)

        public static void main(String[] args) {
             int [] objxdim;
             int [] objydim;
             float [] objqunerr;// som map quantanilization error
             String [] objnames;
             int maxobjincluster=0;
             int somxdim = 1;
             int somydim =1;

             //merge two input files
              BufferedReader input =null;

              try{
                   input = new BufferedReader( new FileReader("somoutput.txt") );
                   String line = null; //not declared within while loop
                   //get number of objects in the input file
```

```
            int numberofobj = 0;
            //load xdim, ydim, and quantilization error to arrays
            while (( line = input.readLine()) != null){
                    numberofobj++;
            }
            numberofobj--;
            System.out.println("nubmer:"+ numberofobj);
            input.close();

           //get the size of the map
          input = new BufferedReader( new FileReader("somoutput.txt") );
          line = input.readLine();
          int idx = line.indexOf("hexa");
          line = line.substring(idx+5);//this is tied to the input file format
          idx = line.indexOf(" ");
          somxdim = Integer.parseInt(line.substring(0, idx));
          line = line.substring(idx+1).trim();
          idx = line.indexOf(" ");
          somydim = Integer.parseInt(line.substring(0, idx));


          //load xdim, ydim, and quantilization error to arrays
          objxdim = new int[numberofobj];
        objydim = new int[numberofobj];
        objqunerr = new float[numberofobj];

        int arrayidx = 0;
        while (( line = input.readLine()) != null){

               idx = line.indexOf(" ");
               objxdim[arrayidx]=Integer.parseInt(line.substring(0,idx));
              //System.out.println("line:"+objxdim[arrayidx]);
               line = line.substring(idx+1).trim();
               idx = line.indexOf(" ");
               objydim[arrayidx]= Integer.parseInt(line.substring(0,idx));
               line = line.substring(idx+1);
               objqunerr[arrayidx]=Float.parseFloat(line);
               //System.out.println("line:"+objqunerr[arrayidx]);
               line = line.substring(idx+1);
               arrayidx++;
                   }
                   input.close();

  // load objectnames
  objnames = new String[numberofobj];
  arrayidx = 0;
```

```
input = new BufferedReader( new FileReader("objectnames.txt") );
      while (( line = input.readLine()) != null){
              idx = line.indexOf(" ");
              objnames[arrayidx] = line.substring(idx).trim();
              //idx = line.indexOf(" ");
              //System.out.println("arrayindx: " + arrayidx);
              //objnames[arrayidx]= line.substring(0,idx);
              arrayidx++;
         }
         input.close();

//sort the data by xdim ascendingly
int tempx = 0, tempy =0;
float tempqunerr =0;
String tempname;
for (int i=0;i<numberofobj;i++){
                    for (int j=0; j<numberofobj;j++){
                             if (objxdim[j] > objxdim[i]){
                                      tempx = objxdim[j];
                                      objxdim[j] = objxdim[i];
                                      objxdim[i] = tempx;
                                            tempy = objydim[j];
                                            objydim[j] = objydim[i];
                                      objydim[i] = tempy;
                                      tempqunerr = objqunerr[j];
                                      objqunerr[j] = objqunerr[i];
                                      objqunerr[i] = tempqunerr;
                                      tempname = objnames[j];
                                      objnames[j] = objnames[i];
                                      objnames[i] = tempname;
                                   }

                          }
                 }

//sort the data by ydim ascending
int count = 0;
for (int i=0;i<numberofobj-1;i++){
                          count++;
                          if (objxdim[i] !=objxdim[i+1]){
                 for (int j=(i+1-count);j<=i;j++){
                         for (int k=(i+1-count); k<=i;k++){
                                  if (objydim[k] > objydim[j]){
                                      tempy = objydim[k];
                                      objydim[k] = objydim[j];
                                      objydim[j] = tempy;
```

```
                                                    tempqunerr = objqunerr[k];
                                                    objqunerr[k] = objqunerr[j];
                                                    objqunerr[j] = tempqunerr;
                                                    tempname = objnames[k];
                                                    objnames[k] = objnames[j];
                                                    objnames[j] = tempname;
                                       }

                                     }
                               }
        count=0;
                                 }
                                 if (i==numberofobj-2){
                         for (int j=(i+1-count);j<=i+1;j++){
                               for (int k=(i+1-count); k<=i+1;k++){
                                       if (objydim[k] > objydim[j]){
                                          tempy = objydim[k];
                                          objydim[k] = objydim[j];
                                          objydim[j] = tempy;
                                                    tempqunerr = objqunerr[k];
                                                    objqunerr[k] = objqunerr[j];
                                                    objqunerr[j] = tempqunerr;
                                                    tempname = objnames[k];
                                                    objnames[k] = objnames[j];
                                                    objnames[j] = tempname;
                                       }

                                     }
                               }
        count=0;
                                 }

                }


                //Sort the data by quantilization error ascendingly
count=0;
for (int i=0;i<numberofobj-1;i++){
                                 count++;
                                 if (objxdim[i] !=objxdim[i+1] ||
 objydim[i] !=objydim[i+1]){
                                       //count the largest number of objects in a cluster
                                       if(count>maxobjincluster) maxobjincluster = count;
                         for (int j=(i+1-count);j<=i;j++){
                               for (int k=(i+1-count); k<=i;k++){
                                       if (objqunerr[k] > objqunerr[j]){
```

```
                                                    tempqunerr = objqunerr[k];
                                                    objqunerr[k] = objqunerr[j];
                                                    objqunerr[j] = tempqunerr;
                                                    tempname = objnames[k];
                                                    objnames[k] = objnames[j];
                                                    objnames[j] = tempname;
                                        }

                                      }
                              }
        count=0;
                                }
                                if (i==numberofobj-2){
                      for (int j=(i+1-count);j<=i+1;j++){
                              for (int k=(i+1-count); k<=i+1;k++){
                                      if (objqunerr[k] > objqunerr[j]){
                                                    tempqunerr = objqunerr[k];
                                                    objqunerr[k] = objqunerr[j];
                                                    objqunerr[j] = tempqunerr;
                                        tempname = objnames[k];
                                                    objnames[k] = objnames[j];
                                                    objnames[j] = tempname;
                                        }

                                      }
                              }
        count=0;
                                }

              }


//transform the data to a format that can be displayed

//calculate how many object should be displayed in a SOM cluster
int numofobjinarow = (int)(Math.sqrt((double)maxobjincluster)+0.5);
//System.out.println("max object in cluster:"+numofobjinarow);

//calculate the size of each SOM cell
int objsize =0;
int somcellwidth = screenwidth/somxdim;
int somcellheight = screenheight/somydim;
int objwidth =(somcellwidth-4*(numofobjinarow-1))/numofobjinarow;
int objheight =(somcellheight-2*(numofobjinarow-1))/numofobjinarow;
//use the small value as the size of object
if (objwidth>objheight) objsize =objheight;
```

```
 else objsize =objwidth;
 // 30 is default size for the object
 if (objsize >30) objsize =30;
 System.out.println("SOM cell width, height, objsize,
 numofobjinarow:"+somcellwidth+" "+somcellheight+ " "+objsize+"
 "+numofobjinarow);

//calculate the position of each object
count=0;
int rowidx=0,columnidx=0;
int objprocessed =0;
for (int i=0;i<numberofobj-1;i++){

                if (objxdim[i] !=objxdim[i+1] || objydim[i] !=objydim[i+1]){
                        for (int j=i-count;j<=i;j++){
                                objxdim[j]= 40 + objxdim[j]*(somcellwidth+2) +
 columnidx*(objsize/2+2);//40 is the size of the boundary
                                objydim[j]= 80 + objydim[j]*(somcellheight+1) +
 rowidx*(objsize/2+1);//80 is the size of the boundary
                                objprocessed++;
                                //System.out.println(i+" "+j+" "+count+" "+rowidx+ "
 "+columnidx);
                                if (columnidx==(numofobjinarow-1))
 {columnidx=0;rowidx++;}
        else columnidx++;

                              }
      count=0;rowidx=0;columnidx=0;
                      }else count++;
            }

System.out.println("objprocessed:"+objprocessed);

//handle the last part of data
count=0;rowidx=0;columnidx=0;
for (int i=objprocessed;i<numberofobj; i++){
                      objxdim[i]= 40 + objxdim[i]*(somcellwidth+2) +
 columnidx*(objsize/2+2);//40 is the size of the boundary
                      objydim[i]= 80 + objydim[i]*(somcellheight+1) +
 rowidx*(objsize/2+1);//80 is the size of the boundary
                      //System.out.println(i+" "+j+" "+count+" "+rowidx+ "
 "+columnidx);
                      if (columnidx==(numofobjinarow-1)) {columnidx=0;rowidx++;}
   else columnidx++;
            }
```

```
//write the data into a new file
File outputfile = new File("clusteringresult.txt");
        DataOutputStream output = new DataOutputStream(new
BufferedOutputStream(new FileOutputStream(outputfile, false)));
output.writeBytes("xdim"+"\t"+"ydim"+"\t"+"urlname"+"\r\n");
for (int i=0; i<numberofobj; i++){
   output.writeBytes(objxdim[i]+"\t"+objydim[i]+"\t"+objnames[i]+"\r\n");
}
output.close();
  } catch (FileNotFoundException ex) {
    ex.printStackTrace();
  }
  catch (IOException ex){
   ex.printStackTrace();
  }

}
}
```

Appendix C.2 Visualization Component – SOMVisualizer.java

```
/*********************************************************************
 * This is part of Semantic Facilitator (TM SM) V2.0 software package.          *
 * Copyright (c) 2007 Georgia State University, Information Integration Lab     *
 *                                                                              *
 * SOMVisualizer.java                                                           *
 * -- The program projects the clustering result to a two dimensional map       *
 *    with interactive functions                                                *
 * Input file:  clusteringresult.txt                                            *
 * Command Line: java SOMVisualizer                                             *
 *                                                                              *
 * Programmer: Lei Li                                                           *
 * Email: lli@cis.gsu.edu                                                       *
 * Version: 2.0                                                                 *
 * Date: Mar 22th, 2007                                                         *
 * Acknowledgement: the code is adapted from Prefuse Visualization Toolkit      *
 *********************************************************************/

package prefuse.demos;

import java.awt.BorderLayout;
import java.awt.Cursor;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.awt.geom.Point2D;
import java.awt.geom.Rectangle2D;
import java.awt.Insets;

import java.awt.Font;
import java.text.NumberFormat;
import java.util.Iterator;
import java.io.*;

import javax.swing.BorderFactory;
import javax.swing.JFrame;
import javax.swing.SwingUtilities;
import javax.swing.JLabel;
import javax.swing.JButton;
import javax.swing.SwingConstants;
import javax.swing.JPanel;
import javax.swing.Box;
import javax.swing.BoxLayout;

import prefuse.Constants;
import prefuse.Display;
import prefuse.Visualization;
import prefuse.action.ActionList;
import prefuse.action.RepaintAction;
import prefuse.action.assignment.ColorAction;
import prefuse.action.layout.graph.ForceDirectedLayout;
import prefuse.activity.Activity;
import prefuse.activity.ActivityAdapter;
import prefuse.activity.ActivityListener;
import prefuse.controls.ControlAdapter;
import prefuse.controls.ToolTipControl;
import prefuse.data.Schema;
import prefuse.data.Table;
import prefuse.data.io.DelimitedTextTableReader;
import prefuse.data.io.DelimitedTextTableWriter;
import prefuse.data.query.NumberRangeModel;
import prefuse.data.tuple.TupleSet;
import prefuse.render.AxisRenderer;
import prefuse.render.DefaultRendererFactory;
import prefuse.render.LabelRenderer;
import prefuse.render.Renderer;
import prefuse.render.RendererFactory;
import prefuse.util.ColorLib;
import prefuse.util.PrefuseLib;
import prefuse.util.force.DragForce;
import prefuse.util.force.ForceItem;
import prefuse.util.force.ForceSimulator;
import prefuse.util.force.NBodyForce;
import prefuse.util.force.SpringForce;
//import prefuse.util.ui.BrowserLauncher;
import prefuse.visual.VisualItem;
//import prefuse.visual.sort.ItemSorter;
import prefuse.action.layout.AxisLabelLayout;
import prefuse.action.layout.AxisLayout;
import prefuse.visual.expression.VisiblePredicate;
//import prefuse.util.ui.UILib;
import prefuse.util.ui.*;

/**
 * @author Lei Li. Last updated: Mar 9th, 2007
 */
public class SOMVisulizer extends JPanel {
   final static int framewidth = 944;
   final static int frameheight = 608;
   private Visualization m_vis;
   private Display m_display;
   private static Table m_data;
   private static JFrame m_frame;
```

```
//private static LabelRenderer nodeRenderer;
private static boolean showlabel1 = true;
private Rectangle2D m_dataB = new Rectangle2D.Double();
private Rectangle2D m_xlabB = new Rectangle2D.Double();
private Rectangle2D m_ylabB = new Rectangle2D.Double();


public SOMVisulizer(Table t, boolean showlabel) {
        super(new BorderLayout());
    //load up the raw data
   // final Visualization
        final Visualization vis = new Visualization();
    m_vis = vis;

    vis.addTable("data", t);
    //showlabel = showlabel1;
    //use a image to represent a node
    final LabelRenderer nodeRenderer = new LabelRenderer("urlname","image");
    if (showlabel) nodeRenderer.setTextField("urlname");
    else nodeRenderer.setTextField(null);
    //nodeRenderer.setTextField(null);
    nodeRenderer.setImagePosition(Constants.TOP);
    nodeRenderer.setHorizontalPadding(1);
    nodeRenderer.setVerticalPadding(1);
    nodeRenderer.setMaxImageDimensions(20,20);
    vis.setRendererFactory(new DefaultRendererFactory(nodeRenderer));

    vis.setRendererFactory(new RendererFactory() {
        Renderer arY = new AxisRenderer(Constants.RIGHT, Constants.TOP);
        Renderer arX = new AxisRenderer(Constants.CENTER,
        Constants.FAR_BOTTOM);
        public Renderer getRenderer(VisualItem item) {
            return item.isInGroup("ylab") ? arY :
                   item.isInGroup("xlab") ? arX : nodeRenderer;
        }
    });
    vis.repaint();
    //set up the x dimension and y dimension for the map
    AxisLayout x_axis = new AxisLayout("data", "xdim", Constants.X_AXIS,
        VisiblePredicate.TRUE);
    x_axis.setRangeModel(new NumberRangeModel(0, framewidth, 0, framewidth));
    vis.putAction("x", x_axis);

    AxisLayout y_axis = new AxisLayout("data", "ydim", Constants.Y_AXIS,
        VisiblePredicate.TRUE);
```

```
y_axis.setRangeModel(new NumberRangeModel(0, frameheight, 0, frameheight));
vis.putAction("y", y_axis);

AxisLabelLayout ylabels = new AxisLabelLayout("ylab", y_axis, m_ylabB,
   frameheight/9);
vis.putAction("ylabels", ylabels);

AxisLabelLayout xlabels = new AxisLabelLayout("xlab", x_axis, m_xlabB,
   framewidth/7);
vis.putAction("xlabels", xlabels);
//System.out.println(framewidth/9+" "+frameheight/7);
ActionList draw = new ActionList();
draw.add(x_axis);
draw.add(y_axis);

draw.add(xlabels);
draw.add(ylabels);
draw.add(new RepaintAction());
vis.putAction("draw", draw);

ActionList update = new ActionList();
//update.add(new SOMVisulizerSizeAction());
update.add(new ColorAction("data", VisualItem.TEXTCOLOR) {
   public int getColor(VisualItem item) {
      return ColorLib.rgb((item.isHover() ? 255 : 0), 0, 0);
   }
});
//update.add(x_axis);
//update.add(y_axis);
//update.add(ylabels);
//update.add(xlabels);
update.add(new RepaintAction());
vis.putAction("update", update);

// this will cause docs to move out of the way when dragging
//final ForceDirectedLayout fl = new SOMVisulizerForceLayout(false);
ActivityListener fReset = new ActivityAdapter() {
   public void activityCancelled(Activity a) {
 //      fl.reset();
    }
};
ActionList forces = new ActionList(Activity.INFINITY);
//forces.add(fl);
forces.add(update);
forces.addActivityListener(fReset);
vis.putAction("forces", forces);
```

```
m_display = new Display(vis);
m_display.setBorder(BorderFactory.createEmptyBorder(15,15,15,15));
m_display.setSize(framewidth,frameheight);
m_display.setHighQuality(true);
m_display.addControlListener(new SOMVisulizerControl());
ToolTipControl ttc = new ToolTipControl(new String[]{"xdim","ydim","urlname"});
m_display.addControlListener(ttc);
// pre-load images, otherwise they will be loaded asynchronously
nodeRenderer.getImageFactory().preloadImages(vis.items(),"image");
displayLayout();
// initialize and present the interface
vis.run ("draw");
vis.runAfter("preforce", "update");
vis.run("preforce");
//m_vis = vis;

JLabel title = new JLabel("SOM Map");
title.setHorizontalAlignment(SwingConstants.CENTER);
title.setFont(new Font("Arial",Font.BOLD, 16));
title.setBorder(BorderFactory.createEmptyBorder(5,5,5,5));

JButton savebutton = new JButton("Save Map");
//savebutton.setPreferredSize(new Dimension (50, 10));
savebutton.addMouseListener(new MouseAdapter() {
    public void mouseReleased(MouseEvent e) {
      try{
          DelimitedTextTableWriter outfile= new DelimitedTextTableWriter();
           //save file in prefuse/data directory
          File datafile = new File("./data/clusteringresult.txt");

          outfile.writeTable(m_data, datafile);
           //System.out.println("I'm here"+datafile.getAbsolutePath());
      }catch(Exception ex) {
          ex.printStackTrace();
      }

   }
});
JButton showlabelbutton;
if (showlabel)  showlabelbutton= new JButton("Hide Label");
else showlabelbutton = new JButton("Show Label");
// showlabelbutton.setBounds(framewidth/2-50 + insets.left, frameheight-40+
   insets.top, size.width, size.height);
 //showlabelbutton.setPreferredSize(new Dimension (50, 10));
 showlabelbutton.addMouseListener(new MouseAdapter() {
         public void mouseReleased(MouseEvent e) {
                 m_frame.setVisible(false);
                 if (showlabel1) {m_frame = loadData(true); showlabel1 = false;}
                 else  {m_frame = loadData(false); showlabel1 = true;}

                 m_frame.setVisible(true);
                 m_frame.repaint();
        }
     });

    JButton cancelbutton = new JButton("Restore Map");
    //cancelbutton.setPreferredSize(new Dimension (50, 10));
    cancelbutton.addMouseListener(new MouseAdapter() {
         public void mouseReleased(MouseEvent e) {
                 /*m_frame.setVisible(false);
                 m_frame = loadData(false);
                 m_frame.setVisible(true);
                 m_frame.repaint();*/
                 loadData(false);
                 m_display.repaint();

        }
    });

    Box buttonbox = new Box(BoxLayout.X_AXIS);
    buttonbox.setBorder(BorderFactory.createEmptyBorder(5,5,5,5));
    buttonbox.add(Box.createHorizontalStrut(framewidth/4));
    buttonbox.add(savebutton);
    buttonbox.add(Box.createHorizontalStrut(80));
    buttonbox.add(showlabelbutton);
    buttonbox.add(Box.createHorizontalStrut(80));
    buttonbox.add(cancelbutton);

    add(title, BorderLayout.NORTH);
    add(m_display, BorderLayout.CENTER);
    add(buttonbox, BorderLayout.SOUTH);

    //add(showlabelbutton, BorderLayout.SOUTH);
    //add(cancelbutton, BorderLayout.SOUTH);

}

// ------------------------------------------------------------------------

public static void main(String[] args) {
  UILib.setPlatformLookAndFeel();
```

```
    m_frame = loadData(false);
    m_frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    m_frame.setVisible(true);
}
//load the raw data to a table
public static JFrame loadData(boolean showlabel) {
    String datafile = "/clusteringresult.txt";
    //Table data = null;
    try {
        m_data = (new DelimitedTextTableReader()).readTable(datafile);
        //String Name = "Name";
       // data.addColumn("image","CONCAT('/images/',urlname,'.jpg')");
        m_data.addColumn("image","CONCAT('/images/','www.msn.com.jpg')");
    } catch (Exception e) {
        e.printStackTrace();
        System.exit(1);
    }
    JFrame frame = new JFrame("S e m a n t i c F a c i l i t a t o r  TM SM V 2.0 |  S O
        M Map");
    frame.setContentPane(new SOMVisulizer(m_data, showlabel));
    frame.pack();
    return frame;

}

private static final String ANCHORITEM = "_anchorItem";
private static final Schema ANCHORITEM_SCHEMA = new Schema();
static {
    ANCHORITEM_SCHEMA.addColumn(ANCHORITEM, ForceItem.class);
}

public class SOMVisulizerForceLayout extends ForceDirectedLayout {

    public SOMVisulizerForceLayout(boolean enforceBounds) {
        super("data",enforceBounds,false);

        ForceSimulator fsim = new ForceSimulator();
        fsim.addForce(new NBodyForce(-0.4f, 25f, NBodyForce.DEFAULT_THETA));
        fsim.addForce(new SpringForce(1e-5f,0f));
        fsim.addForce(new DragForce());
        setForceSimulator(fsim);

        m_nodeGroup = "data";
        m_edgeGroup = null;
    }
```

```
protected float getMassValue(VisualItem n) {
   return n.isHover() ? 5f : 1f;
}

public void reset() {
   Iterator iter = m_vis.visibleItems(m_nodeGroup);
   while ( iter.hasNext() ) {
      VisualItem item = (VisualItem)iter.next();
      ForceItem aitem = (ForceItem)item.get(ANCHORITEM);
      if ( aitem != null ) {
         aitem.location[0] = (float)item.getEndX();
         aitem.location[1] = (float)item.getEndY();
      }
   }
   super.reset();
}
protected void initSimulator(ForceSimulator fsim) {
   // make sure we have force items to work with
   TupleSet t = (TupleSet)m_vis.getGroup(m_group);
   t.addColumns(ANCHORITEM_SCHEMA);
   t.addColumns(FORCEITEM_SCHEMA);

   Iterator iter = m_vis.visibleItems(m_nodeGroup);
   while ( iter.hasNext() ) {
      VisualItem item = (VisualItem)iter.next();
      // get force item
      ForceItem fitem = (ForceItem)item.get(FORCEITEM);
      if ( fitem == null ) {
         fitem = new ForceItem();
         item.set(FORCEITEM, fitem);
      }
      fitem.location[0] = (float)item.getEndX();
      fitem.location[1] = (float)item.getEndY();
      fitem.mass = getMassValue(item);

      // get spring anchor
      ForceItem aitem = (ForceItem)item.get(ANCHORITEM);
      if ( aitem == null ) {
         aitem = new ForceItem();
         item.set(ANCHORITEM, aitem);
         aitem.location[0] = fitem.location[0];
         aitem.location[1] = fitem.location[1];
      }

      fsim.addItem(fitem);
```

```
            fsim.addSpring(fitem, aitem, 0);
        }
    }
} // end of inner class SOMVisulizerForceLayout

public class SOMVisulizerControl extends ControlAdapter {
  // public static final String URL = "http://www.amazon.com/exec/obidos/tg/detail/-/";
    private VisualItem activeItem;
    private Point2D down = new Point2D.Double();
    private Point2D tmp = new Point2D.Double();
    private boolean wasFixed, dragged;
    private boolean repaint = false;

    public void itemEntered(VisualItem item, MouseEvent e) {
        Display d = (Display)e.getSource();
        d.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
        d.setToolTipText(item.getString("urlname"));
        activeItem = item;
        wasFixed = item.isFixed();
    }

    public void itemExited(VisualItem item, MouseEvent e) {
        if ( activeItem == item ) {
            activeItem = null;
            item.setFixed(wasFixed);
        }
        Display d = (Display)e.getSource();
        d.setToolTipText("urlname");
        d.setCursor(Cursor.getDefaultCursor());
    }

    public void itemPressed(VisualItem item, MouseEvent e) {
        if (!SwingUtilities.isLeftMouseButton(e)) return;

        // set the focus to the current node
        Visualization vis = item.getVisualization();
        vis.getFocusGroup(Visualization.FOCUS_ITEMS).setTuple(item);

        item.setFixed(true);
        dragged = false;
        Display d = (Display)e.getComponent();
        down = d.getAbsoluteCoordinate(e.getPoint(), down);

        vis.run("forces");
    }
    public void itemReleased(VisualItem item, MouseEvent e) {
        if (!SwingUtilities.isLeftMouseButton(e)) return;
        if ( dragged ) {
            activeItem = null;
            item.setFixed(wasFixed);
            dragged = false;
        }
        // clear the focus
        Visualization vis = item.getVisualization();
        vis.getFocusGroup(Visualization.FOCUS_ITEMS).clear();
        vis.cancel("forces");
    }

    public void itemClicked(VisualItem item, MouseEvent e) {
        if (!SwingUtilities.isLeftMouseButton(e)) return;
        if ( e.getClickCount() == 2 ) {
            String  url = item.getString("urlname");
            BrowserLauncher.showDocument(url);
        }
    }

    public void itemDragged(VisualItem item, MouseEvent e) {
        if (!SwingUtilities.isLeftMouseButton(e)) return;
        dragged = true;
        Display d = (Display)e.getComponent();
        tmp = d.getAbsoluteCoordinate(e.getPoint(), tmp);
        double dx = tmp.getX()-down.getX();
        double dy = tmp.getY()-down.getY();

        PrefuseLib.setX(item, null, item.getX()+dx);
        PrefuseLib.setY(item, null, item.getY()+dy);

        item.setStartX(item.getX()+dx);
        item.setStartY(item.getY()+dy);
        down.setLocation(tmp);
        if ( repaint )
            item.getVisualization().repaint();
    }
} // end of inner class SOMVisulizerControl

        public void displayLayout() {
            Insets i = m_display.getInsets();
            int w = m_display.getWidth();
            int h = m_display.getHeight();
            int iw = i.left+i.right;
            int ih = i.top+i.bottom;
            int aw = 0;
            int ah = 0;

            //m_dataB.setRect(i.left, i.top, w-iw-aw, h-ih-ah);
            m_xlabB.setRect(i.left, h-ah-i.bottom, w-iw-aw, ah);
            m_ylabB.setRect(i.left, i.top, w-iw, h-ih-ah);

            //m_vis.run("xlabels");
            //m_vis.run("ylabels");
            //m_vis.run("update");
        }

} // end of class SOMVisulizer
```

Appendix D Semantic Facilitator $^{TM\ SM}$ V2.0 GA & Grid Component Source Code

Appendix D.1 GA & Grid Component Source Code – gacontroller.c

```
/**********************************************************************
 * This is part of Semantic Facilitator (TM SM) V2.0 software package.              *
 * Copyright (c) 2007 Georgia State University, Information Integration Lab          *
 *                                                                                  *
 * gacontroller.c                                                                   *
 * -- This program is the main program of GA component. GA parameter values are set *
 *     in this program. The fitness function is also defined in this program         *

* Command Line: ./gacontroller                                                     *
 *                                                                                  *
 * Programmer: Lei Li                                                               *
 * Email: lli@cis.gsu.edu                                                          *
 * Version: 2.0                                                                     *
 * Date: Mar 20th, 2006                                                             *
 * Acknowledgement: the code is adapted from galib software package                 *
 *********************************************************************/
#include <stdio.h>
#include <iostream.h>
#include <fstream.h>
#include <math.h>
#include <string.h>
#include <process.h>
#include "ga.h"


float objective(GAGenome &);

int main(int argc, char **argv)
{

// See if we've been given a seed to use (for testing purposes).  When you
// specify a random seed, the evolution will be exactly the same each time
// you use that seed number.
// Question: Do we have a seed to choose?

  unsigned int seed = 0;
  for(int ii=1; ii<argc; ii++) {
    if(strcmp(argv[ii++],"seed") == 0) {
      seed = atoi(argv[ii]);
    }
  }
```

```
// Declare variables for the GA parameters and set them to some default values.
// When we use convergence as the completion measure we have to specify both
// a convergence value (larger means more converged) and a number-of-gen
// that specifies how many generations back to look to calculate the
// convergence.  The number of generations back defaults to 20, so you do not
// have to set that if you don't want to.

  int popsize  = 120;   // population size
  float pmut   = 0.14f; //permutation rate
  int ngen =20;  //generation numbers
  float pcross = 0.8f; //crossover rate
  float pconv  = 0.001f;                        // threshhold for when we have converged
  int nconv    = 10;                    // how many generations back to look

// Generate a sequence of random numbers using the values in the min and max
// arrays.  We also set one of them to integer value to show how you can get
// explicit integer representations by choosing your number of bits
// appropriately.

  GARandomSeed(seed);

  unsigned int i;
  unsigned int n=4;
  int *target=new int[n];
  float min[]={3, 3, 2, 5000};
  float max[]={10, 10, 6, 60000};

  //get a random parameter value set within the range of min and max.
  for(i=0; i<n; i++)
          target[i]=GARandomInt(min[i], max[i]);


// Create a phenotype then fill it with the phenotypes we will need to map to
// the values we read from the file.  The arguments to the add() method of a
// Bin2Dec phenotype are (1) number of bits, (2) min value, and (3) max value.
// The phenotype maps a floating-point number onto the number of bits that
// you designate.  Here we just make everything use 8 bits and use the max and
// min that were used to generate the target values.  You can experiment with
// the number of bits and max/min values in order to make the GA work better
// or worse.

  GABin2DecPhenotype map;
  for(i=0; i<n; i++)
  {
          map.add(32, min[i], max[i]);
  }

// Create the template genome using the phenotype map we just made.  The
// GA will use this genome to clone the population that it uses to do the
// evolution.

  GABin2DecGenome genome (map, objective, (void *)target);

// Now create the GA using the genome and run it.

  GASteadyStateGA ga(genome);
  ga.scoreFrequency(1);
  ga.flushFrequency(50);
  ga.scoreFilename("bog.dat");
  ga.populationSize(popsize);
  ga.pMutation(pmut);
  ga.pCrossover(pcross);
  ga.nGenerations(ngen);
  ga.pConvergence(pconv);
  ga.nConvergence(nconv);
  //ga.terminator(GAGeneticAlgorithm::TerminateUponConvergence);
  ga.evolve();


  genome = ga.statistics().bestIndividual();
  cout << "\nthe ga generated:\n";
  for(i=0; i<map.nPhenotypes(); i++)
  {
     cout.width(10);
            cout << int(genome.phenotype(i)) << " ";
  }
  cout << "\n\n";
  cout.flush();

  //begin of result output
  ofstream outfile;
  outfile.open("GA_Running_Result.txt");

  outfile<<"Best individual data is:"<<endl;

  outfile<<(int)(genome.phenotype(0))<<' '<<(int)(genome.phenotype(1))<<'
          '<<(int)(genome.phenotype(2))<<' '<<(int)(genome.phenotype(3))<<endl;
  outfile.flush();
  outfile<<genome.score()<<endl;

  outfile<<endl;
  for(i=0; i<(unsigned)ga.population().size(); i++)
  {
            outfile<<i<<endl;

                  outfile<<(int)((GABin2DecGenome&)(ga.population().individual(i))).phe
        notype(0)<<" "<<

                  (int)((GABin2DecGenome&)(ga.population().individual(i))).phenotype(1)
        <<" "<<

                  (int)((GABin2DecGenome&)(ga.population().individual(i))).phenotype(2)
        <<" "<<

                  (int)((GABin2DecGenome&)(ga.population().individual(i))).phenotype(3)
        <<endl;

                  outfile<<ga.population().individual(i).score()<<endl;

  }

  outfile.close();
  //end of result output


// Clean up by freeing the memory we allocated.

  delete [] target;

  return 0;
}

// For this objective function we try to match the values in the array of float
// that is passed to us as userData.  If the values in the genome map to
// values that are close, we return a better score.  We are limited to positive
// values for the objective value (because we're using linear scaling), so we
// take the reciprocal of the absolute value of the difference between the
// value from the phenotype and the value in the sequence.
float objective(GAGenome & c)
{
  // Extract SOM parameter values
  GABin2DecGenome & genome = (GABin2DecGenome &)c;
  int xdim=(int)genome.phenotype(0);
  int ydim=(int)genome.phenotype(1);
  int radius=(int)genome.phenotype(2);
  int length=(int)genome.phenotype(3);
  float value=0;//
  char buffer [10];
  char commandline[80]="MY_SOM_PAK ";
  // Prepare the command line for SOM
  itoa (xdim,buffer,10);
  strcat(commandline,buffer);
  strcat(commandline," ");
  itoa (ydim,buffer,10);
  strcat(commandline,buffer);
  strcat(commandline," ");
  itoa (radius,buffer,10);
  strcat(commandline,buffer);
  strcat(commandline," ");
  itoa (length,buffer,10);
  strcat(commandline,buffer);

  system(commandline);
  system("visual -din kohonen_data.txt -cin kohonen_code.txt -dout somoutput.txt");
  system("java Combine somoutput.txt urlnames.txt clusterresult.txt");
  system("java MetricsCalculator metrics.txt Cluster_Fvalue.txt");


  //get the F-value of SOM
  char buffer1[10];
  ifstream inFile("Cluster_Fvalue.txt", ios::in);
  // Check if there was an error opening the file
  if (!inFile)  cout << "Unable to open the file\n";
  else{
          while (inFile >> buffer1){
                  value =(float) atof(buffer1);
          }
  }
  inFile.close();
  return(value);
}
```

Appendix D.2 GA & Grid Component – CALCULATE_METRICS.java

```
/*********************************************************************
 * This is part of Semantic Facilitator (TM SM) V2.0 software package.          *
 * Copyright (c) 2007 Georgia State University, Information Integration Lab      *
 *                                                                               *
 * CALCULATE_METRICS.java                                                        *
 * -- This program calculates cluster recall, precision and F-measure for clustering *
 *    program                                                                    *

* Command Line: ./gacontroller                                                  *
 *                                                                               *
 * Programmer: Lei Li                                                            *
 * Email: lli@cis.gsu.edu                                                        *
 * Version: 2.0                                                                  *
 * Date: Mar 20th, 2006                                                          *
 * Acknowledgement: the code is adapted from galib software package              *
 *********************************************************************/
import java.util.Vector;
import java.lang.String;
import java.lang.Integer;
import java.io.*;
import java.util.*;
import java.text.NumberFormat;


public class CALCULATE_METRICS
{
	private static double tce, tnce, tcr, tcp;
	private static double tf;

	public static void main(String[] args)
	{
		tce=0.0; tnce=0.0; tcr=0.0; tcp=0.0;
		tf=0.0;

		Vector allObjName=new Vector();

		read_object_name(allObjName, args[0]);

		TextReader file;

		file=new TextReader(args[1]);
		int xdim, ydim;

		xdim=file.readInt();
		ydim=file.readInt();
	Computer_Cluster[] computer_clusters=new
Computer_Cluster[xdim*ydim];
	//System.out.println(computer_clusters.length);

	for(int i=0; i<computer_clusters.length; i++)
	{
		computer_clusters[i]=new Computer_Cluster();
		computer_clusters[i].setX(i%xdim);
		computer_clusters[i].setY(i/xdim);
	}

	read_computer_result(file, computer_clusters, xdim);

		TextReader file2=new TextReader(args[3]);

		Vector expert_clusters=new Vector();

		while(file2.ready())
		{
			int index=file2.readInt();
			//System.out.print("Index is ");
			//System.out.println(index);
			int size=file2.readInt();

			Expert_Cluster onecluster=new Expert_Cluster();
			onecluster.setIndex(index);

			String name;

			Vector names=new Vector();

			for(int j=0; j<size; j++)
			{
				name=file2.readWord();
			//	System.out.println(name);
				names.add(name);
			}

			onecluster.setObjs(names);
			expert_clusters.add(onecluster);
		}

		//System.out.println("size is "+expert_clusters.size());

		calculate(allObjName, computer_clusters, expert_clusters);
		try
		{
			PrintWriter out=new PrintWriter(new FileOutputStream(args[2],
	true));
			NumberFormat nf = NumberFormat.getInstance();
			nf.setMaximumFractionDigits(4);


			out.print(nf.format(tce)+" ");
			out.print(nf.format(tnce)+" ");
			out.print(nf.format(tcr)+" ");
			out.print(nf.format(tcp)+" ");
//			out.println(" = "+nf.format(tcr/6*tcp/6));
			System.out.print(nf.format(tce)+" ");
			System.out.print(nf.format(tnce)+" ");
			System.out.print(nf.format(tcr)+" ");
			System.out.print(nf.format(tcp)+" ");
			out.println(" = "+nf.format(tf));
			out.println();
	System.out.println("The score is: "+nf.format(tf));
			out.close();
			//print the F value to a file
			// file temporaly store the F-Value
	String fileForFValue = args[4];
			out=new PrintWriter(new FileOutputStream(fileForFValue, true));
			out.println(nf.format(tf));
			out.close();
		}
		catch(FileNotFoundException fnfe) { }

	}

	public static void read_object_name(Vector vec, String filename)
	{

		TextReader file;

		file=new TextReader(filename);

		String objname;
		int    index;

		while(file.ready())
```

```
        {
                index=file.readInt();
                objname=file.readWord();
                CMEntry one_entry=new CMEntry();
                one_entry.setIndex(index);
                one_entry.setName(objname);

                vec.add(one_entry);
        }

        //System.out.println("final vector size is " + vec.size());

}

public static void read_computer_result(TextReader file, Computer_Cluster[]
computer_clusters, int xdim)
{
        int x, y, index;

        while(file.ready())
        {
                x=file.readInt();
                y=file.readInt();
                index=file.readInt();

                computer_clusters[x+y*xdim].getObjs().add(new Integer(index));
                //computer_clusters[x+y*xdim].add(new Integer(index));
        }
}

public static void print_computer_result(Computer_Cluster[] computer_clusters)
{
        for(int i=0; i<computer_clusters.length; i++)
        {
                System.out.println(computer_clusters[i].getObjs().size());
        }
}

public static void print_expert_result(Vector expert_clusters)
{
        System.out.println("Here is excuted.");
        System.out.println(expert_clusters.size());
}

public static void calculate(Vector allObjName, Computer_Cluster[]
computer_clusters, Vector expert_clusters)
```

```
{
        int size=allObjName.size();
        int total_associations=size*(size-1)/2;

        int total_association=0;
        int computer_association=0;
        int manual_association=0;

        for(int i=0; i<computer_clusters.length; i++)
        {
                Computer_Cluster oneclus=computer_clusters[i];

                Vector objs=oneclus.getObjs();

                int j=objs.size();
                if (j==1) computer_association++;
                else computer_association+=j*(j-1)/2;
        }

        for(int i=0; i<expert_clusters.size(); i++)
        {
                Expert_Cluster
oneclus=(Expert_Cluster)(expert_clusters.elementAt(i));
                Vector objs=oneclus.getObjs();
                int j=objs.size();
if (j==1) manual_association++;
                manual_association+=j*(j-1)/2;
        }

        total_association=computer_association+manual_association;

        int incorrect_association=0;
        int missed_association=0;
        int error_association=0;
        for(int i=0; i<computer_clusters.length; i++)
        {
                Computer_Cluster compclus=computer_clusters[i];
if (compclus.getObjs().size()>=1){

                        try
                        {
                                //int missed=find_missed(comclus, expclus, index,
allObjName);
                                int
incorrect=find_incorrect(compclus,expert_clusters, allObjName);
                                incorrect_association+=incorrect;
```

```
                              //missed_association+=missed;
                        }
                        catch(NullPointerException npe)
                        {
                              System.out.println("a NullPointerException is
caught.");
                              //System.out.println(index + " "+ objName);
                        }
            }
        }
        /*for(int i=0; i<size; i++)
        {
                String
objName=((CMEntry)(allObjName.elementAt(i))).getName();
                int    index  =((CMEntry)(allObjName.elementAt(i))).getIndex();

                Computer_Cluster
comclus=find_computer_cluster(computer_clusters, index);
                Expert_Cluster   expclus=find_expert_cluster(expert_clusters,
objName);

                try
                {
                        int missed=find_missed(comclus, expclus, index,
allObjName);
                        int incorrect=find_incorrect(comclus, expclus, index,
allObjName);

                        incorrect_association+=incorrect;
                        missed_association+=missed;
                }
                catch(NullPointerException npe)
                {
                        System.out.println("a NullPointerException is caught.");
                        System.out.println(index + " "+ objName);
                }
        }*/

        error_association+=incorrect_association+missed_association;

        double ce=(double)error_association/(double)total_associations;
        double nce=(double)error_association/(double)total_association;
        double cr=(double)(computer_association-
incorrect_association)/(double)(manual_association);
        double cp=(double)(computer_association-
incorrect_association)/(double)computer_association;
```

```
System.out.println("Computer_asso:"+ computer_association +"
   incorrect_asso:"+incorrect_association);
System.out.println("Manual_asso:"+ manual_association +"
   computer_asso:"+computer_association);

            tce+=ce; tnce+=nce; tcr+=cr; tcp+=cp;
            if ((cp+cr)==0) tf =0;
            else tf+=2*cp*cr/(cp+cr);


   }

   private static Computer_Cluster find_computer_cluster(Computer_Cluster[]
   computer_clusters, int index)
   {
            for(int i=0; i<computer_clusters.length; i++)
            {
                     Computer_Cluster oneclus=computer_clusters[i];

                     Vector objs=oneclus.getObjs();

                     for(int j=0; j<objs.size(); j++)
                     {
                              if(((Integer)(objs.elementAt(j))).intValue()==index)
                                       return oneclus;
                     }
            }

            return null;
   }

   private static Expert_Cluster find_expert_cluster(Vector expert_clusters, String
   objName)
   {
            for(int i=0; i<expert_clusters.size(); i++)
            {
                     Expert_Cluster
   oneclus=(Expert_Cluster)(expert_clusters.elementAt(i));
                     Vector objs=oneclus.getObjs();
                     if(objs.contains(objName))
                              return oneclus;
            }

            return null;
   }
   private static int find_missed(Computer_Cluster comclus, Expert_Cluster expclus,
   int index, Vector allObjName)
   {
            Vector com=new Vector();

            Vector objs=comclus.getObjs();

            for(int j=0; j<objs.size(); j++)
            {
                     int num=((Integer)(objs.elementAt(j))).intValue();
                     if(num>index)
                              com.add(new Integer(num));
            }

            Vector manual=new Vector();

            objs=expclus.getObjs();

            for(int j=0; j<objs.size(); j++)
            {
                     String name=(String)(objs.elementAt(j));
                     int num=find_index(allObjName, name);
                     if(num>index)
                              manual.add(new Integer(num));
            }

            int num=0;

            for(int i=0; i<manual.size(); i++)
            {
                     Integer one_int=(Integer)(manual.elementAt(i));
                     if(!(com.contains(one_int)))
                              num++;
            }

            return num;
   }

   private static int find_incorrect(Computer_Cluster compclus,Vector
   expert_clusters, Vector allObjName)
   {
            int num_incorrect=0;
int idx = 0;
String objName ="";
Expert_Cluster expclus1,expclus2;
```

```
            Vector objs=compclus.getObjs();

            if (objs.size()==1){
        idx =((Integer)(objs.elementAt(0))).intValue();
        objName=((CMEntry)(allObjName.elementAt(idx-1))).getName();
        expclus1 = find_expert_cluster(expert_clusters, objName);
        if (expclus1.getObjs().size()>1) num_incorrect++;
            } else{

                for(int i=0; i<objs.size()-1; i++)
                {
                    for (int j=i+1;j<objs.size();j++){
                        idx=((Integer)(objs.elementAt(i))).intValue();
                        objName=((CMEntry)(allObjName.elementAt(idx-
        1))).getName();
                        expclus1 = find_expert_cluster(expert_clusters,
        objName);
                        idx=((Integer)(objs.elementAt(j))).intValue();
                        objName=((CMEntry)(allObjName.elementAt(idx-
        1))).getName();
                        //expclus2 = find_expert_cluster(expert_clusters,
        objName);
           if (!expclus1.getObjs().contains(objName)) num_incorrect++;
                    }
                }
          }
            return num_incorrect;
        }

        private static int find_index(Vector allObjName, String name)
        {
            int size=allObjName.size();

            for(int i=0; i<size; i++)
            {
                CMEntry one_entry=(CMEntry)(allObjName.elementAt(i));

                if((one_entry.getName()).equalsIgnoreCase(name))
                    return one_entry.getIndex();
            }

            return 0;
        }
}
```

Appendix D.3 GA & Grid Source Code – GAPopulation.c

```
/***********************************************************************
 * This is part of Semantic Facilitator (TM SM) V2.0 software package.              *
 * Copyright (c) 2007 Georgia State University, Information Integration Lab          *
 *                                                                                  *
 * GAPopulation .c                                                                  *
 * -- This program generates tasks for Grid nodes and submits the task to          *
  *    the Grid nodes                                                               *
*                                                                                   *
 * Command Line: automatically evoked by gacontroller                               *
* Programmer: Lei Li                                                                *
 * Email: lli@cis.gsu.edu                                                           *
 * Version: 2.0                                                                     *
 * Last updated:  Sep 20th, 2006                                                   *
 * Acknowledgement: the code is adapted from available galib software package      *
**********************************************************************
#include <string.h>
#include <math.h>
#include <ga/GAPopulation.h>
#include <ga/GASelector.h>
#include <ga/garandom.h>
#include <ga/GABaseGA.h>            // for the sake of flaky g++ compiler
#include <ga/ga.h>
#include <stdio.h>
#include <stdlib.h>
#include <ga/std_stream.h>
#define cout STD_COUT

static int Gridoutputfileid = 1;// variable used to identify the output file generated by Grid
          job
// routine to transform an integer to a string

const int numavailableGridnode = 13;
int availableGridnode[numavailableGridnode]={1,24,25,26,27,28,29,30,31,32,34,35,36};
//static int currentGridnodeidx = 1;
//availableGridnode[0]=1;




char* itoa(int val, int base){
        static char buf[32] = {0};
        int i = 30;
        for(; val && i ; --i, val /= base)
                buf[i] = "0123456789abcdef"[val % base];
        return &buf[i+1];
```

```
}
//routine to submit a Grid job
void submitGridJob (GAGenome & c)
{
		//variable for assigning jobID
		FILE* f_write;
		FILE* f_read;

		char jobid[8];
		char *  buffer;

	GABin2DecGenome & genome = (GABin2DecGenome &)c;
		int para1 = (int)genome.phenotype(0);
		int para2 = (int)genome.phenotype(1);
		int para3 = (int)genome.phenotype(2);

		//prepare the pbs script file
		f_write = fopen ("Gridjob.pbs","w");
		fputs("#!/bin/sh\n",f_write);


		char commandline[80]="java WebPageClustering ";
		// Prepare the command line for executable program
		buffer = itoa (para1,10);
		strcat(commandline,buffer);
		strcat(commandline," ");
		buffer = itoa (para2,10);
		strcat(commandline,buffer);
		strcat(commandline," ");
		buffer = itoa (para3,10);
		strcat(commandline,buffer);


		//compose Grid outputfile name
		buffer = itoa (Gridoutputfileid,10);
		char filename[20] = "job";//Grid output filename
		strcat(filename, buffer);
		strcat(filename, ".txt");
		//cout<<"filename:"<< filename<<"\n";
		genome.setGridFileID(Gridoutputfileid);
		//cout<<"Grid output file: "<<filename<<"\n";

		strcat (commandline, " ");
		strcat (commandline,filename);
		//strcat (commandline," >");
		//strcat (commandline,filename);
```

```
        cout<<"commandline:"<<commandline<<"\n";

        Gridoutputfileid++;


        //module for create task and submit a task Grid parameters

        char stmt[160] = "#PBS -W stageout=";
        strcat(stmt, filename);
        strcat(stmt, "@acsGridhead:/home/Griduser/GridGA/results/");
        strcat (stmt, filename);
        fputs(stmt,f_write);
        fputs("\n", f_write);
        fputs(commandline,f_write);
        //cout<<"statement ("<<i<<"):"<<stmt<<"\n";
        fclose(f_write);
        // executing the myjob.pbs script
        system("qsub Gridjob.pbs >myjobID");
        f_read = fopen("myjobID","r");

        // determine how many digit a job ID has

        if (f_read==NULL) perror ("Error opening file");
        else  fgets(jobid, 8, f_read);

        char* Gridjobid = strtok(jobid,".");
        int Gridjobid1=atoi(Gridjobid);
        genome.setGridJobID(Gridjobid1);
        genome.setGridJobSubmissionFlag(gaTrue);
        genome.evaluate();

        fclose(f_read);
}

void writetofile(char * str1, char* filename){
    FILE * f_write;
    f_write = fopen(filename,"w");
    if (!f_write) cout<<"failed\n";
    fputs(str1,f_write);
    fclose (f_write);
}

// windows is promiscuous in its use of min/max, and that causes us grief.  so
// turn of the use of min/max macros in this file.   thanks nick wienholt
#if !defined(NOMINMAX)
#define NOMINMAX
```

```
#endif

// This is the default population initializer.  It simply calls the initializer
// for each member of the population.  Then we touch the population to tell it
// that it needs to update stats and/or sort (but we don't actually force
// either one to occur.
//   The population object takes care of setting/unsetting the status flags.
void
GAPopulation::DefaultInitializer(GAPopulation & p){
            //char * str="null";
  for(int i=0; i<p.size(); i++)
    p.individual(i).initialize();
    //p.individual(i).setGridJobID(str);
}

//  The default evaluator simply calls the evaluate member of each genome in
// the population.  The population object takes care of setting/unsetting the
// status flags for indicating when the population needs to be updated again.
// This is the method that need to be modified for Grid application

/* Two useful Grid related variables and their access methods have been defined in
            GAGenome.h
   _GridjobID set/getGridJobID;
   _Gridoutputfilename set/getGridOutputFilename;


*/

void
GAPopulation::DefaultEvaluator(GAPopulation & p){
  // Create & submit task to Grid
  // int para1=0; int para2=0; int para3=0;
   //int popsize = p.size();//number of population

int i =0;
//   int jobstatus = 3;//
 for(i=0; i<p.size(); i++){
     //cout<<"generation: "<<p.ga->generation()<<"\n";

     GABin2DecGenome & genome = (GABin2DecGenome &)p.individual(i);
     //check if this genome evaluated. there are overlap between generation,

     //only non-evaluated genome will generate new task
     //cout<<"The genome "<<i<<" "<<para1 <<" " <<para2<<" "<<para3<<" is evaluated:
            " <<genome.genomeEvaluated()<<"\n";
     // 0-not evaluated, 1 evaluated.e
    // score =2 means job finished but return wrong result
    //if (genome.genomeEvaluated() == gaFalse && genome.getGridJobSubmissionFlag()
         == gaFalse)
    if (genome.genomeEvaluated() == gaFalse )

    {
       submitGridJob(genome);
      //cout<<"Task"<<i+1<<" submitted\n";
    }

  }
  // check if the submitted jobs finished
 // jobstatus - variable for job status
 // 0<jobstatus<1 - job finished with correct result
 // jobstatus = 2 - jobfinished but result isn't correct range
 // jobstatus = 3 - job unfinished, program will continue checking
// if (i==p.size()){
//          cout<<"system waiting Grid job running results\n";
//          system ("sleep 4");
// }
 // make sure all submitted job are finished
 int alljobfinished =0; // 0-No, 1- Yes
 while (1){
            int alljobfinished =0; // 0-No, 1- Yes
            for(int i=0; i<p.size(); i++){
        GABin2DecGenome & genome1 = (GABin2DecGenome &)p.individual(i);
        if          (genome1.getGridJobSubmissionFlag()          ==          gaTrue          &&
           genome1.genomeEvaluated() == gaFalse)
       // if ( genome1.genomeEvaluated() == gaFalse)
           genome1.evaluate();
        if (genome1.getScore()<1 && genome1.getScore()>=0)//job finished
            alljobfinished ++;
        if (genome1.getScore()==2) submitGridJob(genome1);
    }
    //cout<<"alljobfinished:"<<alljobfinished<<" generation size:"<<p.size();
    if (alljobfinished == p.size()) break;
 }


}



#define GA_POP_CHUNKSIZE 10  // allocate chrom ptrs in chunks of this many
```

```
/* ----------------------------------------------------------------------------
 Population

  The population class is basically just a holder for the genomes.  We also
keep track of statistics about the fitness of our genomes.  We don't care
what kind of genomes we get.  To create the population we call the clone
method of the genome we're given.
  By default we do not calculate the population's diversity, so we set the
div matrix to NULL.
---------------------------------------------------------------------------- */
GAPopulation::GAPopulation() {
  csz = N = GA_POP_CHUNKSIZE;
  n = 0;
  while(N < n) N += csz;

  rind = new GAGenome * [N];
  sind = new GAGenome * [N];
  memset(rind, 0, N * sizeof(GAGenome*));
  memset(sind, 0, N * sizeof(GAGenome*));
//  indDiv = new float[N*N];
  indDiv = 0;

  neval = 0;
  rawSum = rawAve = rawDev = rawVar = rawMax = rawMin = 0.0;
  fitSum = fitAve = fitDev = fitVar = fitMax = fitMin = 0.0;
  popDiv = -1.0;
  rsorted = ssorted = evaluated = gaFalse;
  scaled = statted = divved = selectready = gaFalse;
  sortorder = HIGH_IS_BEST;
  init = DefaultInitializer;
  eval = DefaultEvaluator;
  slct = new DEFAULT_SELECTOR;
  slct->assign(*this);
  sclscm = new DEFAULT_SCALING;
  evaldata = (GAEvalData*)0;
  ga = (GAGeneticAlgorithm*)0;
}

GAPopulation::GAPopulation(const GAGenome & c, unsigned int popsize) {
  csz = N = GA_POP_CHUNKSIZE;
  n = (popsize < 1 ? 1 : popsize);
  while(N < n) N += csz;

  rind = new GAGenome * [N];
  sind = new GAGenome * [N];
  for(unsigned int i=0; i<n; i++)
```

```
    rind[i] = c.clone(GAGenome::ATTRIBUTES);
  memcpy(sind, rind, N * sizeof(GAGenome*));
//  indDiv = new float[N*N];
  indDiv = 0;

  neval = 0;
  rawSum = rawAve = rawDev = rawVar = rawMax = rawMin = 0.0;
  fitSum = fitAve = fitDev = fitVar = fitMax = fitMin = 0.0;
  popDiv = -1.0;
  rsorted = ssorted = evaluated = gaFalse;
  scaled = statted = divved = selectready = gaFalse;
  sortorder = HIGH_IS_BEST;
  init = DefaultInitializer;
  eval = DefaultEvaluator;
  slct = new DEFAULT_SELECTOR;
  slct->assign(*this);
  sclscm = new DEFAULT_SCALING;
  evaldata = (GAEvalData*)0;
  ga = (GAGeneticAlgorithm*)0;
}

GAPopulation::GAPopulation(const GAPopulation & orig){
  n = N = 0;
  rind = sind = (GAGenome**)0;
  indDiv = (float*)0;
  sclscm = (GAScalingScheme*)0;
  slct = (GASelectionScheme*)0;
  evaldata = (GAEvalData*)0;
  copy(orig);
}

GAPopulation::~GAPopulation(){
  for(unsigned int i=0; i<n; i++)
    delete rind[i];
  delete [] rind;
  delete [] sind;
  delete [] indDiv;
  delete sclscm;
  delete slct;
  delete evaldata;
}


// Make a complete copy of the original population.  This is a deep copy of
// the population object - we clone everything in the genomes and copy all of
// the population's information.
```

```
void
GAPopulation::copy(const GAPopulation & arg)
{
  unsigned int i;
  for(i=0; i<n; i++)
    delete rind[i];
  delete [] rind;
  delete [] sind;
  delete [] indDiv;
  delete sclscm;
  delete slct;
  delete evaldata;

  csz = arg.csz; N = arg.N; n = arg.n;
  rind = new GAGenome * [N];
  for(i=0; i<n; i++)
    rind[i] = arg.rind[i]->clone();
  sind = new GAGenome * [N];
  memcpy(sind, rind, N * sizeof(GAGenome*));

  if(arg.indDiv) {
    indDiv = new float[N*N];
    memcpy(indDiv, arg.indDiv, (N*N*sizeof(float)));
  }
  else {
    indDiv = 0;
  }

  sclscm = arg.sclscm->clone();
  scaled = gaFalse;
  if(arg.scaled == gaTrue) scale();

  slct = arg.slct->clone();
  slct->assign(*this);
  selectready = gaFalse;
  if(arg.selectready == gaTrue) prepselect();

  if(arg.evaldata) evaldata = arg.evaldata->clone();
  else evaldata = (GAEvalData*)0;

  neval = 0;                              // don't copy the evaluation count!
  rawSum = arg.rawSum; rawAve = arg.rawAve;
  rawMax = arg.rawMax; rawMin = arg.rawMin;
  rawVar = arg.rawVar; rawDev = arg.rawDev;
  popDiv = arg.popDiv;
  fitSum = arg.fitSum; fitAve = arg.fitAve;
  fitMax = arg.fitMax; fitMin = arg.fitMin;
  fitVar = arg.fitVar; fitDev = arg.fitDev;

  sortorder = arg.sortorder;
  rsorted = arg.rsorted;
  ssorted = gaFalse;                // we must sort at some later point
  statted = arg.statted;
  evaluated = arg.evaluated;
  divved = arg.divved;

  init = arg.init;
  eval = arg.eval;
  ud = arg.ud;
  ga = arg.ga;
}



// Resize the population.  If we shrink, we delete the extra genomes.  If
// we grow, we clone new ones (and we DO NOT initialize them!!!).  When we
// trash the genomes, we delete the worst of the population!  We do not
// free up the space used by the array of pointers, but we do free up the
// space used by the genomes.
//   We do a clone of the genome contents so that we don't have to initialize
// the new ones (what if the population has a custom initilizer?).  We randomly
// pick that ones to clone from the existing individuals.  If the population
// contains no genomes, then we post an error message (since there are no
// individuals from that to clone the new ones).
//   If the population was evaluated, then we evaluate the new genomes.  We
// do not sort nor restat the population, and we tag the statted and sorted
// flags to reflect the fact that they are no longer valid.
//   Resizing to a bigger size is the same as a batch 'add'
int
GAPopulation::size(unsigned int popsize){
  if(popsize == n) return n;
  if(n == 0 && popsize > 0) {
    GAErr(GA_LOC, "GAPopuluation", "size", gaErrNoIndividuals);
    return n;
  }

  if(popsize > n){
    grow(popsize);
    for(unsigned int i=n; i<popsize; i++)
      rind[i] = rind[GARandomInt(0,n-1)]->clone(GAGenome::CONTENTS);
    rsorted = gaFalse;
  }
  else{
    for(unsigned int i=popsize; i<n; i++) // trash the worst ones (if sorted)
      delete rind[i];                      // may not be sorted!!!!
  }

  memcpy(sind, rind, N * sizeof(GAGenome*));
  ssorted = scaled = statted = divved = selectready = gaFalse;
  n = popsize;

  if(evaluated == gaTrue) evaluate(gaTrue);

  return n;
}


// This is a private method for adjusting the size of the arrays used by the
// population object.  Unlike the size method, this method does not allocate
// more genomes (but it will delete genomes if the specified size is smaller
// than the current size).
//   This maintains the integrity of the diversity scores (but the new ones
// will not have been set yet).
//   We return the total amount allocated (not the amount used).
int
GAPopulation::grow(unsigned int s) {
  if(s <= N) return N;

  int oldsize = N;
  while(N < s) N += csz;

  GAGenome ** tmp;

  tmp = rind;
  rind = new GAGenome * [N];
  memcpy(rind, tmp, oldsize*sizeof(GAGenome *));
  delete [] tmp;
  tmp = sind;
  sind = new GAGenome * [N];
  memcpy(sind, tmp, oldsize*sizeof(GAGenome *));
  delete [] tmp;

  if(indDiv) {
    float *tmpd = indDiv;
    indDiv = new float[N*N];
    for(int i=0; i<oldsize; i++)
      memcpy(&(indDiv[i*N]), &(tmpd[i*oldsize]), oldsize*sizeof(float));
    delete [] tmpd;
  }

  return N;
}


// Get rid of 'extra' memory that we have allocated.  We just trash the
// diversity matrix and flag it as being invalid.  Return the amount
// allocated (that is also the amount used).
int
GAPopulation::compact()
{
  if(n == N) return N;

  GAGenome ** tmp;

  tmp = rind;
  rind = new GAGenome * [n];
  memcpy(rind, tmp, n*sizeof(GAGenome *));
  delete [] tmp;
  tmp = sind;
  sind = new GAGenome * [n];
  memcpy(sind, tmp, n*sizeof(GAGenome *));
  delete [] tmp;

//  if(indDiv) {
//    float *tmpd = indDiv;
//    indDiv = new float [n*n];
//    for(unsigned int i=0; i<n; i++)
//      memcpy(&(indDiv[i*n]), &(tmpd[i*N]), n*sizeof(float));
//    delete [] tmpd;
//  }

  if(indDiv) {
    delete [] indDiv;
    indDiv = 0;
  }

  return N = n;
}


GAPopulation::SortOrder
GAPopulation::order(GAPopulation::SortOrder flag) {
  if(sortorder == flag) return flag;
  sortorder = flag;
  rsorted = ssorted = gaFalse;
  return flag;
}


// Sort using the quicksort method.  The sort order depends on whether a high
// number means 'best' or a low number means 'best'.  Individual 0 is always
// the 'best' individual, Individual n-1 is always the 'worst'.
//   We may sort either array of individuals - the array sorted by raw scores
// or the array sorted by scaled scores.
void
GAPopulation::sort(GABoolean flag, SortBasis basis) const {
  GAPopulation * This = (GAPopulation *)this;
  if(basis == RAW){
    if(rsorted == gaFalse || flag == gaTrue){
      if(sortorder == LOW_IS_BEST)
	GAPopulation::QuickSortAscendingRaw(This->rind, 0, n-1);
      else
	GAPopulation::QuickSortDescendingRaw(This->rind, 0, n-1);
      This->selectready = gaFalse;
    }
    This->rsorted = gaTrue;
  }
  else if(basis == SCALED){
    if(ssorted == gaFalse || flag == gaTrue){
      if(sortorder == LOW_IS_BEST)
	GAPopulation::QuickSortAscendingScaled(This->sind, 0, n-1);
      else
	GAPopulation::QuickSortDescendingScaled(This->sind, 0, n-1);
      This->selectready = gaFalse;
    }
    This->ssorted = gaTrue;
  }
}


// Evaluate each member of the population and store basic population statistics
// in the member variables.  It is OK to run this on a const object - it
// changes to physical state of the population, but not the logical state.
//   The partial sums are normalized to the range [0,1] so that they can be
// used whether the population is sorted as low-is-best or high-is-best.
// Individual 0 is always the best individual, and the partial sums are
// calculated so that the worst individual has the smallest partial sum.  All
// of the partial sums add to 1.0.
void
GAPopulation::statistics(GABoolean flag) const {
  if(statted == gaTrue && flag != gaTrue) return;
  GAPopulation * This = (GAPopulation *)this;

  if(n > 0) {
    float tmpsum;
    This->rawMin = This->rawMax = tmpsum = rind[0]->score();

    unsigned int i;
    for(i=1; i<n; i++){
      float scr = rind[i]->score();
      tmpsum += scr;
      This->rawMax = GAMax(rawMax, scr);
      This->rawMin = GAMin(rawMin, scr);
    }
    float tmpave = tmpsum / n;
    This->rawAve = tmpave;
    This->rawSum = tmpsum;  // if scores are huge we'll lose data here

    float tmpvar = 0.0;
    if(n > 1){
      for(i=0; i<n; i++){
	float s = rind[i]->score() - This->rawAve;
	s *= s;
	tmpvar += s;
      }
      tmpvar /= (n-1);
    }
    This->rawDev = (float)sqrt(tmpvar);
    This->rawVar = tmpvar;     // could lose data if huge variance
  }
  else {
    This->rawMin = This->rawMax = This->rawSum = 0.0;
    This->rawDev = This->rawVar = 0.0;
  }

  This->statted = gaTrue;
}


// Do the scaling on the population.  Like the statistics and diversity, this
// method does not change the contents of the population, but it does change
// the values of the status members of the object.  So we allow it to work on
// a const population.
void
GAPopulation::scale(GABoolean flag) const {
  if(scaled == gaTrue && flag != gaTrue) return;
  GAPopulation* This = (GAPopulation*)this;

  if(n > 0) {
    sclscm->evaluate(*This);

    float tmpsum;
    This->fitMin = This->fitMax = tmpsum = sind[0]->fitness();

    unsigned int i;
    for(i=1; i<n; i++){
      tmpsum += sind[i]->fitness();
      This->fitMax = GAMax(fitMax, sind[i]->fitness());
      This->fitMin = GAMin(fitMin, sind[i]->fitness());
    }
    float tmpave = tmpsum / n;
    This->fitAve = tmpave;
    This->fitSum = tmpsum;    // if scores are huge we'll lose data here

    float tmpvar = 0.0;
    if(n > 1){
      for(i=0; i<n; i++){
            float s = sind[i]->fitness() - This->fitAve;
            s *= s;
            tmpvar += s;
      }
      tmpvar /= (n-1);
    }
    This->fitDev = (float)sqrt(tmpvar);
    This->fitVar = tmpvar;       // could lose data if huge variance
  }
  else {
    This->fitMin = This->fitMax = This->fitSum = 0.0;
    This->fitVar = This->fitDev = 0.0;
  }

  This->scaled = gaTrue;
  This->ssorted = gaFalse;
}


// Calculate the population's diversity score.  The matrix is triangular and
// we don't have to calculate the diagonals.  This assumes that div(i,j) is
// the same as div(j,i) (for our purposes this will always be true, but it is
// possible for someone to override some of the individuals in the population
// and not others).
//   For now we keep twice as many diversity numbers as we need.  We need only
// n*(n-1)/2, but I can't seem to figure out an efficient way to map i,j to the
// reduced n*(n-1)/2 set (remember that the diagonals are always 0.0).
//   The diversity of the entire population is just the average of all the
// individual diversities.  So if every individual is completely different from
// all of the others, the population diversity is > 0.  If they are all the
// same, the diversity is 0.0.  We don't count the diagonals for the population
// diversity measure.  0 means minimal diversity means all the same.
void
GAPopulation::diversity(GABoolean flag) const {
  if(divved == gaTrue && flag != gaTrue) return;
  GAPopulation* This = (GAPopulation*)this;

  if(n > 1) {
    if(This->indDiv == 0) This->indDiv = new float[N*N];

    This->popDiv = 0.0;
    for(unsigned int i=0; i<n; i++){
      This->indDiv[i*n+i] = 0.0;
      for(unsigned int j=i+1; j<n; j++){
	This->indDiv[j*n+i] = This->indDiv[i*n+j] =
	  individual(i).compare(individual(j));
	This->popDiv += indDiv[i*n+j];
      }
    }
    This->popDiv /= n*(n-1)/2;
  }
  else {
    This->popDiv = 0.0;
  }

  This->divved = gaTrue;
}



void
GAPopulation::prepselect(GABoolean flag) const {
  if(selectready == gaTrue && flag != gaTrue) return;
  GAPopulation* This = (GAPopulation*)this;
  This->slct->update();
  This->selectready = gaTrue;
}



// Return a reference to the scaling object.
GAScalingScheme &
GAPopulation::scaling(const GAScalingScheme& s){
  delete sclscm;
  sclscm = s.clone();
  scaled = gaFalse;
  return *sclscm;
}


// Return a reference to the selection object.
GASelectionScheme&
GAPopulation::selector(const GASelectionScheme& s) {
  delete slct;
  slct = s.clone();
  slct->assign(*this);
  selectready = gaFalse;
  return *slct;
}



// Replace the specified genome with the one that is passed to us then
// return the one that got replaced.  Use the replacement flags to determine
// that genome will be replaced.  If we get a genome as the second
// argument, then replace that one.  If we get a NULL genome, then we
// return a NULL and don't do anything.
//   If the population is sorted, then we maintain the sort by doing a smart
// replacement.
//   If the population is not sorted, then we just do the replacement without
// worrying about the sort.  Replace best and worst both require that we know
// that chromsomes are that, so we do a sort before we do the replacement,
// then we do a smart replacement.
//   In both cases we flag the stats as out-of-date, but we do not update the
// stats.  Let that happen when it needs to happen.
//   If that is < 0 then it is a flag that tells us to do a certain kind of
// replacement.  Anything non-negative is assumed to be an index to a
// genome in the population.
//   This does not affect the state of the evaluated member - it assumes that
// the individual genome has a valid number for its score.
GAGenome *
GAPopulation::replace(GAGenome * repl, int that, SortBasis basis)
{
  int i=-1;
  GAGenome * orig=(GAGenome *)0;
  if(repl == (GAGenome *)0) return orig;

  switch(that){
  case BEST:
    sort(gaFalse, basis);
    i = 0;
    break;

  case WORST:
    sort(gaFalse, basis);
    i = n-1;
    break;

  case RANDOM:
    i = GARandomInt(0, n-1);
    break;

  default:
    if(0 <= that && that < (int)n)
      i = that;
    break;
  }

  if(i >= 0){
// We could insert this properly if the population is sorted, but that would
// require us to evaluate the genome, and we don't want to do that 'cause that
// will screw up any parallel implementations.  So we just stick it in the
// population and let the sort take care of it at a later time as needed.
    if(basis == RAW){
      orig = rind[i];                    // keep the original to return at the end
      rind[i] = repl;
      memcpy(sind, rind, N * sizeof(GAGenome*));
    }
    else{
      orig = sind[i];                    // keep the original to return at the end
      sind[i] = repl;
      memcpy(rind, sind, N * sizeof(GAGenome*));
    }
    rsorted = ssorted = gaFalse;           // must sort again
// flag for recalculate stats
    statted = gaFalse;
// Must flag for a new evaluation.
    evaluated = gaFalse;
// No way to do incremental update of scaling info since we don't know what the
// scaling object will do.
    scaled = gaFalse;
// *** should do an incremental update of the diversity here so we don't
// recalculate all of the diversities when only one is updated
    divved = gaFalse;
// selector needs update
```

```
    selectready = gaFalse;

// make sure the genome has the correct genetic algorithm pointer
    if(ga) repl->geneticAlgorithm(*ga);
  }

  return orig;
}


// Replace the genome o in the population with the genome r.  Return a
// pointer to the original genome, o.  This assumes that o exists in the
// population.   If it does not, we return a NULL.  If the genomes are the
// same, do nothing and return a pointer to the genome.
GAGenome *
GAPopulation::replace(GAGenome * r, GAGenome * o)
{
  GAGenome * orig=(GAGenome *)0;
  if(r == (GAGenome *)0 || o == (GAGenome *)0) return orig;
  if(r == o) return r;
  unsigned int i;
  for(i=0; i<n && rind[i] != o; i++);
  if(i < n) orig = replace(r, i, RAW);
  return orig;
}


//   Remove the xth genome from the population.  If index is out of bounds, we
// return NULL.  Otherwise we return a pointer to the genome that was
// removed.  The population is now no longer responsible for freeing the
// memory used by that genome.
//   We don't touch the sorted flag for the array we modify - a remove will not
// affect the sort order.
GAGenome *
GAPopulation::remove(int i, SortBasis basis)
{
  GAGenome * removed=(GAGenome *)0;
  if(i == BEST) { sort(gaFalse, basis); i = 0; }
  else if(i == WORST) { sort(gaFalse, basis); i = n-1; }
  else if(i == RANDOM) i = GARandomInt(0,n-1);
  else if(i < 0 || i >= (int)n) return removed;

  if(basis == RAW){
    removed = rind[i];
    memmove(&(rind[i]), &(rind[i+1]), (n-i-1)*sizeof(GAGenome *));
    memcpy(sind, rind, N * sizeof(GAGenome*));
    ssorted = gaFalse;
  }
  else if(basis == SCALED){
    removed = sind[i];
    memmove(&(sind[i]), &(sind[i+1]), (n-i-1)*sizeof(GAGenome *));
    memcpy(rind, sind, N * sizeof(GAGenome*));
    rsorted = gaFalse;
  }
  else return removed;

  n--;
  evaluated = gaFalse;

// *** should be smart about these and do incremental update?
  scaled = statted = divved = selectready = gaFalse;

  return removed;
}


//   Remove the specified genome from the population.  If the genome is
// not in the population, we return NULL.  We do a linear search here (yuk for
// large pops, but little else we can do).  The memory used by the genome is
// now the responsibility of the caller.
GAGenome *
GAPopulation::remove(GAGenome * r)
{
  GAGenome * removed=(GAGenome *)0;
  if(r == (GAGenome *)0) return removed;
  unsigned int i;
  for(i=0; i<n && rind[i] != r; i++);
  if(i < n) removed = remove(i, RAW);
  return removed;
}


// Add the specified individual to the population.  We don't update the stats
// or sort - let those get updated next time they are needed.
//   Notice that it is possible to add individuals to the population that are
// not the same type as the other genomes in the population.  Eventually we
// probably won't allow this (or at least we'll have to fix things so that the
// behaviour is completely defined).
//   If you invoke the add with a genome reference, the population will make
// a clone of the genome then it owns it from then on.  If you invoke add with
// a genome pointer, then the population does not allocate any memory - it uses
// the memory pointed to by the argument.  So don't trash the genome without
// first letting the population know about the change.
GAGenome*
GAPopulation::add(const GAGenome& g)
{
  return GAPopulation::add(g.clone());
}

// This one does *not* allocate space for the genome - it uses the one that
// was passed to us.  So the caller should not free it up or leave it dangling!
// We own it from now on (unless remove is called on it), and the population
// will destroy it when the population destructor is invoked.
GAGenome*
GAPopulation::add(GAGenome* c)
{
  if(c == (GAGenome *)0) return c;
  grow(n+1);
  rind[n] = sind[n] = c;
  if(ga) rind[n]->geneticAlgorithm(*ga);
  n++;

  rsorted = ssorted = gaFalse;  // may or may not be true, but must be sure
  evaluated = scaled = statted = divved = selectready = gaFalse;

  return c;
}


GAGeneticAlgorithm *
GAPopulation::geneticAlgorithm(GAGeneticAlgorithm& g){
  for(unsigned int i=0; i<n; i++)
    rind[i]->geneticAlgorithm(g);
  return(ga = &g);
}


#ifdef GALIB_USE_STREAMS
void
GAPopulation::write(STD_OSTREAM & os, SortBasis basis) const {
  for(unsigned int i=0; i<n; i++){
    if(basis == RAW)
      os << *rind[i] << "\n";
    else
      os << *sind[i] << "\n";
  }
  os << "\n";
}
```

```
#endif





void
GAPopulation::QuickSortAscendingRaw(GAGenome **c, int l, int r) {
  int i,j; float v; GAGenome *t;
  if(r > l){
    v = c[r]->score(); i = l-1; j = r;
    for(;;){
      while(c[++i]->score() < v && i <= r);
      while(c[--j]->score() > v && j > 0);
      if(i >= j) break;
      t = c[i]; c[i] = c[j]; c[j] = t;
    }
    t = c[i]; c[i] = c[r]; c[r] = t;
    GAPopulation::QuickSortAscendingRaw(c,l,i-1);
    GAPopulation::QuickSortAscendingRaw(c,i+1,r);
  }
}
void
GAPopulation::QuickSortDescendingRaw(GAGenome **c, int l, int r) {
  int i,j; float v; GAGenome *t;
  if(r > l){
    v = c[r]->score(); i = l-1; j = r;
    for(;;){
      while(c[++i]->score() > v && i <= r);
      while(c[--j]->score() < v && j > 0);
      if(i >= j) break;
      t = c[i]; c[i] = c[j]; c[j] = t;
    }
    t = c[i]; c[i] = c[r]; c[r] = t;
    GAPopulation::QuickSortDescendingRaw(c,l,i-1);
    GAPopulation::QuickSortDescendingRaw(c,i+1,r);
  }
}

void
GAPopulation::QuickSortAscendingScaled(GAGenome **c, int l, int r) {
  int i,j; float v; GAGenome *t;
  if(r > l){
    v = c[r]->fitness(); i = l-1; j = r;
    for(;;){
      while(c[++i]->fitness() < v && i <= r);
      while(c[--j]->fitness() > v && j > 0);
      if(i >= j) break;
      t = c[i]; c[i] = c[j]; c[j] = t;
    }
    t = c[i]; c[i] = c[r]; c[r] = t;
    GAPopulation::QuickSortAscendingScaled(c,l,i-1);
    GAPopulation::QuickSortAscendingScaled(c,i+1,r);
  }
}
void
GAPopulation::QuickSortDescendingScaled(GAGenome **c, int l, int r) {
  int i,j; float v; GAGenome *t;
  if(r > l){
    v = c[r]->fitness(); i = l-1; j = r;
    for(;;){
      while(c[++i]->fitness() > v && i <= r);
      while(c[--j]->fitness() < v && j > 0);
      if(i >= j) break;
      t = c[i]; c[i] = c[j]; c[j] = t;
    }
    t = c[i]; c[i] = c[r]; c[r] = t;
    GAPopulation::QuickSortDescendingScaled(c,l,i-1);
    GAPopulation::QuickSortDescendingScaled(c,i+1,r);
  }
}
```

Appendix D.4 GA & Grid Source Code – GAGenome.c

```
/***********************************************************************
 * This is part of Semantic Facilitator (TM SM) V2.0 software package.            *
 * Copyright (c) 2007 Georgia State University, Information Integration Lab        *
 *                                                                                 *
 * GAGenome.c                                                                      *
 * -- This program check if a Grid-task is completed in the assigned Grid nodes      *

* Command Line: automatically evoked by gacontroller.c                            *
 *                                                                                 *
 * Programmer: Lei Li                                                              *
 * Email: lli@cis.gsu.edu                                                          *
 * Version: 2.0                                                                    *
 * Date: Sep 26th, 2006                                                            *
 * Acknowledgement: the code is adapted from galib software package                 *
  **********************************************************************/
// $Header: /home/cvs/galib/ga/GAGenome.C,v 1.1.1.1 1999/11/11 18:56:03 mbwall Exp
        $
/* ----------------------------------------------------------------------------
  genome.C
  mbwall 19apr95
  Copyright (c) 1995 Massachusetts Institute of Technology
                 all rights reserved

 DESCRIPTION:
  Definitions for genome base class.  See the header file for complete
documentation for deriving new classes.  Comments here are implementation-
specific details about base class member functions.
---------------------------------------------------------------------------- */
#include <ga/GAGenome.h>
#include <ga/ga.h>
#include <stdlib.h>
#include<stdio.h>
#include<string.h>

//   These are the default genome operators.
// None does anything - they just post an error message to let you know that no
// method has been defined.  These are for the base class (that has no
// function by itself).
void
GAGenome::NoInitializer(GAGenome & c){
  GAErr(GA_LOC, c.className(), "initializer", gaErrOpUndef);
}
int
GAGenome::NoMutator(GAGenome & c, float){
  GAErr(GA_LOC, c.className(), "mutator", gaErrOpUndef); return 0;
}
float
GAGenome::NoComparator(const GAGenome& c, const GAGenome&){
  GAErr(GA_LOC, c.className(), "comparator", gaErrOpUndef); return -1.0;
}



GAGenome::
GAGenome(Initializer i, Mutator m, Comparator c){
  if(i==0) i=NoInitializer;
  if(m==0) m=NoMutator;
  if(c==0) c=NoComparator;
  _score=_fitness=0.0; _evaluated=gaFalse; _neval=0;
  ga=0; ud=0; eval=0; evd=0;
  init=i; mutr=m; cmp=c;
  sexcross = 0;
  asexcross = 0;
}

GAGenome::GAGenome(const GAGenome & orig){
  evd=0; _neval=0;
  GAGenome::copy(orig);
}

GAGenome::~GAGenome(){
  delete evd;
}

GAGenome*
GAGenome::clone(CloneMethod) const {
  GAErr(GA_LOC, className(), "clone", gaErrOpUndef);
  return new GAGenome(*this);
}

// The eval count is not copied from the other genome - that would inflate the
// count.
void
GAGenome::copy(const GAGenome & orig){
  if(&orig == this) return;
  _score=orig._score; _fitness=orig._fitness; _evaluated=orig._evaluated;
  ga=orig.ga; ud=orig.ud; eval=orig.eval;
  init=orig.init; mutr=orig.mutr; cmp=orig.cmp;
  sexcross=orig.sexcross; asexcross=orig.asexcross;
  _neval = 0;
  if(orig.evd){
    if(evd) evd->copy(*orig.evd);
    else evd = orig.evd->clone();
  }                                    // don't delete if c doesn't have one
}


float
GAGenome::evaluate(GABoolean flag) const {
 // cout<<"evaluating is going on.\n";


 if(_evaluated == gaFalse || flag == gaTrue){
    GAGenome *This = (GAGenome*)this;


    if(eval){ This->_neval++; This->_score = (*eval)(*This); }
    // the score should in [0, 1]
    if (This->_score<1 && This->_score>=0) {
                    This->_evaluated = gaTrue;
                    cout<<"Generation:       "<<This->ga->generation()       <<":"<<This->ga-
          >nGenerations()<<" score: "<< _score <<"\n";
          }
    else This->_evaluated =gaFalse;
    //cout<<"Generation:               "<<This->ga->generation()               <<":"<<This->ga-
          >nGenerations()<<" score: "<< _score <<"\n";

  }
  return _score;
}
```

Curriculum Vitae

BIOGRAPHICAL DETAILS

Name: Lei Li
Birthplace: Tengzhou, Shangdong, P.R. China, December 20, 1972
Address: D. Abbott Turner College of Business, Columbus State University
4225 University Avenue
Columbus, GA 31907

EDUCATION

2007	Doctor of Philosophy (Business Administration - CIS), Georgia State University
2002	Master of Science (Computer Science), Georgia State University
1995	Bachelor of Science (Mineral Processing), China University of Mining & Technology

WORK EXPERIENCE

2001-2007	Graduate Research Assistant, Georgia State University, Atlanta GA
2003-2005	Graduate Teaching Assistant, Georgia State University, Atlanta GA
1998- 2000	Application Developer, Aite Mineral Engineering & Technology, China

REFEREED PUBLICATIONS

1. Lei Li, Yi Pan, and Jie Li, “An Improved Movement-Based Location Management Scheme for PCS Network,” *Proc. of IEEE 58th Vehicular Technology Conference*, Vol. 2, pp: 757- 760, 2003.
2. Lei Li, Vijay Vaishnavi, and Art Vandenberg, "An Architecture for Semantic Facilitation and Reuse of Directory Metadata,” *Proc. of the 2004 International Conference on Information and Knowledge Engineering* , pp:332-338, 2004
3. Lei Li , R. G. Singh, G. Zheng, A. Vandenberg, V. Vaishnavi, and S. B. Navathe, “A Methodology for Semantic Integration of Metadata in Bioinformatics Data Sources, *Proc. of 43rd ACM Southeast Conference*, pp: 131-135, 2005.
4. Art Vandenberg, Vidya Rangaswamy, Seema Metikurke, Guangzhi Zheng, Lei Li , Vijay K. Vaishnavi, Sham Navathe, Christopher D. Shaw, and Diane Gromala, "The Self-Organized, Adaptive Information Integration for Web-Based Resources", *Proc of 15th Workshop on Information Technologies and Systems*, pp: 257, 2005.
5. Mark Keil, Lei Li, Lars Mathiassen, and Guangzhi Zhang, “The Influence of Checklists and Roles on Software Practitioner Risk Perception and Decision-Making,” *Proc. of 39th Hawii International Conference on System Sciences (HICSS) conference*, pp: 229.2, 2006.
6. Lei Li, Seema Metikurke, Art Vandenberg, and Vijay K. Vaishnavi, "Generating Adaptive Knowledge Models for Web-Based Resources,” *Proc. of 1st International Conference on Design Research*, 2006.

7. Lei Li, Art Vandenberg, Vijay Vaishnavi, “A Genetic Algorithm Based Approach for Systematic SOM Clustering of Directory Metadata,” *Proc. of IEEE International Conference on Granular Computing*, 2006.
8. Seema Metikurke, Vijay K. Vaishnavi, Art Vandenberg, and Lei Li, “Grid-Enabled Automatic Web Page Classification, *Proc.of IEEE World Congress on Computational Intelligence*, 2006.
9. Art Vandenberg, Vijay Vaishnavi, Lei Li and Milan Pandya, “LAIDBACK: A Workbench for User-Driven Dynamic Web Mining”, *Proc. Of 2nd International Conference on Design Science Research in Information Systems & Technology*, 2007.

CONFERENCE PRESENTATIONS

1. Lei, Li , R. G. Singh, G. Zheng, A. Vandenberg, V. Vaishnavi, and S. B. Navathe, “A Methodology for Semantic Integration of Metadata in Bioinformatics Data Sources, *presented at 43rd ACM Southeast Conference*, Atlanta, GA, 2005.
2. Lei Li, Seema Metikurke, Art Vandenberg, and Vijay K. Vaishnavi, "Generating Adaptive Knowledge Models for Web-Based Resources,” *presented at 1st International Conference on Design Research*, Claremont, CA, 2006.
3. Lei Li, Art Vandenberg, Vijay Vaishnavi, “A Genetic Algorithm Based Approach for Systematic SOM Clustering of Directory Metadata,” *presented at IEEE International Conference on Granular Computing*, Atlanta, GA, 2006.

PAPERS UNDER REVIEW

1. Mark Keil, Lei Li, Lars Mathiassen, Guangzhi Zheng, “The Influence of Checklists and Roles on Software Practitioner Risk Perception and Decision-Making,” revised and re-submitted to Journal of Software and System.
2. Lei Li , Vijay K. Vaishnavi, Art Vandenberg, “Automating SOM Clustering to Promote Interoperability of Directory Metadata,” submitted to *IEEE Transactions on Systems, Man, and Cybernetics.* Status.

SPECIAL AWARDS and HONORS

1. International Conference on Design Science Scholarship (2006)
2. Georgia State University Brain & Behavior Fellowship (2004, 2005, 2006, 2007)

SERVICE

Journal Reviews: MIS Quarterly, Information Technology & Management.

Conference Reviews: International Conference on Information Systems (ICIS), Workshop on Information Technologies & Systems (WITS), International Conference on Design Science, IEEE International Conference on Granular Computing, ACM Southeast Conference, International Conference on Information and Knowledge Engineering, Business Process Modeling Conference.