Fall 1-31-2000

Hytexpros : a hypermedia information retrieval system

Hong Shen
New Jersey Institute of Technology

Recommended Citation

ABSTRACT

HYTEXPROS: A HYPERMEDIA INFORMATION RETRIEVAL SYSTEM

by
Hong Shen

The Hypermedia information retrieval system makes use of the specific capabilities
of hypermedia systems with information retrieval operations and provides new kind
of information management tools. It combines both hypermedia and information
retrieval to offer end-users the possibility of navigating, browsing and searching a
large collection of documents to satisfy an information need. TEXPROS is an intel-
ligent document processing and retrieval system that supports storing, extracting,
classifying, categorizing, retrieval and browsing enterprise information. TEXPROS
is a perfect application to apply hypermedia information retrieval techniques. In
this dissertation, we extend TEXPROS to a hypermedia information retrieval
system called HyTEXPROS with hypertext functionalities, such as node, typed and
weighted links, anchors, guided-tours, network overview, bookmarks, annotations
and comments, and external linkbase. It describes the whole information base
including the metadata and the original documents as network nodes connected
by links. Through hypertext functionalities, a user can construct dynamically an
information path by browsing through pieces of the information base. By adding
hypertext functionalities to TEXPROS, HyTEXPROS is created. It changes its
working domain from a personal document process domain to a personal library
domain accompanied with citation techniques to process original documents. A four-
level conceptual architecture is presented as the system architecture of HyTEXPROS.
Such architecture is also referred to as the reference model of HyTEXPROS. Detailed
description of HyTEXPROS, using the First Order Logic Calculus, is also proposed.
An early version of a prototype is briefly described.

HYTEXPROS: A HYPERMEDIA INFORMATION RETRIEVAL SYSTEM

by
Hong Shen

A Dissertation
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy

Department of Computer and Information Science

January 2000

APPROVAL PAGE

HYTEXPROS: A HYPERMEDIA INFORMATION RETRIEVAL SYSTEM

Hong Shen

Dr. Peter A. Ng, Dissertation Advisor	Date
Professor of Computer Science, University of Nebraska at Omaha	

Dr. Gary Thomas, Co-Advisor, Committee Member	Date
Professor of Electrical and Computer Engineering, NJIT	

Dr. D.C. Hung, Committee Member	Date
Associate Professor of Computer and Information Science, NJIT	

Dr. Franz Kurfess, Committee Member	Date
Assistant Professor of Computer and Information Science, NJIT	

Dr. Taiming Chu, Committee Member	Date
Assistant Professor of Mechanical Engineering, NJIT	

Dr. Ronald Curtis, Committee Member	Date
Assistant Professor of Computer Science, William Paterson University	

BIOGRAPHICAL SKETCH

Author: Hong Shen

Degree: Doctor of Philosophy

Date: January 2000

Undergraduate and Graduate Education:

- Doctor of Philosophy in Computer Science,
  New Jersey Institute of Technology, USA, 2000

- Master of Science in Electrical Engineering,
  Shanghai Jiao Tong University, Shanghai, P.R. China, 1990

- Bachelor of Science in Electrical Engineering,
  Shanghai Jiao Tong University, Shanghai, P.R. China, 1987

Major: Computer Science

Publications:

Nancy H. Shen, Q. Liu and Peter A. Ng, "HyTEXPROS: A Hypermedia Information Retrieval System", *Proceeding of Intelligent Information Systems Conference*, pp.399-404, December 1997.

Nancy H. Shen, J.T. Lin, T. M. Chu, M. Tanik and Peter A. Ng, "Automatic Authoring In HyTEXPROS", To appear in *Proceeding on the Fourth World Conference on Integrated Design and Process Technology (IDPT)*, Kusadasi, Turkey, June, 1999

J.T. lin, Nancy H. Shen, T. M. Chu, R. Curtis and Peter A. Ng, "Folder Organization Query Language", To appear in *Proceeding on the Fourth World Conference on Integrated Design and Process Technology (IDPT)*, Kusadasi, Turkey, June, 1999

Nancy H. Shen and Zhongming Yin, "F-P System in Optical Fibre", *Proceeding on the Third Conference of Chinese Optical Fiber Communications*, pp.121-124, Shanghai, P.R. China, August, 1989

This work is dedicated to
my family

ACKNOWLEDGEMENT

I would like to express my deep appreciation to Dr. Peter A. Ng, my advisor, for his fruitful resources, invaluable support and constant encouragement. Without his help, it is impossible for me to pursue this Ph.D. degree. I'd like to thank Dr. Gary Thomas, my co-advisor, for his tremendous support and help, especially during the final stage of my dissertation. I'd like also thank Dr. D.C. Hung, Dr. Franz Kurfess, Dr. Taiming Chu and Dr. Ronald Curtis for actively participating in my committee.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure **Page**

CHAPTER 1

INTRODUCTION

This dissertation describes how a hypertext information retrieval(HIR) system [52] can serve as a search and navigation tool better than a normal information retrieval(IR) system [17, 84].

Both HIR and IR systems provide access to databases, primarily consisting of text documents. For both systems, the contents of documents are structured in certain forms. Both require interactions with the users in order to improve the effectiveness of retrieval. Despite of these similarities, HIR and IR are generally regarded as separate research areas, with some overlapping, but essentially different research objectives. In IR systems, representations of the content of text documents are generated by an automatic or manual indexing process. A description of the user information needs is acquired during the query formulation process. The query and the indexed documents are then compared to retrieve the targeted documents. The user evaluates these retrieved documents. A feedback process is initiated. During the feedback process, the initial query is modified and additional documents are retrieved. Development emphasize statistical techniques that can be summarized as using simple, word-based automatic text indexing [43], weighted words based on their statistical properties [77], ranking of documents according to the probability of relevance [86] , and the automatic relevance feedback for query modification [11].

HIR is a way of information management in which data are stored as nodes that are connected by links to form a network. This network provides a structure for connecting semantically or structurally related documents or fragments. Nodes can contain text, images, graphics, audio and video. The nodes can be viewed by means of an interactive browser and manipulated through the use of a structure

editor. In HIR systems, information access emphasizes multimedia data, network representations, and browsing instead of queries commonly used in IR systems.

Research areas involved in HIR are: (1) generating hypertext structure automatically from text, using techniques such as document clustering and similarity measures; (2) integrating querying and browsing, including feedback techniques during browsing to construct queries [35]; (3) retrieval models for networks; (4) providing guidance during browsing. Links are an additional source of information. It is possible to design retrieval strategies that use this information [34]

A new dual-layered paradigm for document management is embodied in TEXPROS (TEXt PROcessing System)[83, 48]. TEXPROS is a personalized and customized office information process system for processing and retrieving office documents. Basically, it has the following major features:

- Modeling the behaviors of common office activities using a state-of-the-art document model

- Classifying documents into types based on their structures. Each document type is defined in terms of attributes to form a frame template.

- Extracting most the significant information from an original document to form a frame instance of the original document, by means of the frame template . The frame instance is a synopsis of the original document.

- Filing frame instances into folders using a predicate-driven approach. That is, a frame instance is filed in a folder if it satisfies the predicate of the folder.

- Retrieving information from the folder organization. Users retrieve documents or information contained in documents on the basis of the information in their corresponding frame instances.

According to Salton's description [64], TEXPROS is an information retrieval system since it aims to provide users with information of potential interest based on the stored documents. Shneiderman [69] maintained that hypertext is appropriate in situations where "there is a large body of information organized into numerous fragments; the fragments relate to each other, and the user needs only a small fraction at any time." TEXPROS fits this pattern exceptionally well. So TEXPROS is a perfect application to combine information retrieval and hypertext to form a hypertext information retrieval system, we call that system – **HyTEXPROS**. In the following section, we will give an overview of HyTEXPROS.

1.1 HyTEXPROS Overview

A direct manipulation interface will be developed for TEXPROS that provides access to all explicit and implicit relationships among original documents and their metadata. We combine searching and navigation mechanisms in the query of hypermedia[1] information base, which can be authored automatically from TEXPROS using special authoring methodology. The links allow users to move to the node to which it leads by clicking the mouse. A schematic diagram of such a system is shown in Figure 1.1. The user can read the text of a given node, by clicking with the mouse pointer on the screen object that signifies the start of a link at the node, have another node displayed, which may then be read and used in the same way. Here it is important to recall the difference between explicit and implicit relationships. An *explicit* relationship is a link that makes available an explicit reference between two documents. An explicit relationship is built during the authoring process. It constitutes the main part of HyTEXPROS. An *implicit* relationship is a link that is implicitly present and starts from a document. An implicit relationship can be

---

[1]We don't distinguish between words *hypertext* and *hypermedia*. We always use the word *hypermedia* with the belief that our discussion applies to multimedia environments with proper extensions.

activated using a word present in a document: if a user asks to see all documents that contain a word and all documents containing the word are made available to him, then the implicit links are usually activated at run time and these links correspond to links not specifically created during the authoring processing.

HyTEXPROS has its own reference model and formal description. HyTEXPROS provides browsing or querying search strategies that allow users access to a hypertext by browsing after a query has been issued. Thus, users have access to documents that do not match the query. In particular, given a retrieved document, the user can access its neighboring documents even though they do not match the query. The system can also provide guidance to users during their browsing process. It is a further development of TEXPROS's retrieval subsystem, which is capable of processing incomplete or vague queries and providing semantically meaningful response to the users.

By incorporating hypertext functionality into TEXPROS to form HyTEXPROS in terms of objects and relationships, both developers and users have a better understanding of TEXPROS functions, user parameters and information input. The hypermedia associated functionality [13] builds on relationships to augment applications with annotation, navigation and structuring features. This gives us an opportunity to view and manage TEXPROS knowledge, by navigating among items of interest, and annotating with comments and relationships. Like a majority of hypertext system, such as SEPIA [75], KMS [4], Aquanet and VIKI [55], NoteCards [37] and Intermedia [87], HyTEXPROS is designed to facilitate authoring, creating relationships, displaying information and navigating among large information spaces.

HyTEXPROS functionality (navigation, annotation, view-oriented and structure-oriented features) is constructed using the basic building blocks of nodes, links and anchors. This helps us in identifying indirect structural relationships between individual documents and their related document, which could be inferred automat-
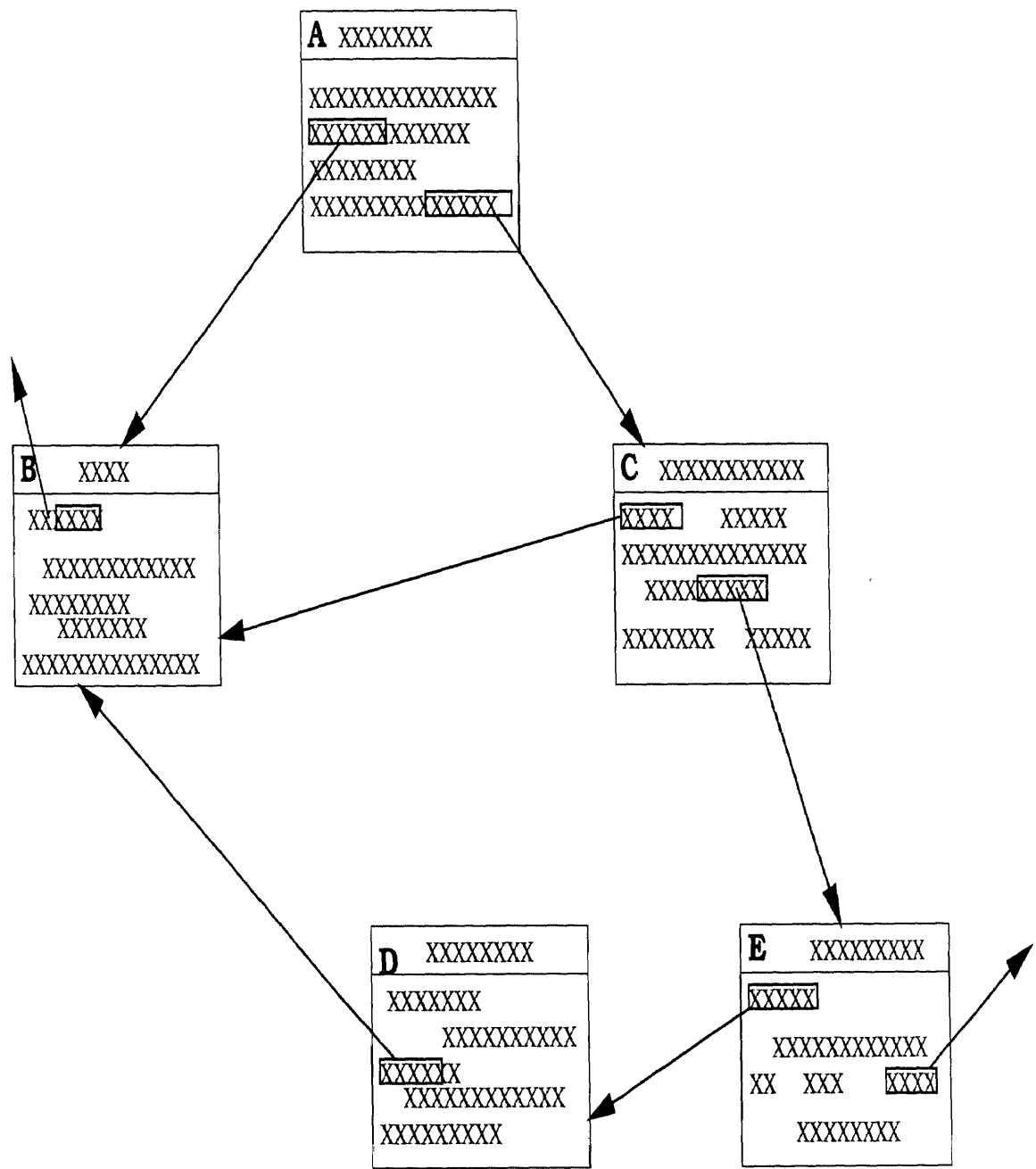
Figure 1.1 A Hypertext Network

ically by the system and thus could be accessed directly by the users. Nodes represent objects of interest to TEXPROS: original documents, frame templates, frame instances, folders, attributes, values. Links represent relationships, e.g. among folders (subfolder and superfolder), among original documents, among frame instances, among frame templates, among one original document and its corresponding frame instance, among one frame template and all the frame instances instantiated from it, among one folder and frame templates in it. Anchors define the endpoints of links. Depending on a link's focus, its anchors could map to an entire node or to specific objects within the node's content, such as one of the papers cited in a document.

Hypertext functionality includes navigational, annotation, structure-oriented and view-oriented features. So what should link to TEXPROS ?

We view hypertext as a philosophy of maximum access – give the user freedom to access and explore as much information and meta-information as possible, both to better understand an application as a whole and have more confidence in its result. Under this philosophy any item of interest to a user is a candidate for a node (though developers must take care not to overwhelm novice users). For TEXPROS this includes: (1) original documents; (2) frame templates; (3) frame instances; (4) folders; (5) attributes; (6) values.

Developers need to consider carefully the types of possible relationships that a user can access and ensure that the application structure is flexible enough to represent these. Users may need to access many associates and relationships. In HyTEXPROS, possible links includes (1) connections between one document and all the papers cited; (2) connections between folders; (3) connections between frame templates; (4) connection between frame instances; (5) connections between one document and its correspondence frame instance; (6) frame template and its instantiated frame instance; (7) one folder and frame templates inside it; (8) one folder

and frame instance inside it; and (9) connection between frame instances. Manual linking in TEXPROS is impractical, in part due to the sheer number of nodes, and in part because many relationships do not exist when the developer originally designs the system and could manually link its contents. For example, the indirect structural relationships between documents and related folders, the system could infer automatically and users could access them directly.

Navigation transports users between locations (e.g. original document, frame template, frame instance, folder) in a hypertext network. Navigation features include browsing, backtracking, and component-based query. Browsing allows users to select and traverse a link. Backtracking enables users to retrace their steps, e.g. from one frame instance to its original document. It is important to differentiate between backtracking and undoing. Backtracking returns the users to a previous location, restoring the context and preserving the state of the nodes. Component-based query can lead to a start point for browsing a large information space or can produce a starting point for a pruned subset of application data. The queries HyTEXPROS provides could be augmented with hypertext functionality, allowing users to select elements of a query specification and request meta-data or even a guided-tour [76, 54] before determining how to complete it. In addition, Users could query the hypertext meta-data base, e.g. display all comments containing the text Memo.

Annotation enables a user to add to HyTEXPROS beyond the links that already exist in the system. Comments can record personal notes and analysis, which provide information to the author (person who creates the comments) and other users, e.g. when newcomers start to read papers stored previously in TEXPROS group, comments could help them to locate the information they need and not necessarily start from scratch.

Overviews, paths and guided-tours provide alternate ways of viewing HyTEXPROS and navigating within it. Overviews graphically display HyTEXPROS nodes as

anchors, with interconnecting lines representing their interrelationships. Alternatively, overviews may take the form of a spatially-coherent display such as a map or the set-based representations. Unfortunately, overviews often cannot depict adequately all documents and structural relationships. But on the other hand, presenting all relationships would overwhelm a user. Thus, developers could consider pruning or allowing users to restrict the link types included. Variations on overviews could represent a specific task or process, or only the regions inferred to be relevant. Paths and guided-tours provide a recommended trail through HyTEXPROS. Paths permit users to leave the trail and return to it, while guided-tours restrict users to the trail's anchors, links and annotations. Trails will guide novices, functioning as a hands-on demonstration or training tool.

On the other hand, HyTEXPROS also transforms the working domain of TEXPROS from a personal document system to a personal library system [78]. Here the personal library system is s special one used by researchers who want to search required information and to use information effectively for their research. The system has a detailed knowledge of information that researchers need, which includes the results of research, papers of related works and handbooks. And on the other hand, this personal library system also include information of a specific field library such as all the literature in chemistry, computer science and others. The personal library system can help a researcher to organize information neatly in such a way that the information could be accessed efficiently. The personal library system is also valuable for information sharing between researchers with similar research interests. For example, one researcher's comments and annotation on an article could be valuable for other researchers, before they read this article they already have a rough idea about the content of the article and know which part of the article is the most interesting.

1.2 Related Work

Hypermedia information retrieval makes use of the specific capabilities of hypermedia systems together with information retrieval operations and provides a new kind of information management tools. Different aspects need to be addressed when the functionalities and capabilities of both hypertext and information retrieval systems are to be combined to offer the end-users the possibility of navigating, browsing, and searching a large collection of documents to satisfy an information need.

Aspects to be considered in designing and constructing efficient hypertext information retrieval systems include the following aspects:

- Navigation versus direct search;

- The possibility for modifying the status of a hypertext from passive to active;

- Automatic construction of hypertext;

- User modeling and interfaces.

In the following subsections, these aspects with their related works will be addressed individually.

1.2.1 Navigation versus Direct Search

The information retrieval modalities carried out by a hypertext system are different from those of a traditional information retrieval system. Search is conducted by navigation through the information base, not by direct search using a search language. The navigation allows the user to dynamically construct an information path by browsing through pieces of the information base. Most of the work in HIR has been devoted to the presentation of new retrieval models to combine both navigation and direct search features. Some of these new retrieval models incorporate the facility of managing the representation of objects at different levels of abstraction, whereby "object" means any element of the HIR; examples of managed

objects would be nodes and links. This facility provides for the creation of nodes and links which form structures allowing different levels of search depth. An example of such a structure is the *composite node* which is derived from the application of the aggregation mechanism to a collection of related nodes and gives the possibility of managing and using a collection as a single node.

Dunlop in [21] and [22] presented a combined model of IR which encompasses the principles and benefits of both free text retrieval and hypermedia. It is a hybrid approach combining the capabilities of browsing and querying to retrieve information from a large textual and multimedia collection of documents. Contextual information is used for the retrieval of non-textual documents. This model gives the users access to large document bases with limited structure, which can be browsed whatever its topology. The model approximates the contents of documents that can not be directly retrieved by contents (e.g. images), and in fact it makes use of contextual information extracted for this purpose from a hypermedia network. Moreover, this model has been only partially evaluated because of the lack of methods to evaluate HIR systems. However, experiments and observations of a prototype system have shown that the use of context information from hypermedia networks to retrieve non-textual nodes by querying is effective.

In [2, 3] an architecture and a new functional model have been introduced to overcome major limitations of hypertext systems in relation to IR operations. The model has been named EXPLICIT, because its main focus is on the *explicit* presentation to the user of the network of index terms and concepts that are used for the representation of the document collection. EXPLICIT incorporates some important IR functions and assists the end-user by means of a new type of associative IR. The two most important features of this model are a semantic association and an associative reading function. The purpose of the semantic association function is to make transparent and to communicate the meaning the system assigns to the

concepts with which the user expresses his or her information needs. Associative reading, in contrast rescues the disorientation of the user by providing a guide to browse the structure of concepts and the hypertext of documents. The possibility of using different techniques for semantic representation of the information being administered are exploited by EXPLICIT, and in the prototypes developed so far the opportunity to choose between different indexing techniques is given to the user and this, in turn, means the availability of different semantic interfacing capabilities for those users having different knowledge levels of the specific field covered by the managed document collection.

Croft and *Turtle* in [18] found that the relationship between information contained in hypertext links improves the retrieval effectiveness. The results are used to develop a new Hypertext Information Retrieval model which also has the capability of enabling the automatic construction of links.

Bruza and *van der Weide* generalized a two-level approach for hypertext information retrieval systems into hypermedia structures in [9]. This is a general framework in which a number of approaches, such as state-of-the-art hypermedia, documents and keyword-based systems, can be considered. Furthermore, the stratified hypermedia architecture based on these structures constitutes an integration between logic-based information retrieval and the two-level hypertext approaches. This integration is realized by considering the retrieval process as navigation between layers. The hyperindex structure that is derived by applying this approach facilitates the process of query formulation.

1.2.2 Status of the Hypertext

Hypertext information retrieval applications have the possibility to modify the status of a hypertext from passive to active. This may be accomplished for example, by attaching a sort of "level of importance" to a link in order to provide a different link

relevance depending on the user's path. In this way, the user could be advised to follow one path instead of another, depending on the previous path taken from a node or from an anchor. This possibility could be implemented by establishing different types of links or using weights on links at time of construction of the hypertext. An example of a model that supports weights associated with links to express the strength of the relationship can be found in [49, 51].

Another way to provide active hypertext is through *relevance feedback* as reported in [3]. Although it is often thought that only hypertext can provide browsing capabilities, it must be noted that the ability to move between related documents can also be provided by information retrieval systems supporting relevance feedback [79]. Unlike hypertext, which generally has fixed links, relevance feedback allows the user to dynamically create links at run time by searching for documents similar to some others marked as relevant. However, browsing by means of relevance feedback is a very complex process and most existing IR systems supporting relevance feedback do not have satisfactory user interfaces for browsing, as it has been pointed out by Aalbersberg in [1].

1.2.3   Automatic Construction of Hypertext

The collection of documents made available through a HIR tool is usually a "flat" collection of documents. To transform a flat collection into a hypertext, it is necessary to author the hypertext. This means producing fragments of documents from the original complete documents and building up links among them. At present, it is a common practice to author hypertexts manually. If the initial collection of documents is very large and also consists of multimedia documents, a completely manual authoring can be difficult to achieve. It is therefore important to have automatic techniques to segment documents, tools for automatic generation of links,

and procedures for automatic authoring of the hypertext to insert, modify, and cancel part of it over time.

Automatic authoring has been addressed by researchers since the earliest days of hypertext. One of the earliest works in the field of the automatic transformation of text into hypertext is reported in [27]. This work illustrated the methodology and the implementation of a technique for converting a regularly and consistently structured document into a hypertext. Regularly and consistently structured documents are those having a well-identified and fixed structure, for example, bibliographic references. The resulting hypertext is made of nodes corresponding to the document parts connected by means of structural links. The methodology used is based on the reasonable assumption that there is a close relationship between the physical components of a document and the hypertext nodes. This methodology is well suited for medium-grained documents that are regularly and consistently structured, such as, the collection of dissertation abstracts they used for their experiments.

Rana[59] addressed the combination of structural links and content links. The author distinguishes first-order and second-order hypertext. The former uses only structural links based upon the document markup and determined by the document author. These structural links include links connecting outline headings, citations, cross-references, and indices. In the second-order hypertext, links are not explicitly put into the text by the author but are detected using some automatic procedure. These second-order links are set up between index terms using co-occurrence data. The use of first and second order links in the same hypertext allows both the structural schema of the source documents, which is the document author's schema, and a second alternative schema reflecting the way index terms are distributed across the documents, to be combined. Alternative outlines are different views of the same documents that users can employ to improve their understanding during browsing since alternative outlines offer different semantic points of view of the same document.

As the author suggests, some more work should be carried out to test which type of hypertext, first or second order hypertext, the user appreciate most.

Salton and *Buckley*[62] did not directly tackle the problem of automatic construction of a hypertext, but they proposed a technique for creating links between text segments that practically build up a hypertext at retrieval time. The proposed technique is the first attempt to use vector similarity to produce a network of text segments that are semantically related. The normalized *if.idf* weighting schema is used in the context of the vector space model to evaluate the similarity between two text segments. The major advantage of this approach is that it can be performed at retrieval time, its disadvantage is that it could take a long time to produce a large number of links.

The approach to automatic hypertext construction proposed in [72] is based on the computation of node-node similarity. This approach is different from other approaches based on IR techniques because it uses the overall hypertext topology as a decision support to link settings. A measure of the hypertext topology is used to assess the degree of hypertext compactness. The lower the number of jumps from one node to another that the user has to follow to access desired information, the more compact the hypertext. This measure of hypertext compactness is employed to decide whether a similarity-based link should be added. Proposing guidelines to control the automatic construction of the hypertext is one of the major contribution of this work. What remains to be discussed is the relevance of the hypertext topology to the hypertext effectiveness. Some highly compact hypertext may result in user disorientation because of too many links. Moreover, a compact hypertext is not always desirable. A hypertext with a large number of links helps the user; such as information about the link type.

The problem of discovering link types has been addressed by *Allan* in [5]. The proposed technique provides a way of setting up links between passages of documents.

The novelty of this work is that classical IR techniques are employed to determine the type of relationship occurring in a hypertext whose nodes are topics. In addition, the problem of the number of links is also addressed. To reduce the number of links and to make the visualization of the resulting graph easier, techniques for merging links are suggested and described. The techniques for automatic link type identification are based on values calculated from merged links. For example, to identify a summary link, we can compute the amount of unlinked text that was added to a link end-point during link merging.

1.2.4   User Modeling and Interfaces

The use of hypertext applications has shown over time the difficulty the user may have in understanding the cognitive model, which has been used in the preparation of a hypertext. User modeling and interface development techniques are necessary to build effective interfaces for the use of a hypertext.

Belkin, Marchetti and *Cool* present in [7] the design of an interface supporting BRowsing And QUEry formulation (BRAQUE) to a large bibliographic information retrieval system. The interface scheme is based upon a progressive development of the capabilities that the final interface is going to have. The framework for the interface is articulated on the basis of an information seeking strategy model, a cognitive task analysis and a two-level hypertext model for information systems. The design reported in the paper has been translated into a prototype of an operational system.

Pollard in [58] reports on work that provides an online thesaurus as an interface which helps the end-user in his or her information search. The thesaurus is presented to the user as a browsing interface implemented through a hypertext. Through this interface the user uses the information stored in a bibliographic database. The paper presents the design and implementation of the interface established using a commercially available hypertext system.

1.3 Contributions

A hypermedia information retrieval system makes use of the specific capabilities of hypermedia systems together with information retrieval operations and provides new kind of information management tools. It combines both hypermedia and information retrieval to offer end-users the possibility of navigating, browsing and searching a large collection of documents to satisfy an information need. TEXPROS is a perfect application to apply hypermedia information retrieval techniques.

HyTEXPROS aims to enhance TEXPROS by adding hypertext functionalities, such as nodes, typed and weighted links, anchors, guided-tours, network overview, bookmarks, annotations, and an external linkbase.

HyTEXPROS's major contributions are the following: (1) It describes the entire information base including, the metadata and the original documents, as network nodes connected by links. Through hypertext functionalities, a user can construct dynamically an information path by browsing through pieces of the information base. By adding hypertext functionalities to TEXPROS, HyTEXPROS is created. (2) It changes its working domain from a personal document processing domain to a personal library domain accompanied with citation techniques to process original documents. (3) A four-layer conceptual architecture is presented as the system architecture of HyTEXPROS, which still keeps the Document Type Hierarchy and Folder Organization properties. Such architecture is also referred to as the reference model of HyTEXPROS. (4) Detailed description of HyTEXPROS, using the First Order Logic Calculus, is given. (5) A HyTEXPROS prototype. The HyTEXPROS prototype successfully embeds hypertext functionalities into the original TEXPROS system, which demonstrates the correctness of the architecture and model of HyTEXPROS.

We summarize the relationship between HyTEXPROS and TEXPROS in Figure 1.2. In this figure, the central dot line rectangle includes all the components of HyTEXPROS.

1.4 Outline of the Dissertation

The remainder of the dissertation is organized as follows. Chapter 2 presents the rationale of HyTEXPROS. Firstly, it introduces TEXPROS, secondly it proposes the conceptual architecture of HyTEXPROS in detail. Chapter 3 gives a detailed description of document citation techniques and the relationships with conceptual architecture in chapter 3. Chapter 4 discusses all kinds of HyTEXPROS's hypermedia functionalities including nodes, typed and weighted links, anchors, guided-tours, network overviews, bookmarks, annotations and an external linkbase . Chapter 5 introduces the formal description of HyTEXPROS using first order logic calculus. Chapter 6 gives the entire picture of the system architecture of HyTEXPROS. Chapter 7 describes the implementation of the HyTEXPROS prototype in detail and also gives the evaluation method of the system. And chapter 8 concludes this dissertation by comparing HyTEXPROS with other hypertext information retrieval systems, outlining potential research directions based on extension of HyTEXPROS, and also identifying HyTEXPROS's contributions and limitations.

Document Type Hierarchy
Thesaurus
Folder Hierarchy

DTH   FO

Retrieval
Subsystem

Scanner

Classification
Subsystem

Apply Hypermedia
Functionalities

Conceptual
Level

Hypermedia
Components

Template
Level

Entity
Level

Document
Level

Document Base

Frame Instance
Base

Document
Citation

HyTEXPROS

Information
Extraction

Filing
Subsystem

Storage
Subsystem
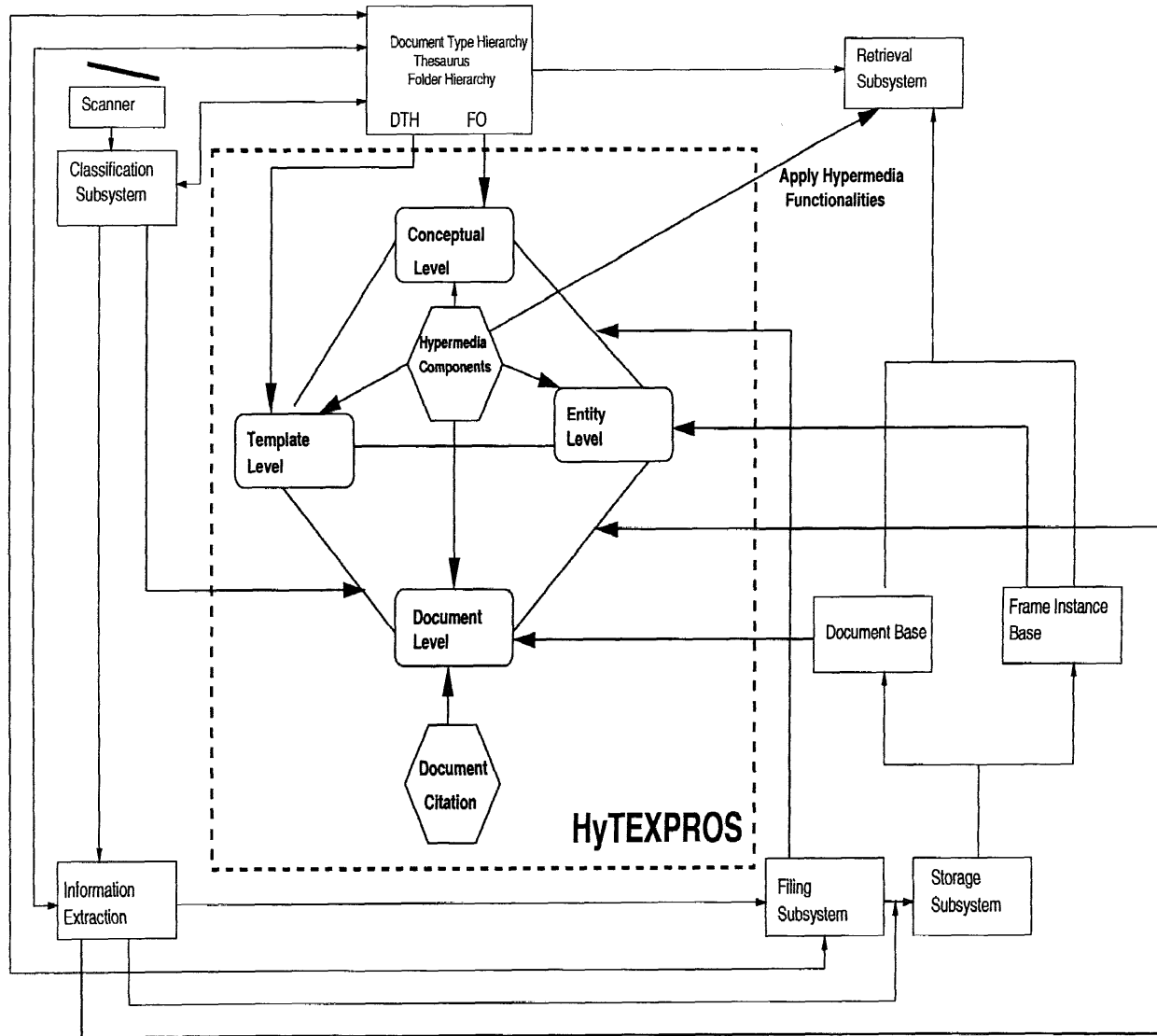
Figure 1.2 HyTEXPROS in TEXPROS

18

CHAPTER 2

TEXPROS'S DATA MODEL

In this chapter, we give the description of the original TEXPROS system and its sophisticated dual model architecture. We also present what are the shortcomings of the browser subcomponent within TEXPROS system and how HyTEXPROS system will improve them.

2.1 TEXPROS Approach

TEXPROS is an intelligent document processing system. The system is a combination of filing and retrieval systems which supports storing, classifying, categorizing, retrieving and reproducing documents, as well as extracting, browsing, retrieving and synthesizing information from a variety of documents. TEXPROS distinguishes itself from other systems by employing a dual model approach to describe documents and to adopt an intelligent agent-based architecture that allows us to automate document filing and file reorganization [24, 88]. The dual-model approach separates the *folder* structure from the document type structure, providing two independent, but integrated, bases for categorizing and accessing documents.

TEXPROS uses an object-oriented approach to model documents. Documents are grouped into *classes*. Each class is associated with a semantic document type to describe the common properties of the class of documents. A data structure called a *frame template* is employed to represent a document class type. A frame template can be instantiated by filing its attributes with values extracted from the original document. The instantiated object is a *frame instance* that represents the synopsis of an original document. This novel approach enables us to describe succinctly the important contents of a document. All the frame templates are constructed in the *system catalog* [48]. Given a document, the document classification subsystem

A Model for Hypertext-Based Information Retrieval

Dario Lucarella

Department of Science and Information

University of Milano

Via Moretto da Brescia 9, I-20133 Milano, Italy

Abstract: This paper approaches the problem of information retrieval from hypertext. In this context, the retrieval process is regarded as a process of inference that can be carried out either by the user exploring the hypertext network (browsing), or by the system, exploiting the hypertext network as a knowledge base (searching). That is the reason why a comprehensive model should take into account both of the perspectives, combining effectively browsing and searching in a unified framework. In the following, such a model is defined and implementation issues are outlined for a hypertext-based information retrieval system.

Keyword: Hypertext Models, Information Retrieval, Knowledge-based Systems, Intelligent Searching, Plausible Reasoning

Section 1 Introduction

There is a growing interest today in hypertext as a key technology for the implementation of massive multimedia information systems. Hypertext can simply be defined as a system to manage a collection of information that can be accessed non sequentially. It consists of a network of nodes and logical links between nodes. The variety of nodes and links that can be defined make hypertext a very flexible structure in which information is provided both by what is stored in each node and by the way the information nodes are linked each

Figure 2.1 An original article

determines the type of the document by identifying its corresponding frame template. Then the information extraction subsystem forms its frame instance by instantiating the frame template. Given a document as shown in Figure 2.1, the selected frame template and the corresponding frame instance of this document are shown in Figure 2.2. The frame instance contains only the most relevant information of the document. Here, we should emphasize that there are two special attributes contained in the frame template. A citation attribute contains the bibliographic information, and an annotation attribute contains comments from different readers.

Frame instances are grouped into various folders. A *folder* is a logical repository of documents comprising a set of frame instances of various types. Rather than being directories (in the operating system sense), the notion of folders represents the user's logical file structure. We could consider frame instances as grouped into a folder on the basis of user-defined criteria, specified as *predicates*, which determine whether a frame instance belongs to a folder. Folders can be naturally organized into *folder organizations* for which the basic graph model is a tree, where there is an edge from folder a to folder b if b is a subfolder of a (i.e. every frame instance of b is in a). The tree model for a folder organization generalizes naturally to a DAG(*Directed Acyclic Graph*) *Folder Organization*, where the underlying modeling graph is a rooted DAG, each vertex corresponds to a folder specified by its global predicates, and a designed vertex , which corresponds to the root folder, is the starting point of document filing. The detailed explanation of how to establish a folder organization and the filing procedure is given in [23].

Folders are heterogeneous repositories and are related by an inclusion relationship to form a folder organization. This folder organization is defined by a user corresponding to the user's view of the document organization, which is obtained by repeatedly dividing documents of particular areas of discourse into groups until well-defined groups are reached. To manipulate the file structure, a set of operations can be applied on folders. These operations include *insert, merge, delete* and *link*. Formal definitions and semantic descriptions of these operations can be found in [48, 83, 88].

In TEXPROS, we employed the concept of frame templates and frame instances at the operational level and the system level, both of them are used to describe the system catalog [41]. The system catalog as shown in Figure 2.3 in TEXPROS contains the metadata of the database (about the document type hierarchy and the folder organization) and the database itself (frame instances). The system catalog

Title		
Authors	Number	
	Name	
	organization	
	Name	
	organization	
	Name	
	organization	
Publication		
Volume		
No.		
Location	City	
	State	
	Country	
Page		
Abstract		
Keyword		
Citation		
Annotation		

Title	A Model for Hypertext Information Retreival	
Authors	Number	1
	Name	Dario Lucarella
	organization	University of Milano
	Name	
	organization	
	Name	
	organization	
Publication	Information Processing & Management	
Volume	33	
No.	2	
Location	City	London
	State	
	Country	United Kingdom
Page	133-144	
Abstract	This paper presents	
Keyword	Hypertext Model, Information Retrieval, knowledge-based System, Intelligent Searching, Plausible Reasoning	
Citation	P. Brown "Linking ans searching within hypertext" Electro Publishing, vol.1(1) , pp 45-33. 1988 J. Conklin,"Hypertext:An Introduction and Survey", IEEE Computer, vol.20, n9, pp.17-41, 1987	
Annotation	1 Good starting paper for HIR. 2 Use Artifial Intelligent techniques in HIR.	

Figure 2.2 A frame template and a frame instance

is an important facility which provides the capability of managing and maintaining the consistency and integrity of the data stored in the storage. The information that makes up the metadata is represented by the following five system frame templates: SYSFOLDER, SYSFRAMETEMPLATES, SYSFOLDERHIERARCHY, SYSATTRIBUTES, SYSFTHIERARCHY, SYSFRAMEINSTANCES. The frame instance of the type SYSFOLDERS contain the information of each folder in terms of the folder name, its filing predicate and threshold. It also describes the type of frame instances in the folder in terms of their types. They are specified in terms of the attributes "FTNames" and "FrameInstanceIDs", which are repeatable attributes. A set of frame instances of the type SYSFOLDERHIERARCHY is used to describe the structure of a folder organization. The repeatable and composite attribute "Depends_On" is used to specify the parent folders of a folder in terms of the parent folder name, and their linkage information in terms of the link type with a label. The repeatable and composite attribute "Parent_of" is used to specify the children folders of a folder and their linkage information. A frame instance of the type SYSFRAMETEMPLATES describes a frame template, which is defined by a set of attributes. A repeatable and composite attribute "Attribute" is used to describe the detailed information of each attribute contained in the frame templates, "FTName". A set of frame instances of the type SYSFTHIERARCHY describes the structure of a document type hierarchy. For each frame template, "FTName", it stores its only parent frame template, "Parent_FTName", and a number of its children frame templates, "Child_FTNames". A set of frame instances of the type SYSATTRIBUTES gives all the frame templates, "FTNames", which has the attribute, "AttributeName". A set of frame instances of the type SYSFRAMEINSTANCES describes the folders "FolderNames" which have the frame instance, specified in terms of "FrameInstanceID" of the type "FTName".

Frame Template SYSSYNONYMS

KeyTerm	
SynKeyTerms	

Frame Template SYSNARROWER

KeyTerm	
NarrKeyTerms	

Frame Template SYSTERMASSOC

KeyTerm	
IndexTm	
IndexTmType	

THESAURUS

Frame Template SYSFOLDERS

FolderName	
Predicate	
Threshold	
FTNames	
FrameInstanceIDs	

Frame Template SYSFRAMETEMPLATES

FTName		
	AttibuteName	
	Ancestors	
	Whether_Rept	
Attribute	WhetherS/C	
	DataDomain	
	CompTemplate	
FolderNames		
FrameInstanceIDs		

Frame Template SYSFOLDERHIERARCHY

FolderName		
	FolderName	
Depends_on	Linktype	
	Label	
	FolderName	
Parents_of	Linktype	
	Label	

Frame Template SYSFTHIERARCHY

FTName	
Parent_FTName	
Child_FTNames	

METADATA

Frame Template SYSFRAMEINSTANCES

FrameInstanceID	
FTName	
FolderNames	
original_Document	

Frame Template SYSATTRIBUTES

AttributeName	
FTNames	

Figure 2.3 A system catalog structure

In addition to reflecting the metadata of the document filing organization, the system catalog also includes a thesaurus. The thesaurus comprises three major components. The first component contains synonymous keyterms. The second component describes the terms that have semantic associations with keyterms. The third component describes the associations of the keyterms in terms of folders, frame template and attributes.

2.2 The Browser Subcomponent

Within the information retrieval component of TEXPROS, there is a browser subcomponent [47, 48, 82] and an object network (ON)[81] is created to describe the view of the meta-data of TEXPROS and the documents. The object network regards folders, frame templates, attributes and values as four kinds of objects. One of the characteristics of the object network is that the network captures the dual model and the information of all stored documents; and provides a snapshot of the system catalog. But an object network has drawbacks. (1) It is not a trivial task to identify the related information from an object in the realm of the object network. (2) The connection between the objects in the object network and the frame instances in the frame instance base is lost. Without this connection, the browsing process cannot be performed at the frame instance level.

Since object network cannot fully support the associations between the frame instances and the objects, we need to transform object network into another network – called operation network (OP-Net). Details of the transformation can be viewed in [81]. OP-Net has the following characteristics: it supports the browsing process; answers any questions related to the system catalog; and retrieves the documents. Using this network, the user can gain more knowledge about the document and information bases to form an exact query. The main difference between OP-Net and object network are as follows: In OP-Net, there is only one kind of link – called *relates*. Documents are not explicitly represented in the OP-Net and therefore the size of the network is small. And finally the OP-Net is constructed dynamically based upon the arrival of users' query.

Figure 2.4 shows a library's logical structure. Figure 2.5 depicts the OP-Net which corresponds to a user's query, namely *topic*: PUBLICATION [82]. Each object in the operation network is described in terms of metadata and data elements. The metadata are presented in three vertical levels: the folders, the frame templates, and

the attributes. As shown in Figure 2.5, PUBLICATION is a folder; CONF-ARC and JOUR-ARC are two frame templates, and AUTHOR is a common attribute of these two frame templates. Attribute values, which are specified at the fourth level of the OP-Net, are data elements. For example, the attributes CONFERENCE and LOCATION of the frame template CONF-ARC have the values SIR and NYC, respectively. In addition to the four vertical levels, the OP-Net also integrates the folder organization and document type hierarchy using the horizontal representation, which represents the relationships among folders and the relationships among frame templates. In a folder organization, there are two kinds of relationships: *is-parent-of* and *depends-on*; and in a document type hierarchy, there are also two types of relationships: *is-a-supertype-of* and *is-a-subtype-of*.

However, this OP-Net has its own shortcomings. First of all, the original documents are not involved in the OP-Net. The OP-Net does not provide users with any mechanism to move to the original documents from the metadata. This may happen after a user goes through the metadata to collect all the knowledge about the document to be retrieved. However, the users may want to traverse through the original documents. A good example is that from a given paper the user may go through all the papers which are cross-referenced. Secondly, it is rather difficult to implement the OP-Net if there are too many values at the fourth vertical level, which lead to the scale problem. Thirdly, the use of vertical and horizontal descriptions in the OP-Net has a limitation of mimicking the complete information base. For instance, the OP-Net will fail to express the relationship among values. For example, as shown in Figure 2.5 , the network fails to state clearly whether Wu and Motro are the co-authors of an article that appeared in the SIR conference; and whether the SIR conference is sponsored by ACM. Fourthly, the user may select objects from the OP-Net to gain more knowledge by proceeding to the *topic refining process*. The problem is that these topics must be predefined by the system, and therefore users have to
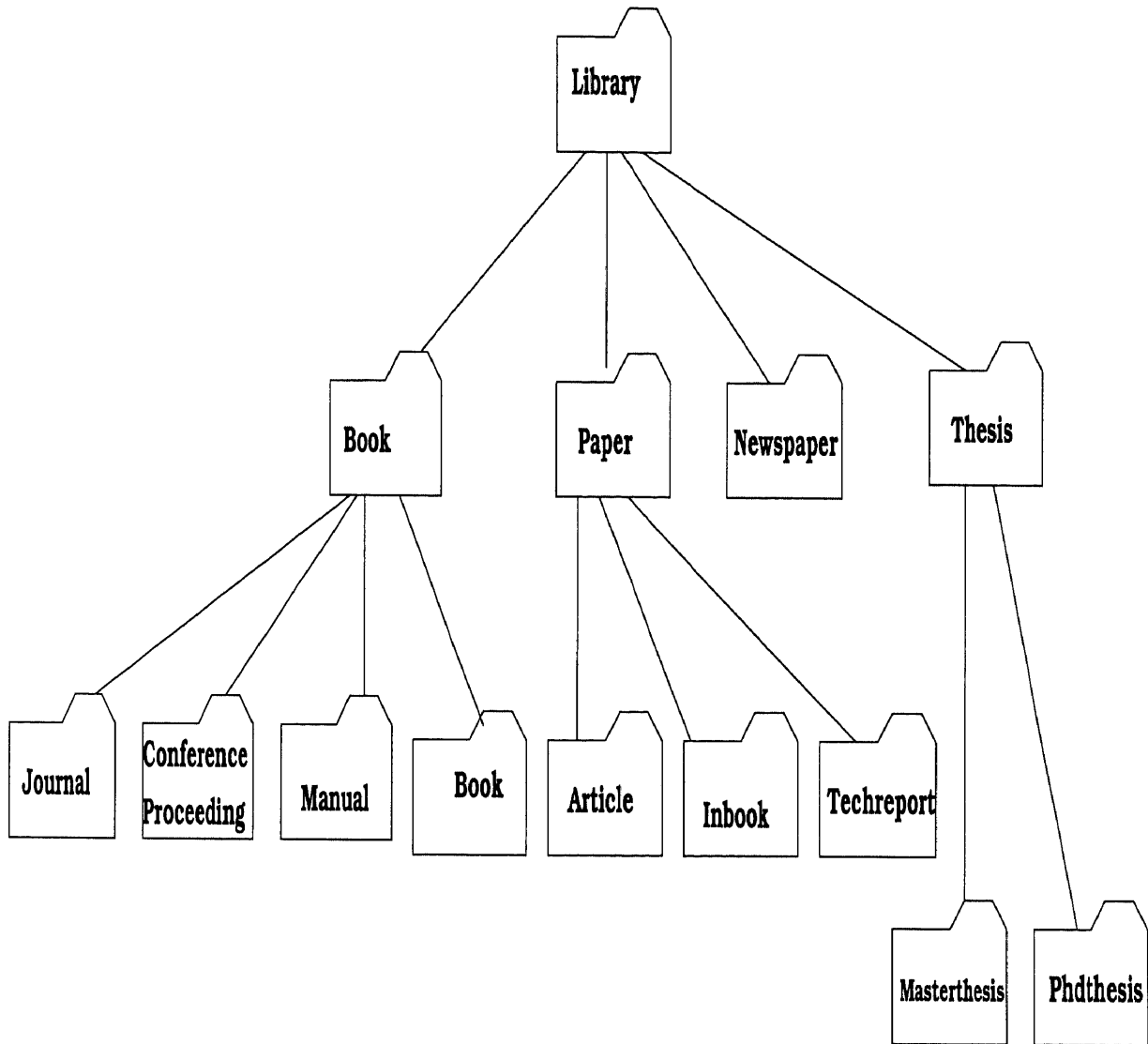
Figure 2.4 A library's logical structure

Folder

(is-parent-of, depends-on)

Frame Template

(is-a-subtype-of,

is-a-supertype-of)

Attribute

Value

PUBLICATION

CONF-ARC

JOUR-ARC

CONFERENCE

LOCATION

SPONSER

TITLE

AUTHOR

VOLUME

JOURNAL

SIR

NYC

ACM

WU

MOTRO

HYPERMEDIA

IEEE COMPUTER
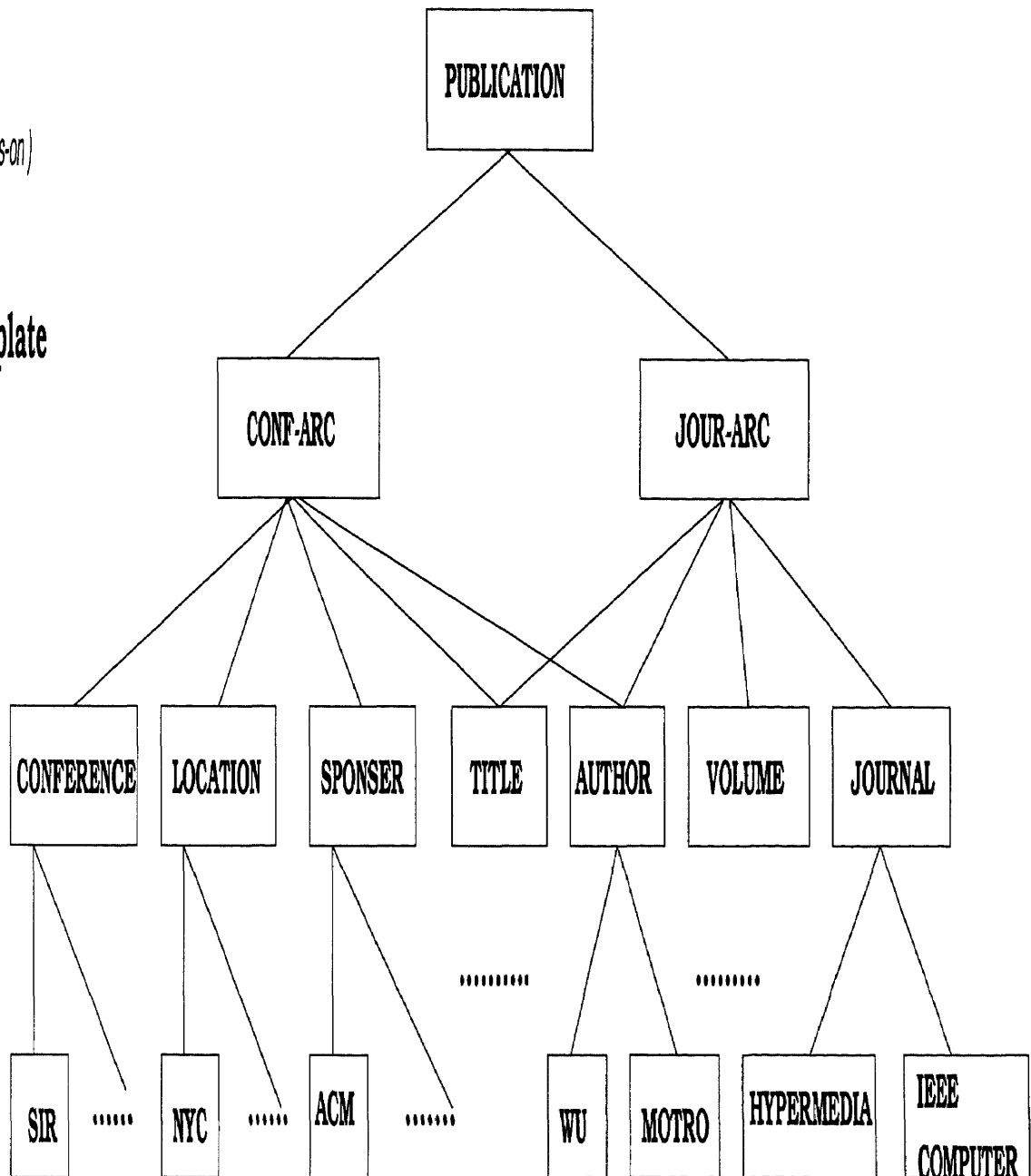
Figure 2.5 An operation network

select from this predefined group of topics and could not define the topics of their own, and at the same time, it does not give users the flexibility of adding their own links with annotation to meet their special requirements. Fifthly, the thesaurus in OP-Net is quite limited, it only includes two components, one is SYSSYNONYMS, the other one is SYSNARROWER, although we know the relationships inside thesaurus could be scope, equivalence, hierarchical and associative [74].

By incorporating hypertext functionalities into TEXPROS, the extension of TEXPROS to HyTEXPROS, allows us to infer implicit semantic connection between all kinds of nodes and to give end-users a number of ways of navigating, browsing and searching the system. In HyTEXPROS, hypertext nodes represent the TEXPROS entities such as folders, frame templates, frame instances , original documents, attributes and values. And hypertext links represent node relationships and operations. The whole system conceptual architecture and detailed description of link types is given in the next chapter.

CHAPTER 3

CONCEPTUAL ARCHITECTURE OF HYTEXPROS

In this chapter, we present the conceptual architecture of HyTEXPROS, which is a generalized architecture from the original TEXPROS design. We also give a detailed construction procedure for creating the architecture layers.

3.1 HyTEXPROS's Conceptual Architecture

In order to enhance TEXPROS with hypertext functionalities and at the same time retain TEXPROS's components already has, we propose a conceptual architecture[68, 67, 66] depicted in Figure 3.1. HyTEXPROS consists of four levels, namely, *the Document Level, the Entity Level, the Template Level* and *the Concept Level.* From this figure, we know that document level is the lowest level of abstraction, entity level and template level are on the top of it, and concept level is at the highest abstraction level.

This architecture enables users to navigate and browse the document base in a natural way. It allows users to browse through documents, frame templates, frame instances and folders. The network structure, which allows users to navigate, is built automatically from a collection of documents. The conceptual architecture must include the abstraction mechanisms that are necessary for the semantic structuring of the entities and concepts.

3.2 The Conceptual Paradigm

Three essential abstraction mechanisms are: the classification mechanism, the generalization/specialization mechanism and the aggregation mechanism.

The Classification Mechanism is a fundamental and intuitive one. It identifies the document types and other application object types as entities; it also

30
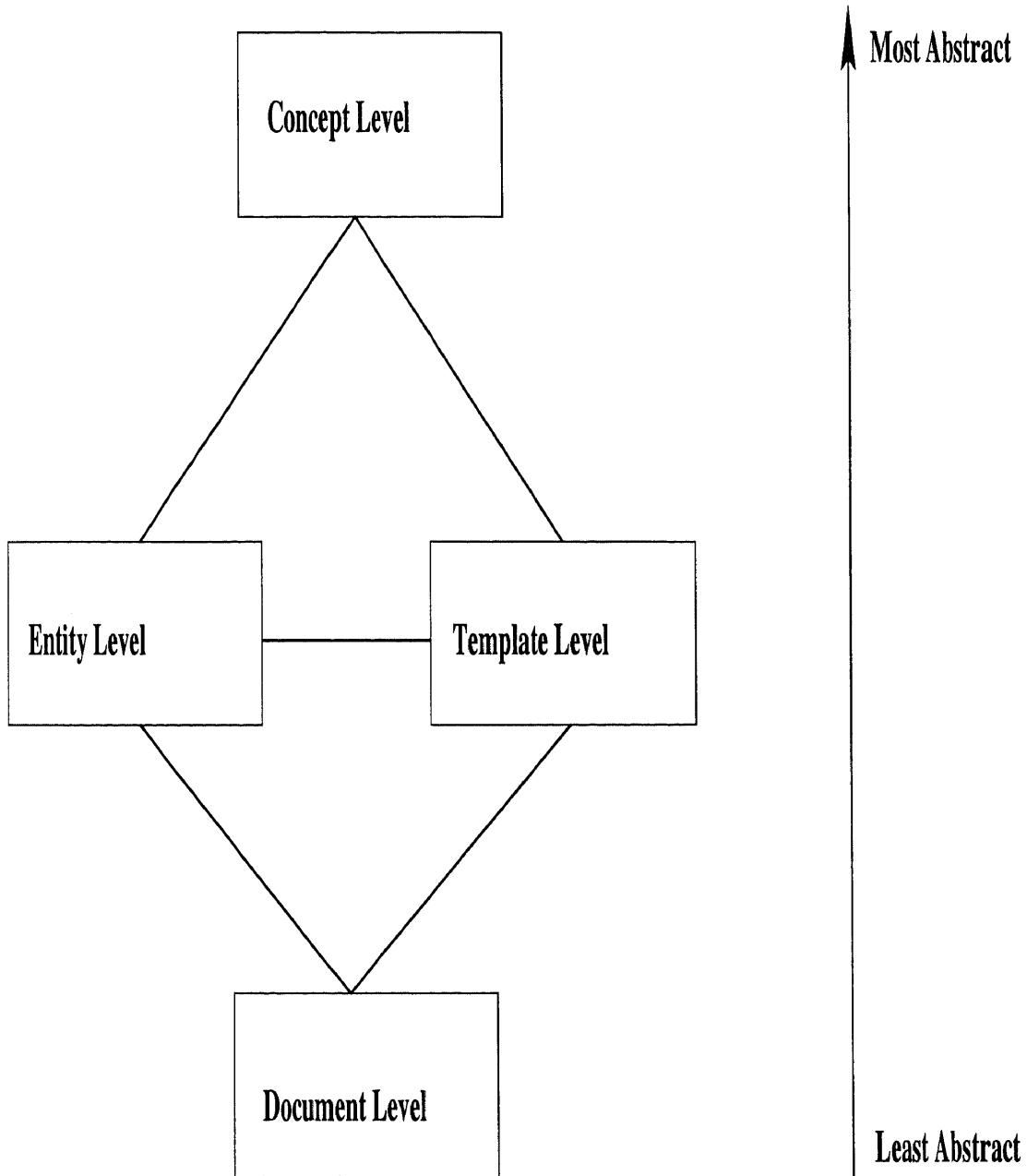
Figure 3.1 A conceptual architecture

identifies the objects that are going to play the role of entities of an application. A class and its instance are related by means of an *"instance-of"* relationship. The instance-of relationship between a document and a class is implemented by associating the frame instance which identifies the particular frame template class to the document.

The Generalization/Specialization Mechanism relates a set to its subsets or a class to its subclasses. The representation of the generalization/specialization mechanism is usually expressed as a "subset-of" or "subclass-of" relationship. Generalization/specialization indicates property inheritance between classes. That is, if class A is the generalization of class B, then we can construct B based on A. B will inherit all of the properties of A and will also have its own additional properties. These properties include attributes and methods (or operations) applicable to the individual classes. An example can be given by examining the relationship that exists between folders. This is-a relationship and the mechanism of inheritance helps to reduce the complexity of models and the redundancy in specification [73].

The Aggregation Mechanism relates objects of the same or of different types by transforming a relationship between several objects into a single object of a higher level. This new object can have specific individual characteristics of its own. For example, a folder can have different kinds of frame instances.

3.3 Conceptual Modeling of HyTEXPROS Data

At the document level, which is the lowest conceptual level, the elementary objects of interest are contained. Each object is referred to as a document. Each document has its own identity; the identity of the document is independent from how it is represented or structured. The identity of the document is represented by an identifier. Documents can be related to each other by means of bibliographic citation or by statistically determined similarity relationships. If a document cites another

document, then there corresponds a direct link from one node to the other. Such a link is referred as a document-document link.

At the template level, templates in HyTEXPROS are actually frame templates in TEXPROS. The relationship between different templates embed the document type hierarchy (DTH). As a powerful abstraction to share similarities among document classes while preserving their differences, the frame templates are related by specialization/generalization. They are organized in a document type hierarchy whose members are related by an is-a relationship. Document Type Hierarchy is established or initialized by a learning process in the classification subsystem of TEXPROS. At the template level, each node corresponds to a frame template of TEXPROS's document type hierarchy. For a template-template link from a template node T1 to another template node T2 in HyTEXPROS, there must be a correspondence between two frame templates T1 and T2, such that T2 is a subtype of T1. For a template-document link, there corresponds to a document which is of type of the class represented by the template. A typical example is shown in Figure 3.2 which is a extension of Figure 2.4.

At entity level, this level arises from the application of the classification abstraction mechanism to the elementary objects of the first level. For abstraction at this level, each entity characterizes a class of documents on the basis of their semantic content. Entities are indeed frame instances. That is, these documents are members of the class identified by the specific entity. Entities are established during extraction process. The relationships between entities and documents is one-to-one correspondence. That means, one entity links with one and only one document and at the same time, one document links with one and only one entity. The frame instance is used as the document representative instead of using a vector of key terms. By using frame instance, we incorporate not only the vector composed by key terms which is well known in the information retrieval research field [61], but
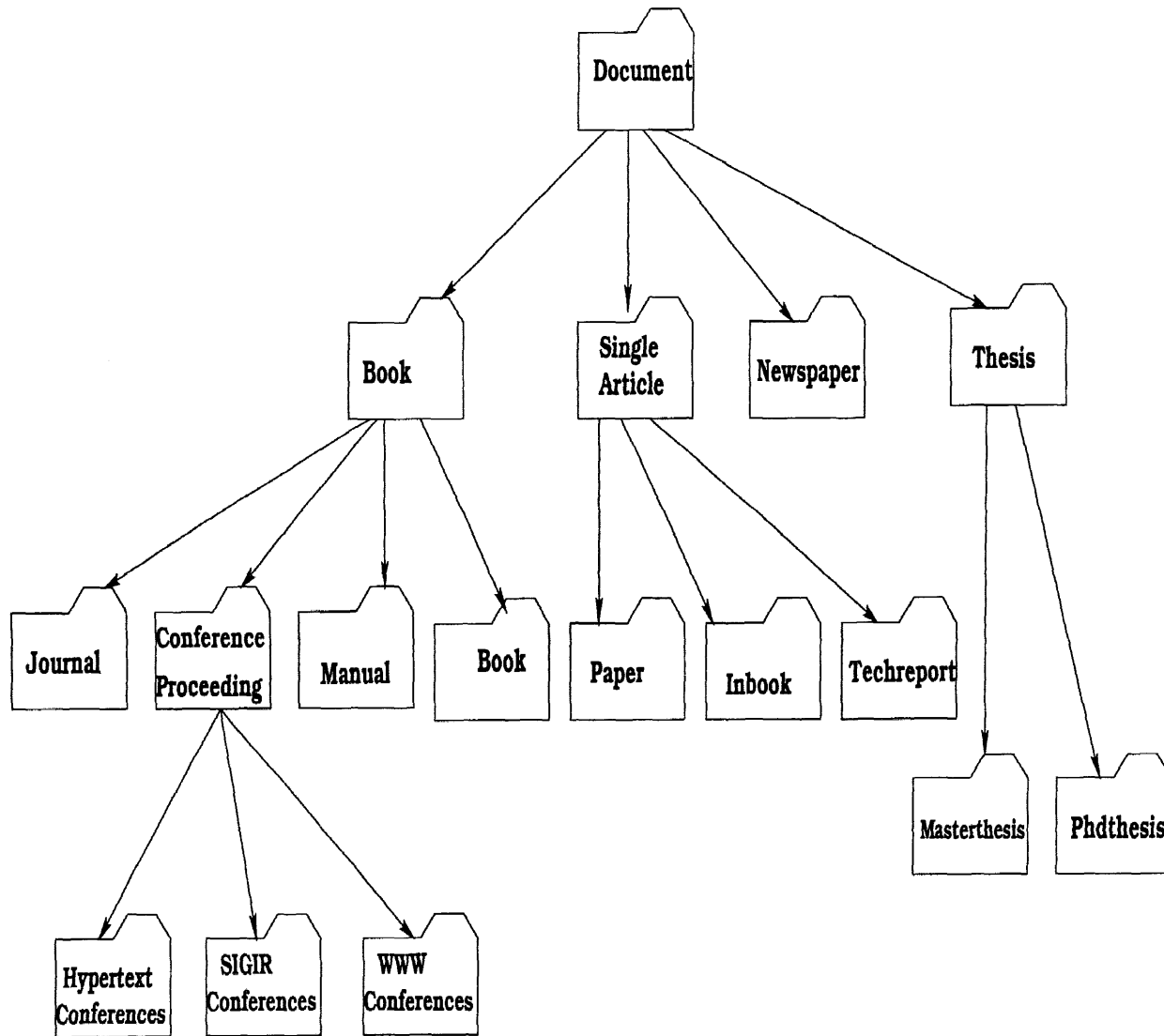
Figure 3.2 A Document Type Hierarchy

also the information about the metadata. Since the vector representation is a subset of the frame instance, all the mechanisms developed for the key term vector can still be used in our system to a certain extent.

We consider now all implicit relationships between entities. Firstly, through the relationship between two entities' corresponding documents, we know that these two entities are related to each other; Secondly, if two entities' corresponding documents link to the same template, then we also know that these two entities belong to the same frame template. This could be known automatically by having their attribute names of frame instances; Thirdly, if two entities link to the same concept, we know that they are included in the same folder. All of these implicit relationships are obtained through other types of links in the conceptual architecture. But one can ask about the direct relationship between entities. The relationship depends on the same values between two entities. For example, two articles written by the same author, the entities of these two articles have the same value of attribute – "author". Another example, two articles are published in the same journal issue, they have same journal name and publication date. The direct relationship between entities is established by the way the system or the user want to group entity information together. Once the entities are grouped together, their direct relationships are also created.

At concept level, concepts are actually folders in HyTEXPROS. The links between concepts depict the folder organization (FO) of TEXPROS. The links between two concepts are referred to the subfolders of two corresponding folders. Links between entities and concepts are established during filing process. A DAG structured folder organization is presented in Figure 3.3. In this figure, each node represents a folder and each edge (f_i, f_j) denotes that folder f_j is a child folder of f_i(or f_i is a parent folder of f_j). As its criteria, each folder has a user-defined predicate, called the local predicate, which governs the filing of frame instances into it. Each

of the folders is associated with its local predicate as stated beside the folder. To file a frame instance in the folder f_j from a folder f_i, it must satisfy the local predicate of the folder f_i, and in turn, it must satisfy the local predicate of the folder f_j.

This special conceptual architecture is always dynamically changing. It could happen that as one document is added/deleted at the document level, one template will be added at on the template level; one concept is added/deleted at the concept level. This architecture also must be initialized. The system initialization is that once we already have TEXPROS system, which means there exists a DTH, a FO, a document base, and a frame instance base. When the situation changes, the architecture will be modified consistently according to the dynamically changes.

This conceptual architecture provides a very powerful framework for navigating and browsing through the HyTEXPROS objects because it includes nine kinds of relationships and four kinds of nodes in one map. Four kinds of nodes are: document, entity, template and concept. Nine kinds of links are: document-document, document-entity, document-template, entity-entity, entity-template, entity-concept, template-template, template-concept, concept-concept. A detailed explanation of how to construct the nodes and links will be given in next section.

3.4 Detail of Construction

In this section, we shall present automatic authoring in HIR. Without using an HIR system, manual authoring is only feasible if the collection of the documents to be authored is manageable. This is because the process of manual authoring is time consuming. Moreover, manual authoring depends on the subjective criteria of an individual. In contrast, automatic authoring represents a way of constructing a hypermedia from a large collection of documents, without time limitations or the subjectives of an expert user. On the other hand, this conceptual model integrates
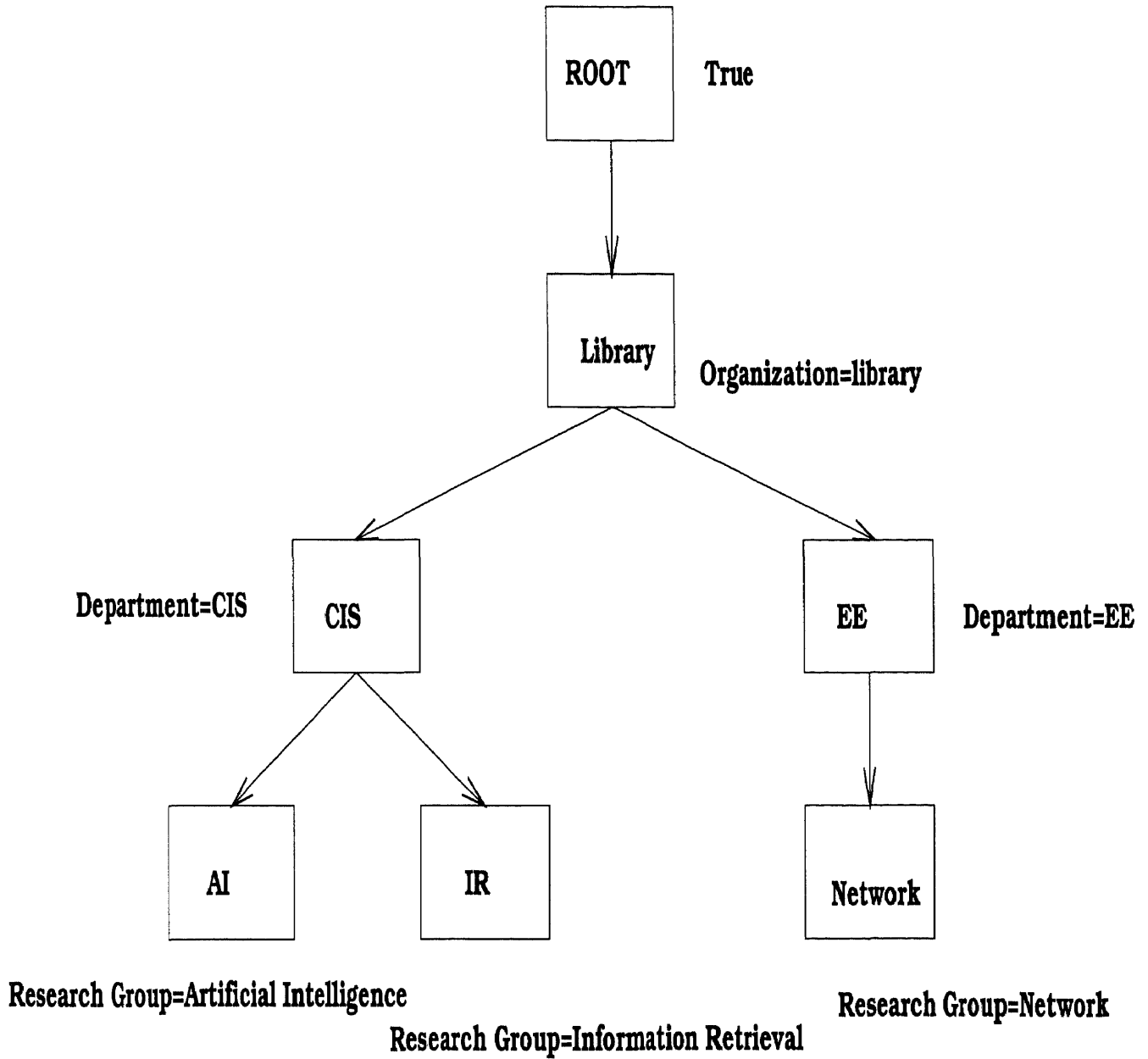
Figure 3.3 A Folder Organization

the original dual model of TEXPROS by incorporating the DTH and FO into the conceptual architecture.

3.4.1 Construction of the Collection of Documents

The design process begins with construction collection of a set of documents as shown in Figure 3.4.

We can convert existing paper-based documents into a form, retrievable by a computer. We use an Optical Character Recognizer (OCR) to transform every document in an electronic format such as ASCII. For each document, the content of its textual part is recognized and the description of its non-textual part such as logos, figures and pictures is extracted. The method of transformation from paper-based documents to electronic format is beyond the scope of this dissertation but the reader can refer to [39, 46].

We also can convert documents into HTML instances using a number of converters. The reason for transforming all the original documents into HTML instances instead of normal files is that we can preserve all the relationships between original documents. There always exist citing and cited documents. This kind of relationships, although they are not crucial in the office domain, are very important in the library domain.

3.4.2 The Hypertext Markup Language (HTML)

The *HyperText Markup Language*, or HTML, is designed to specify the logical organization of a text document, with important extensions for hypertext links and user interaction. HTML was not designed to be the language of a What You See Is What You Get (WYSIWYG) word processors, such as Word or WordPerfect. Instead, HTML requires that you construct documents with sections of text marked as logical units, such as titles, paragraphs, or lists, and leave the interpretation of these marked elements up to the browser displaying the documents.
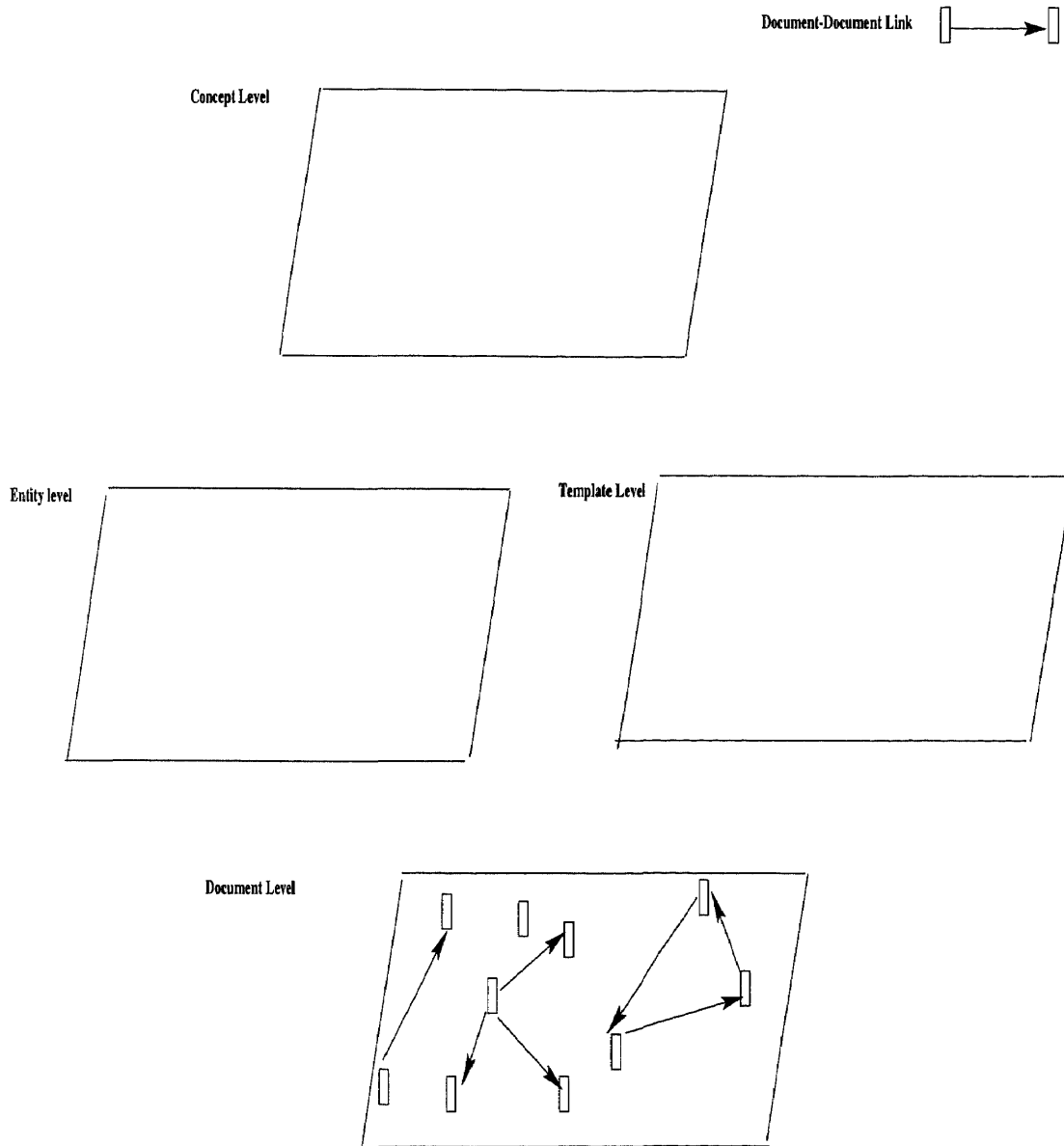
Document-Document Link

Concept Level

Entity level

Template Level

Document Level

Figure 3.4 Construction of the Documents Collection

A document representing the HTML mark-up language is called a HTML document. The difference between a HTML document and a simple text document lies in the HTML markup *tags*. These are the portions of text surrounded by the less than and greater than signs ($\langle$ .. $\rangle$) and are the instructions that tell the browser what each part of the document means. For example, the tag $\langle$H1$\rangle$ indicates the start of a heading of level 1, while the $\langle$/H1$\rangle$ tag marks the end of a heading of level 1. Thus, the text string: $\langle H1 \rangle$ *This is the Heading 1* $\langle /H1 \rangle$ marks the string "This is the Heading 1" as a level one heading. There are six possible heading levels, from H1 to H6.

We must observe that a HTML document must have the following minimal structure:

$\langle$HTML$\rangle$ beginning of the document

$\langle$HEAD$\rangle$ beginning of the header

$\langle$TITLE$\rangle$ beginning of the document identifier; if empty HyTEXPROS will automatically assign one

$\langle$/TITLE$\rangle$ end of the title

$\langle$/HEAD$\rangle$ end of header

$\langle$BODY$\rangle$ beginning of the content to be extracted

$\langle$/BODY$\rangle$ ending of the document

$\langle$/HTML$\rangle$ ending of the document

The document identifier will be used to link to other documents within the network.

The following examples consists of two distinct documents, 1.html and 2.html, with a hypertext link from one to the other, and the hyperlink is created because the first article is citing the second article. The documents in HTML format are shown in the following pages. They are the HTML source code and should be displayed by Web browsers like Netscape or Microsoft Internet Explorer.

```
<HTML>
<HEAD>
    <META HTTP-EQIV="Content-type" CONTENT="text/html; charset=iso-8859-1">
    <META NAME="GENERATOR" CONTENT="Mozilla/4.04 [en] (X11; I; SunOS 5.4 sun4d) [Netscape]">
</HEAD>
<BODY>
 
<H1>
Toward Support for Hypermedia On the World Wide Web </H1>
  Ashley Travis
<BR> <I> University of Bologna </I>
<BR>  

<P>  Many organizations will embrace the World Wide Web as their primary application infrastructure.
However, in the rush to acquire and retrofit Web application, they risk bypassing the Web's greatest
supplemental benefit.

<P>  Hypermedia lets you add, access and navigate links in textual and multimedia information. With
hypermedia features, Web applications can provide clearer information, and they can be easier and more
effective to use than traditional applications. We foresee a hypermedia-influenced era in which users will
expect (and application developers will provide) browsing and supplemental linking in all applications.

<P>.....................................
<P>  <B>Reference: </B>
<BR>  <A HREF="2.html">1. F. Garzotto, L. Mainetti, and P. Paolini, " Hypertext Design, Analysis, and
Evaluation Issues. " Comm. ACM, Aug. 1995, pp. 74-86. </A>

<P>2 J. Nielsen, Multimedia and Hypertext: The Internet and Beyond, AP Professional, San Diego, 1995

<P>3 E. Rivlin, R. Botafogo, and B. Shneiderman, "Navigating in Hyperspace: Designing a Structured-Based
Tollbox," Comm. ACM, FEB. 1994, pp 87-108.

<P>..............
</BODY>
</HTML>
```

Figure 3.5 1.html

```
<HTML>
<HEAD>
    <META HTTP-EQIV="Content-type" CONTENT="text/html; charset=iso-8859-1">
    <META NAME="GENERATOR" CONTENT="Mozilla/4.04 [en] (X11; I; SunOS 5.4 sun4d) [Netscape]">
</HEAD>
<BODY>

<P><B><FONT SIZE=+4>Hypertext Design, Analysis and Evaluation Issues </FONT></B>
<B><FONT SIZE=+1></FONT></B>

<P><B>F. Garzotto, L. Mainetti and P.   Paolini </B>
 <BR><B></B> 

<P>The information revolution is upon us. As enabling technologies continue to evolve and mature, the ability
to quickly and easily access information improves.   we now have access to so much information that it
is difficult, if not impossible, to sift through it and identify those areas relevant to us.

<P>Hypermedia lets us organize information in accordance with the ways in which we naturally access and
manipulate it. This tool presents information in such a way that we can readily see the conceptual associations
between information chunks, revealing the structure and interrelationships within the information.

<P>...........................
</BODY>
</HTML>
```

Figure 3.6 2.html

The reference part of Figure 3.5 shows hypertext links. The form is straight-forward: ⟨A HREF = "2.html"⟩ 1.F. Garzotto, L. Mainetti, and P. Paolini, "Hypertext Design, Analysis, and Evaluation Issues," Comm. ACM, Aug. 1995, pp. 74-86. ⟨/A⟩

Here is an example on how we could go from one article to its cited articles. The element marking a hypertext link is called an A, or *anchor* and the marked text is referred to as a hypertext anchor. The area between the beginning ⟨A⟩ and ending ⟨/A⟩ tags becomes a *hot* part of the text. With Netscape browser, this section of text is displayed with an underline and usually in a different color(often blue). Placing the mouse over this region and clicking the mouse button causes the client to access the indicated document or other Internet resource. You can also use images as hypertext anchors. The target of the hypertext link is indicated by the anchor attribute **HREF**, which takes as its value the **Uniform Resource Locator or URL** of the target document or resource. A URL is a text string that indicates the server protocol such as HTTP, FTP, file, news and etc. to use in accessing the resource, the Internet Domain Name of the server, and the location and name of the resource on that particular server.

The example above is an example of a *partial URL*, which is a shorthand way of referring to files or other resources relative to the URL of the document being currently viewed, which means these two file are located on the same sever and same file directory.

We also have a number of pre-existing files that we could convert to HTML format. If the documents were prepared by programs such as Microsoft Word, Latex or FrameMaker, there are translators or converters, which can convert the document to HTML format.

Table 3.4.2 matches conversion programs with standard word processors and document formats.

Table 3.1 Converters to HTML

Format	*Program*
Framemaker	frame2html, Webmaker
Latex	latex2html, hyperlatex
Microsoft Word	TagWrite
troff/nroff	mm2html
plain text	asc2html
PostScript	ps2html
WordPerfect	wp2x, cyberleaf
Rich Text Format(RTF)	rtftohtml

Many of these packages are script programs, often written in perl. Some of them are research projects, the other are commercial tools. Information about these packages appears on different web sites. The most important two packages are the following: (1) **Asc2html** is a perl script for converting plain ASCII files into HTML. (2) **TagWrite** is a commercial document conversion tool for Windows platform. It can read Microsoft RTF and WordPerfect documents and convert them into HTML format.

There are tools to verify HTML elements and check the grammar of HTML, and also tools to ensure that the hypertext links go to valid locations. The first set of tools are called HTML verifiers and the second are called link verifiers.

For an automatic setup to generate links between documents (document-document) links, it is possible to use statistical techniques. Other techniques for setting up a network of related documents makes use of bibliographic citations or co-authoring. We use bibliographic method because this method can be used to build up a network implicitly assuming that the documents cited by a document must be somehow related to a cited document and the document base we use are all the documents in a library. If document A cites document B, then we say there is a document-document link from document A to document B. It is possible to form

a circle. For example, document A cites document B, document B cites document C, and document C cites document A. In this situation, we say that document A, B and C are strongly connected and give users recommendation. For a detailed explanation about document citations, please refer to chapter 4.

3.4.3 Construction of the Collection of Templates in HyTEXPROS

At the template level, each node corresponds to a frame template of TEXPROS's document type hierarchy. For a template-template link from a template node T1 to another template node T2 in HyTEXPROS, there corresponds two frame templates T1 and T2, such that T2 is a subtype of T1. For a template-document link, there corresponds to a document which is of type of the class represented by the template. The document classification process is a method to find the best fitting type for a given document and produces the frame instance. In TEXPROS, the task of document classification is to determine the types of the office documents. That is, given an office document, the document classification subsystem identifies the corresponding frame template of the document. By identifying the defined type of the documents, it is possible to implement efficient storage and access methods to enhance the performance of retrieval. From TEXPROS, we could get DTH from classification subsystem [39, 85, 46]. The classification subsystem uses a learning process and tree matching to create the initial hierarchy. In TEXPROS, a classifier creates frame templates for the office documents in an office environment by sampling a stream of office documents, abstracting their general attributes, and group them into classes. The frame template, populated by the instances of particular office documents, yields an organized synopsis of the original document which we call a frame instance.

In the sample-based classification, instead of asking the user to generalize the relationships between the conceptual and layout structures of a document type, a

set of document samples of the same type are stored in an appropriate way so that a document can be classified with certainty if it belongs to a type of the pre-stored samples. This approach attempts to achieve reliability and efficiency of the document classification by maximizing the use of direct matching between a sample and a new document. A sample base is created by acquiring all samples of various document types from the user. A document is classified by comparing its layout and conceptual features against the samples in the sample base. An office document is first digitized and threshold into binary images by a scanner, then the document is encoded through the recognition system. After the recognition process, the content of its textual part is recognized and a description of the non-textual part of the document such as logo, figures, and pictures are extracted. The document classification and information extraction system begins with the layout process. During the process, the layout structure of a document is obtained in the form of nested segmentation of the document which is represented by a tree structure called the *Layout Structure Tree*. In the very first time of executing the system, the sample base is empty and the system is not able to process automatically the document. The user will enter the document type and the frame instance. This document will be learned as a document sample which is used later to facilitate the automatic classification and information extraction of other documents of the same document type. The sample base grows as more documents of different document types, or of the same document type but with different layout are processed by the system. The document type is identified based on its layout, conceptual and content structures.

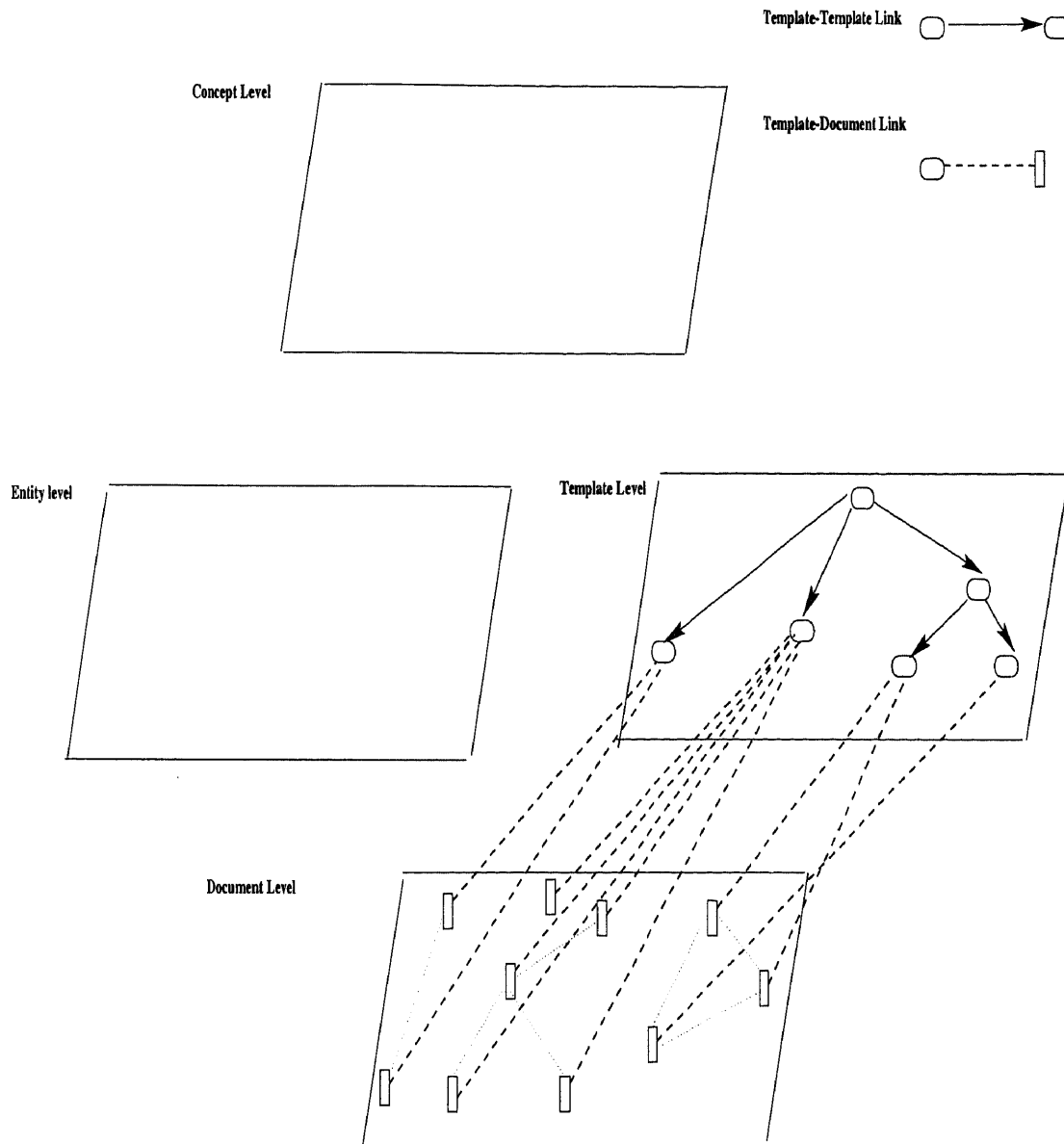We just need to transform it to the template level in HyTEXPROS. The DTH is a DAG structure with root represented as generic document. All the original documents are only connected to the leaves of the DTH. From the Figure 3.7, it shows that when two documents are related by citation, it is most likely that they belong to the same template, which means same document type. Once a new document comes

into the system, the classification subsystem needs to exam all existing templates one by one and put this newcomer into the correct template. If this newcomer does not belong to any of the existing templates, a new template needs to be created and added to the DTH.
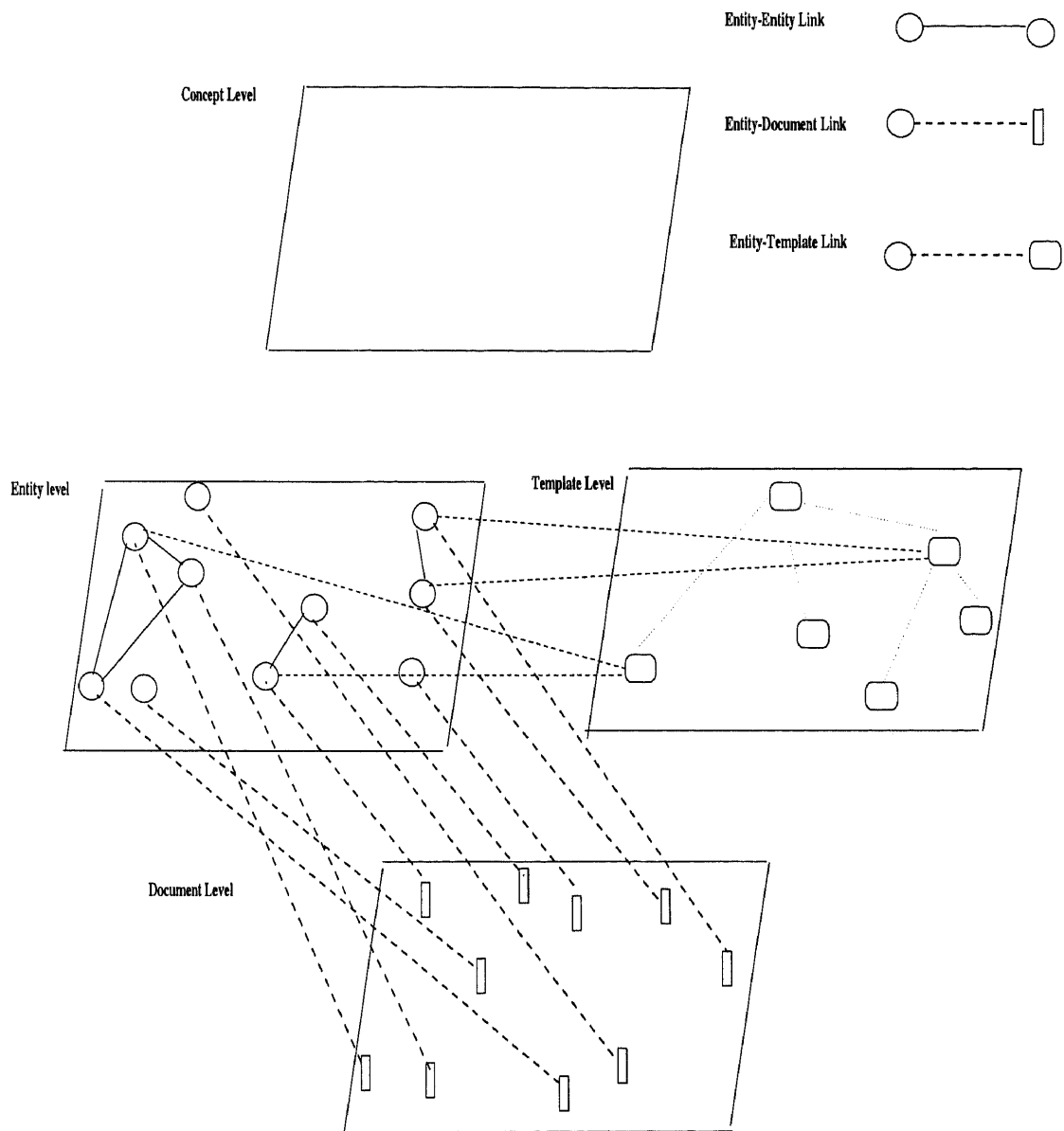
3.4.4   Construction of the Collection of Entities in HyTEXPROS

The extraction subsystem of TEXPROS creates a corresponding frame instance for a document. That is, given an office document, the information extraction subsystem forms its frame instance by instantiating its corresponding frame template. The document classification and information extraction can be achieved in aid of analyzing the document structure. After Extraction, a particular office document summarized from the viewpoint of its frame template, yields a synopsis of the document – frame instance. Once a frame instance is assigned to a document it is also linked to that document. In this way, it is possible to determine the document-entity links of the conceptual model as shown on Figure 3.8. This process creates the network of frame instances and links of a entity-document at the same time.

We consider first all implicit relationships between entities. Firstly, through the relationship between two entities' corresponding documents, we know that these two entities are related to each other; Secondly, if two entities' corresponding documents link to the same template, then we also know that these two entities belong to the same frame template. This could be known automatically by having their attributes of frame instances; Thirdly, if two entities link to the same concept, we know that they are included in the same folder. All these implicit relationships are getting through other links in the conceptual architecture. what about direct relationship between entities ? It depends on the same values between two entities. For example, two articles written by the same author, the entities of these two articles have the same value of attribute – author. Another example, two articles are published in the

Figure 3.7 Construction of the Collection of Templates

Concept Level

Entity-Entity Link

Entity-Document Link

Entity-Template Link

Entity level

Template Level

Document Level

Figure 3.8 Construction of the Collection of Entities

same journal issue, they have same journal name and publication date. The direct relationship between entities is established by the way users want entities information grouped together.
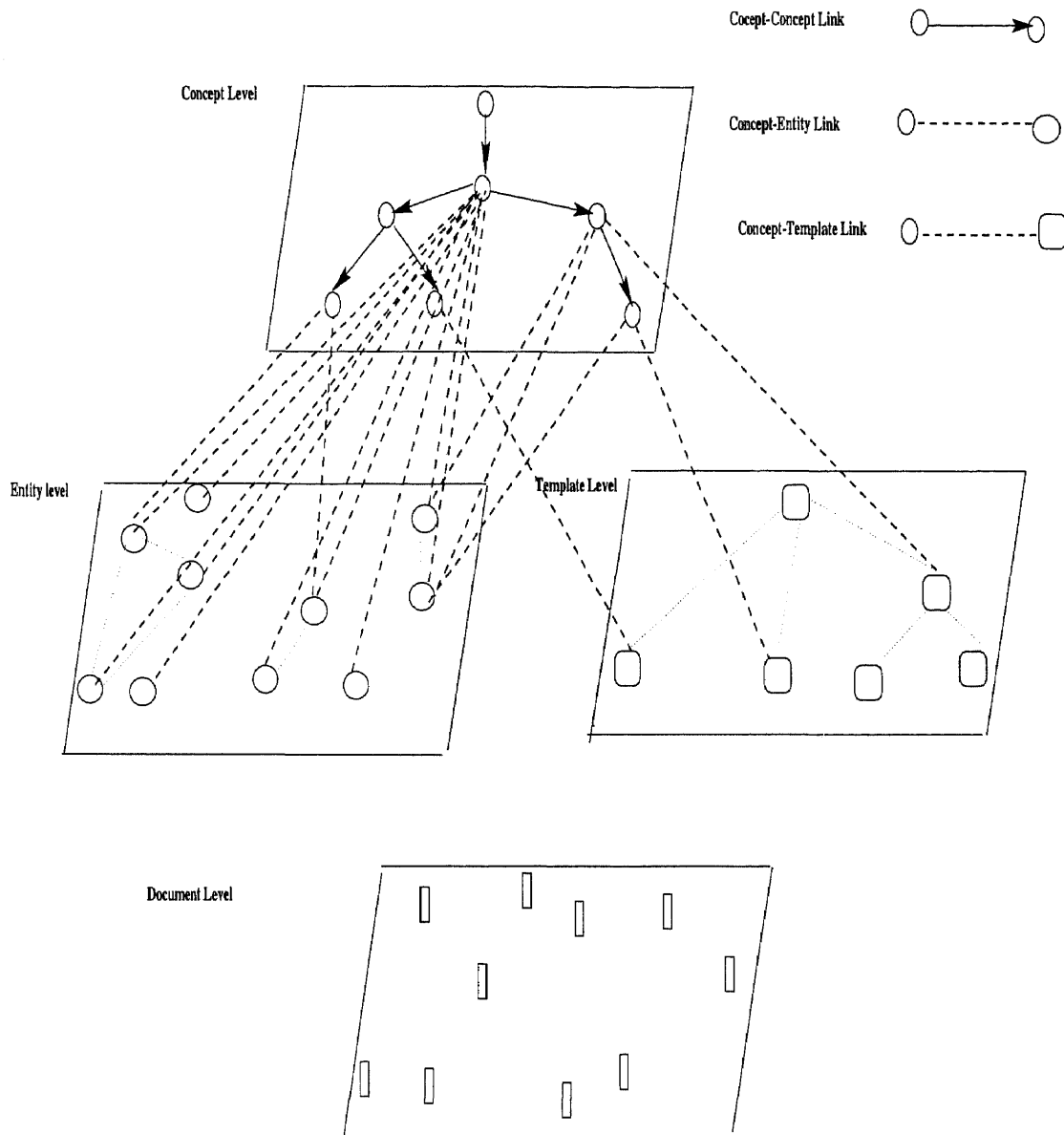
3.4.5 Construction of the Collection of Concepts in HyTEXPROS

At concept level, concepts are actually folders in HyTEXPROS. The links between concepts depict the folder organization (FO) of TEXPROS. The links between two concepts are referred to the subfolders of two corresponding folders. Links between entities and concepts are established during filing process. A DAG structured folder organization is presented in Figure 3.3. In this figure, each node represents a folder and each edge (f_i, f_j) denotes that folder f_j is a child folder of f_i(or f_i is a parent folder of f_j). As its criteria, each folder has a user-defined predicate, called the local predicate, which governs the filing of frame instances into it. Each of the folders is associated with its local predicate as stated beside the folder. To file a frame instance in the folder f_j from a folder f_i, it must satisfy the local predicate of the folder f_i, and in turn, it must satisfy the local predicate of the folder f_j.

At the concept level, concepts are created, and they are referred to folders. There is always a core part and a part that is added by the user. The core part of concepts is provided by the system, however users are allowed to add new concept into the core part. The system gives the users flexibility to add more concepts depending on their requirements as shown on Figure 3.9.

Links between concepts are bi-directional, whenever concept A is a parent of concept B, it is always true that concept B depends on concept A.

This construction procedure involves the filing process. From [23], an agent-based filing system is employed for this filing process, which can automate the process by depositing an incoming instance into appropriate folders and coping with the subtleties of the folder organization. Each concept is associated with a filing agent

Figure 3.9 Construction of the Collection of Concepts

containing a criterion. The criteria used to categorize documents are defined in predicate. The approach to filing entities into concepts is the method of depositing a entity into a concept if the entity belongs to the concept. We deposit a entity into a concept if the entity belongs to the concept, in this way, the same entity probably links to two related concepts. As shown in Figure 3.8, every entity links to one universal concept because all concepts belong to that concept. For the details on how to file entities, refer to [89, 23]. It is also possible that some entities do not belong to any concept. In order to solve the problem of false drop in the filing process, Xien Fan [23] introduces AND-Link and OR-Link.

From [23], several operations are also defined once the folder organization has been changed. They are: CreateConcept, DestroyConcept, LinkConcept, UnlinkConcept, RenameConcept. CreateConcept creates new concept. Destroy-Concept destroys an exist concept. LinkConcept links two concepts. UnLinkConcept removes the exist link between two concepts. RenameConcept changes the name of a concept to another name. Adding a link between two folders A and B will change the semantics of the folder organization. It requires to update the contents of same folders (the child folder B and its descendants, removes the frame instances that no longer belong to these folders, and deposits the ones that should belong to these folders. Removing a link from folder A to folder B, requires that the child B and B's descendant will update their repositories. Deleting a frame instance from the folder organization, one has to remove it from the root folder.

CHAPTER 4
DOCUMENT CITATION

In chapter 3, we gave the conceptual architecture of HyTEXPROS. At the document level of the conceptual architecture, bibliographic citations [70] in scientific papers can be used to establish relationships among documents, especially in a personal library system.

There are three kinds of citations [71] as show in Figure 4.1. The first citation is the direct citation, which is a new document citing an earlier document; the second citation is the bibliographic coupling, which specifies the sharing of one reference by two documents; the third is the co-citation, which describes the frequency of two documents each being cited together.

There are different aspects of the citation problem [65] to consider. Most important as a controlling element is the document type. For example, most survey and tutorial articles carry more extensive bibliographies than specific research reports. Similarly, articles covering a wide variety of topics may be cited more frequently than others that are more specialized. As a result, two articles that cover identical topics from somewhat different points of view may include quite distinct bibliographies. A second important criterion is the availability of the cited document. Thus, reports included in certain books or in important journals are likely to be cited more often than those not generally available to the public. The third important issue is the date of publication, which also affects the probability of being cited. Very recent documents which have not had a chance to circulate, and very old ones which no longer circulate are cited much less than current articles distributed within the recent past. The final consideration pertains specifically to the author of a document. In many cases, personal preferences are evident both as to number and type of papers cited. Authors have different backgrounds and there may also exist a tendency toward self-citation regardless of relevance.

Direct Citation:
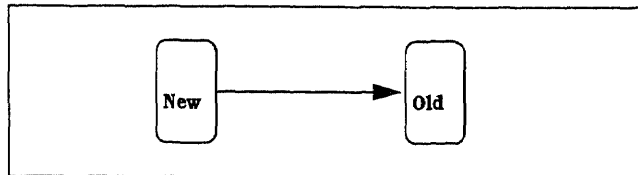
Bibliographic Coupling:
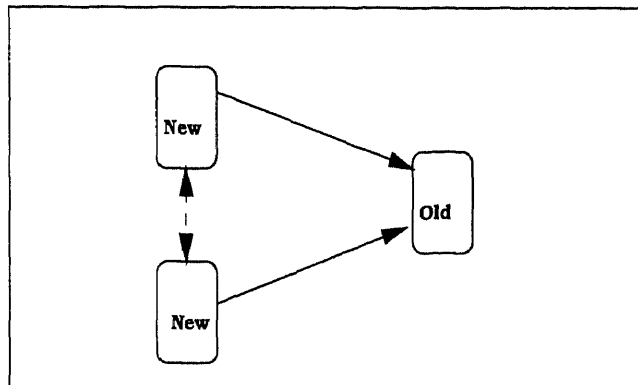
Co-Citation:
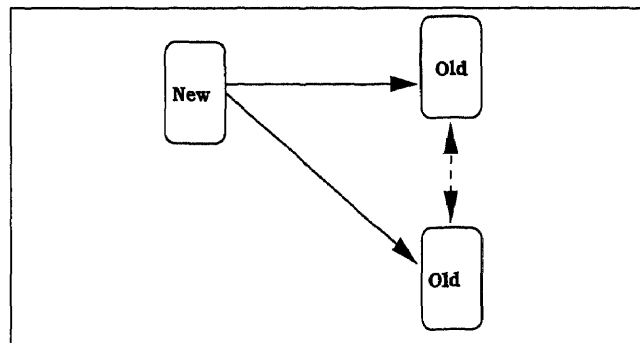
Figure 4.1 Three kinds of Citation

References contained in academic articles are used to give credit to previous work in the literature and provide a link between the "citing" and "cited" articles. A citation [28] indexes the citations that an article makes, linking the articles with the cited works. Citation were originally designed mainly for information retrieval [29]. The citation links allow navigating the literature in unique ways. Papers can be located independent of language, and words in the title, keywords or document. A citation allows navigation backward in time (the list of cited articles) and forward in time (which subsequent articles cite the current article?) Citation can be used in many ways, e.g. a) citations can help to find other publications which may be of interest, b) the context of citations in citing publications may be helpful in judging the important contributions of a cited paper and the usefulness of a paper for a given query [60], c) a citation allows finding out where and how often a particular article is cited in the literature, thus providing an indication of the importance of the article, and d) a citation can provide detailed analyses of research trends and identify emerging areas of science [30], which means trace the flow of information and measure the relative importance of various journals to the scientific community.

4.1 Citation Matrix

Consider a collection of m documents each of which is characterized by the property of being cited by one or more of the other documents in the same collection. Each document can then be represented by an m dimensional logical vector X^i, where X^i_j = 1 if and only if document i is cited by document j, and X^i_j =0 otherwise, as shown on Figure 4.2. If these m vectors are arranged in rows one below the other a square logical incidence matrix is formed similar to the matrix exhibited in Figure 4.2. The number of ones in the row vectors of X represents the degree of "citedness" for the documents listed at the head of the rows. Similarly, the number of ones in the column vectors of X measures the amount of "citing" for the documents listed at the head of

Cited Document	Citing Documents			
	D1	D2 ...		Dm
D1	0	X_2^1 ...		X_m^1
D2	X_1^2	0		X_m^2
. . .				
Dm	X_1^m	X_2^m		0

Figure 4.2 Matrix X Exhibiting Direct Citation

the columns. For a closed document collection, the set of row identifies is the same as the set of column identifiers.

4.2   Co-citation

The strength of co-citation between two cited papers can be easily determined from a citation index such as the *Science Citation Index (SCI)*. SCI is one of the index systems published by the Institute for Scientific Information (ISI), besides SCI, ISI also has *Social Sciences Citation Index (SSCI)* and *Arts and Humanities Citation Index (AHCI)*.

Each of the two papers is located in the citation index section of the SCI, and their list of citing papers can be scanned. The number of identical cited items defines the strength of co-citation between the two cited papers. An identically cited item is simply a new document that has cited both earlier papers. Therefore, co-citation is the frequency with which two items of earlier literature are each cited by a later document.

We can also give a more formal definition of co-citation in terms of set theory notation. If S_a is the set of papers which cites document a and S_b is the set of papers which cites document b, then the intersection of S_a and S_b, that is $S_a \cap S_b$, is the set which cites both a and b. The number of elements in $S_a \cap S_b$, that is $|S_a \cap S_b|$, the cardinality of $S_a \cap S_b$, is the co-citation frequency.

We could use an example to show how the algorithm works in Figure 4.3 . We use upper-case letter to represent new documents such as A, B, C and D, and lower-case letter to represent old documents such as a, b, c, d, e, f, g and h. For every new document , we could easily find out what are the old document it references. Using R_i to represent the set of old documents referenced by new document i, where i=A, B, C, D. We have following:

$R_A = \{a, c, e, f\}$

$R_B = \{b, c, d, f\}$

$R_C = \{b, c, d, f, h\}$

$R_D = \{b, c, e, g\}$

Then we use S_j, where j=a, b, c, d, e, f, g, h to represent the set of new documents reference old document j. We then have following:

$S_a = \{A\}$

$S_b = \{B, C, D\}$

$S_c = \{A, B, C, D\}$

$S_d = \{B, C\}$

$S_e = \{A, D\}$

$S_f = \{A, B, C\}$

$S_g = \{D\}$

$S_h = \{C\}$

New Document   **Old Document**
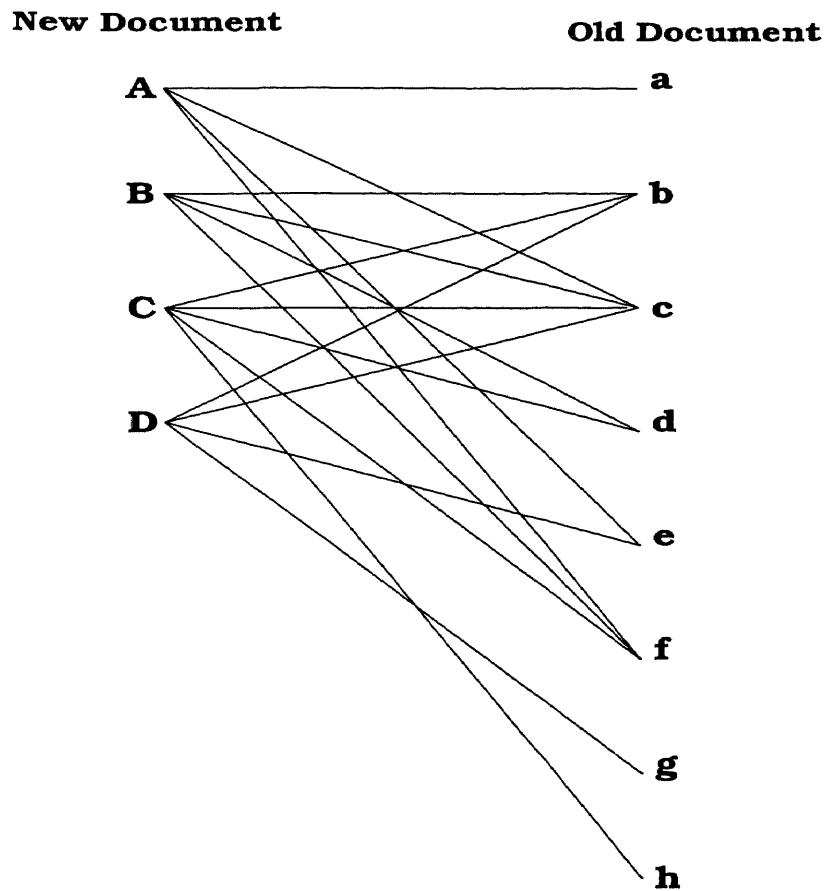

Figure 4.3 Example of Calculating Co-citation

```
int Calculating_Cocitation_Frequency( document a, document b)

begin
    for each new document i
        create arrary Ri to represent the set of old documents it references;

    for each Ri
        while Ri is not empty do
            begin
                if document j in Ri is not in Bucket
                    insert j in Bucket;
                    append document i in the link list of j;
                else
                    append document i in the link list of j;

            end

    compare the link list of document a and document b to create a new
    link list, which is the intersection of two link list;

    return the number of elements of the new link list;

end
```

Figure 4.4 Pseudo-code of the Algorithm

We use CC_{pq} to represent the co-citation frequency between two old documents p and q, where p, q=a, b, c, d, e, f, g and h, noted that $CC_{pq} = CC_{qp}$. We could have following examples:

$$CC_{ab} = |\ S_a \cap S_b\ | = |\ \{\}\ | = 0$$

$$CC_{ac} = |\ S_a \cap S_c\ | = |\ \{A\}\ | = 1$$

$$CC_{bc} = |\ S_b \cap S_c\ | = |\ \{B, C, D\}\ | = 3$$

$$CC_{cd} = |\ S_c \cap S_d\ | = |\ \{B, C\}\ | = 2$$

In Figure 4.4, the algorithm of calculating co-citation frequency is provided in pseudo-code, and in Figure 4.5, shows the running procedure of the algorithm.

To be strongly co-cited, a large number of authors must cite the two earlier works. Therefore Co-citation is a relationship established by the citing authors. In

(1) After processing the 1st element of R_A :
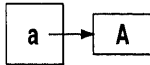
(2) After processing R_A :
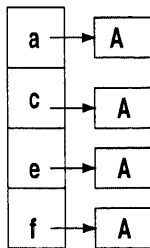
(3) After processing R_A and R_B :
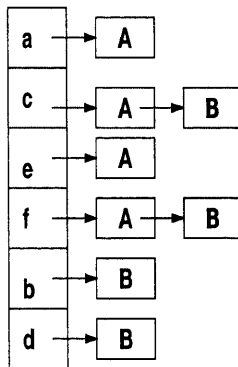
(4) After processing R_A , R_B , R_C and R_D:
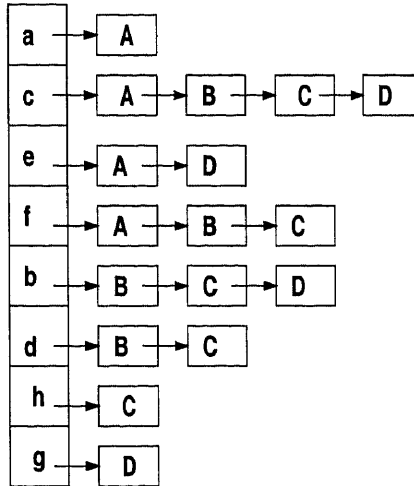
☐ : Bucket   ☐ : Element of Link List

Figure 4.5 Running Procedure of the Algorithm

measuring co-citation strength, we measure the degree of relationship or association between papers as perceived by the population of citing authors. Furthermore, because of this dependence on the citing authors, these patterns can change over time, just as vocabulary co-occurrences can change as subject fields evolve. Bibliographic coupling, on the other hand, is a fixed and permanent relationship because it depends on references contained in the coupled documents. Co-citation patterns change as the interests and intellectual patterns of the field change.

When two papers are frequently co-cited, they are also necessarily frequently cited individually as well. It can be shown [15] that frequently cited papers represent the key concepts, methods, or experiments in a field. Then co-citation patterns can be used to map out in great detail the relationships between these key ideas. This may lead to an objective way of modeling the intellectual structure of scientific specialties. Changes in the co-citation patterns when viewed over a period of years may provide clues to understanding the mechanisms of specialty development. On the other hand, it also provides guidance to newcomers in the field, such as helping to select a group of initial articles.

An illustration of co-citation pattern which is shown in Figure 4.6 Each box in the network diagram represents a cited paper in a specialty of hypermedia information retrieval. The full bibliographic citation for each of the articles in the network is given in Table 4.1. The number of undirected lines which connect the papers reflects the strength of co-citation coupling. The network was drawn from data given in Figure 4.7, omitting co-citation linkages which fall below the threshold of seven. In this figure, each paper is indicated by the first author's last name and the year the paper was published. There are three entries for each pair of papers. the top entry in each box is the co-citation frequency, the second entry is the bibliographic coupling strength, and the third one is a "Yes" if the later paper cited the earlier one, and "No" if it did not.

It should be emphasized that while these ten papers may not represent all the frequently cited papers in this field, they probably constitute the core of that literature. They are the common pool of papers from which the people working in such a field draw their citations. A new paper is added to this pool if the number of its citations is greater than a threshold value in the corresponding bibliography.

Table 4.1 Core papers in the network

J.Conklin, "Hypertext: An introduction and survey," *IEEE Computer*, vol.20, no.9, pp.17–41, 1987.

P.Garg, "Abstraction mechanisms in hypertext," *Communications of the ACM*, vol.31, no.7, pp.862–870, 1988.

F.Halasz, "Reflections on notecards: Seven issues for the next generation of hypermedia systems," *Communications of ACM*, vol.31, no.7, pp.836–852, 1988.

R.Furuta and B.Schneiderman, "Automatically transforming regularly structured linear documents into hypertext," *Electronic Publishing*, vol.4, no.2, pp.211–229, 1989.

J. Waterworth and M. Chignell, "A model for information exploration", *Hypermedia*, vol.2 , no. 4, pp. 35-58, 1990

R.Rana, "Converting a textbook to hypertext," *ACM Transactions on Information Systems*, vol.10, no.3, pp.294–315, 1992.

W.Croft and H.Turtle, "Retrieval strategies for hypertext," *Information Processing & Management*, vol.29, no.3, pp.313–324, 1993.

A, Ginige, D. Lowe and J. Robertson, "Hypermedia Authoring", *IEEE Multimedia*, vol. 30, no. 7, pp. 24-35, Janurary 1995

M.Agosti, F.Crestani, and M.Melucci, " Automatic authoring and construction of hypermedia for information retrieval", *Multimedia System*, vol 7, no. 3. pp. 15-24, 1995

D. Lucarella and A. Zanzi, "A visual retrieval environment for hypermedia information systems", *ACM Transaction on Information Systems*, vol. 14, no. 1, pp. 3-29, 1996

From Figure 4.7, we see the strong connection between co-citation and direct citation patterns. Of the 11 cases of strong co-citation(15 or more), eight are direct citation connections.

Co-citation could be used to establish a cluster or core of earlier literature for a particular field. This core should serve as a profile for that field. Co-citation could also provide a tool for monitoring the development of scientific fields. Because it by-passes the need of classifying all the past literature by automatically constructing bibliographies of papers from the most relevant portion of the past literature.

Single words (and even phrases to a lesser degree) may not have much power to represent the topic of or concepts discussed in a research paper. Citations of other works on the other hand, are hand picked by the authors as being related documents. It seems reasonable [19] to use citation information to judge the relatedness of documents. HyTEXPROS uses common citations [33, 45] to make an estimate of which documents in the downloaded database of research papers are the most closely related to a document picked by the user. This measure, "Common Citation x Inverse Document Frequency" (CCIDF) is analogous to the word oriented TFIDF [63] word weights. The algorithm to calculate the CCIDF relatedness of all documents in the database to a document of interest A and choose the best M documents is:

- Assign a weight w_i to each citation i, equal to the inverse of the frequency of the citation in the entire database.

- Determine the list of citations and their associated weights for document A and query the database to find the set of n documents $B_j : j = 1...n$ which share at least one citation with A.

- For each $j = 1...n$, determine the relatedness of the document R_j as the sum of the weights of the citations shared with A.
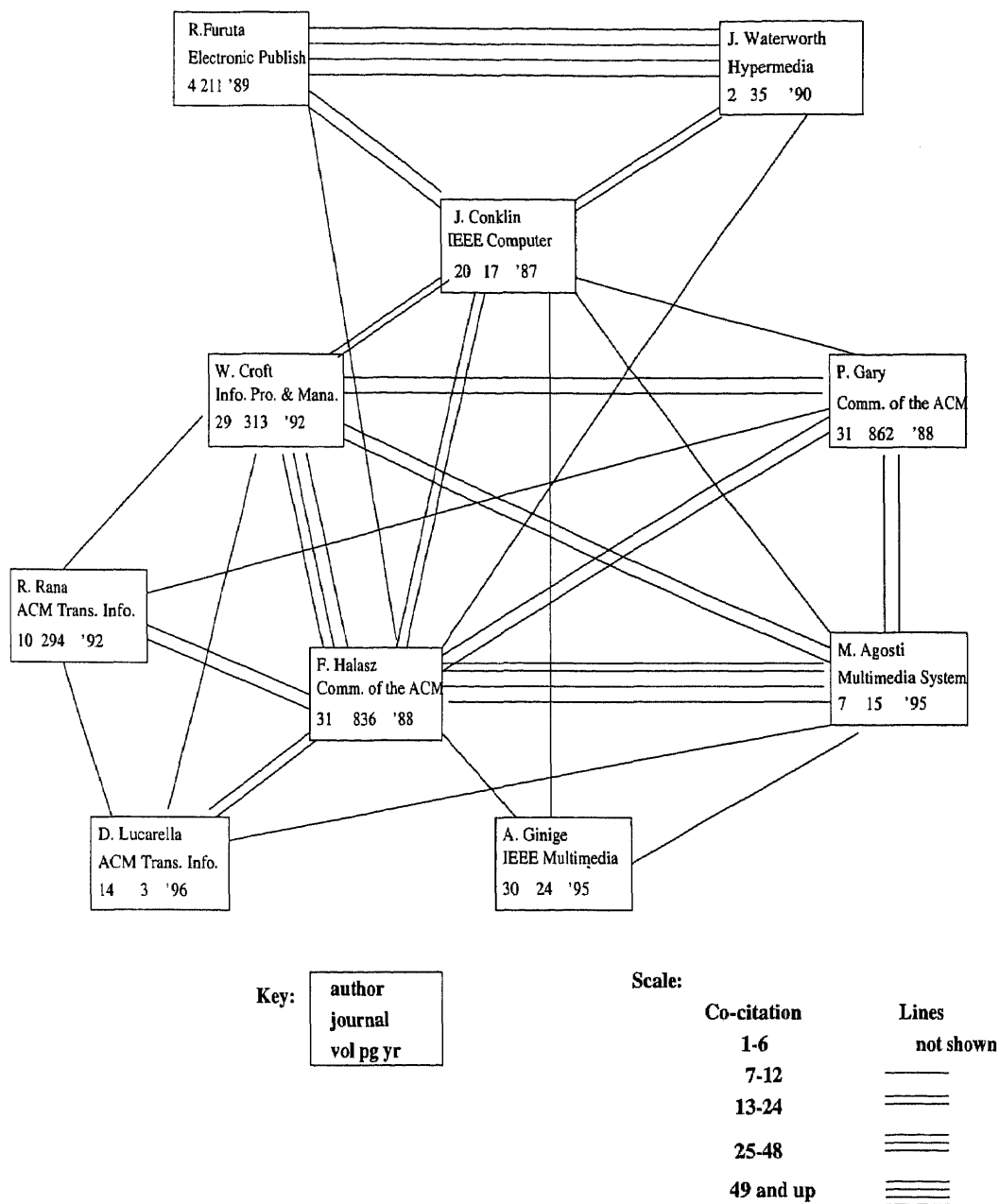
Figure 4.6 Co-citation Network for Frequently Cited Papers in HIR

	Conklin '89	Gary '88	Halasz '88	Furuta '89	Water-Worth '90	Rana '92	Croft '93	Ginige '95	Agosti '95	Lucarella '96
Conklin '89		7 1 Yes	13 3 No	15 4 Yes	22 3 No	2 17 No	21 4 Yes	2 0 No	0 0 Yes	0 0 No
Gary '88			15 3 No	1 0 No	0 0 No	9 20 Yes	16 6 No	2 0 No	17 2 No	0 0 No
Halasz '88				8 12 Yes	10 15 No	20 4 Yes	31 2 No	9 18 Yes	55 1 Yes	21 3 Yes
Furuta '89					56 23 Yes	1 0 No	0 0 No	0 0 No	0 0 No	0 0 No
Water-Worth '90						2 0 No	0 0 No	1 0 No	0 0 No	1 0 No
Rana '92							8 3 Yes	2 0 No	0 0 No	10 7 Yes
Croft '93								0 0 No	20 12 No	9 4 Yes
Ginige '95									11 4 Yes	2 0 No
Agosti '95										12 7 Yes
Lucarella '96										

Figure 4.7 Co-citation, bibliographic coupling and direct citation

$$R_j = \sum_{i \in A \cap i \in B_j} w_i$$

- Sort the R_j values and return the documents B_j with the M highest R_j values.

As in the use of TFIDF, CCIDF assumes that if a very uncommon citation is shared by two documents, this should be weighted more highly than a citation made by a large number of documents. We have found CCIDF is very useful, and it performs better than the word vector document retrieval method.

CHAPTER 5
HYTEXPROS COMPONENTS

Hypertext is the concept of interrelating information elements (linking pieces of information) and using these links to access related pieces of information. (An information element or node can range from a single idea or chunk to an entire document.) A hypertext is a collection or web of interrelated or linked nodes. A hypertext system allows an author to create the nodes and the links among them, and allows a reader to traverse these links, i.e., to navigate from one node to another using these links. Typically hypertext systems mark link access points or link anchors in some manner within a node when displaying it on a computer screen (e.g., underlined text displayed within documents on World Wide Web browsers). When the user selects the link marker, e.g., by clicking on it with a mouse cursor, the hypertext system traverses to and displays the node at the other end of the link. If a single link marker represents multiple links, the hypertext system may present the user with a list of available links. System designers may have to rank, filter or layer this list if the number of possible links might overwhelm the reader. Hypertext user interface design principles recommend that authors label the link marker if the links purpose or destination is not clear. Hypertext systems include many navigation, annotation and structural features, which take advantage of the node and link structure to support authors and readers.

Hypertext, when well designed, enables people to read, author and comprehend information more effectively than traditional documents. People typically read documents from start to end, i.e., in a linear, sequential manner. Paper constrains authors to compose information in a linear format. By tradition as well as from the need to print, many write computerized documents in a linear format. Hypertext frees readers and authors from this linear, sequential forms of expression. Authors can structure information as a web of information chunks and interrelating links.

For example, authors could place their main idea or an overview in an entry-point chunk (node) with multiple links connecting logical next steps or related tangential information chunks. Presenting information as a web enables readers to access information in the order most appropriate to their purposes. Hypertext links and structural features enable authors to provide a rich context of related information around elements. For readers, freedom of access within a web enhanced with contextual information provides a richer environment for understanding the information they find. Many believe that hypertext also enhances comprehension because it mimics the associative networks that people use cognitively to store and retrieve information.

Large numbers of links can lead to the well-known problems of cognitive overhead and disorientation. Cognitive overhead occurs when readers feel overwhelmed with too many choices of where to navigate next. Disorientation occurs when readers lose track of their position within the hypertext web while traversing links. Both can be avoided by using good user interface design principles; semantic node and link types; filtering based on user task and preferences; and hypertext navigation, annotation and structural features. The hypertext community strongly believes in the reader as author. All readers should be able to add their own annotation and structural features, as well as additional links to all nodes in any hypertext web.

In the following sections, we give detail explanation on how to apply hypertext functionality to TEXPROS component by component.

5.1　Nodes

HyTEXPROS classifies nodes into three subclasses: plain atomic nodes, structured atomic nodes and composite nodes.

5.1.1   Atomic Nodes

An atomic node in HyTEXPROS cannot embed other nodes in its content. An atomic node could be structured or unstructured. A *plain* atomic node is an unstructured atomic node which has content without an internal structure. In a multimedia environment, typical examples of plain atomic node include a page of text , a picture, a raster of image, an audio tape, an animated sequence. The content of a plain atomic node is primitive. Currently we only consider text atomic node in HyTEXPROS but we believe it can be extended to include other kinds of data resources.

The content of a *structured* atomic node comprises a sequence of *attribute-value* pairs. We model nodes with only attributes(e.g. a database table schema which is a sequence of field names) as *NONE-value* structured atomic node by specifying *NONE* in their values. For structured atomic node, we can define certain structure-based operations, such as linking to defined on an attribute or value. Complicated node content structures, such as " table of content" or a database table, can be represented as a composite node with internal structures.

5.1.2   Composite Nodes

The concept of composite node greatly improves the organization of HyTEXPROS. Composites provide a more powerful way to construct a hypertext network over the pure lower-level node-link model and assist both user and authors at various levels. During navigation, for instance, with composite nodes the user can *zoom* into a particular node for details or *zoom* out to navigate along the overview structure of a composite node.

A composite node is constructed from other nodes. Individual nodes embedded in a composite node could be any type of node themselves, including composite nodes, plain atomic nodes, and structured atomic nodes.

HyTEXPROS explicitly classifies composite nodes based on the representation of their internal structured as *Set, List, Tree* and *Graph*. A *Set* consists of a set of nodes and no explicit links exist among these nodes. A *List* is composed of an ordered set of nodes connected linearly. A *Tree* is constructed from a set of nodes connected as a tree-like structure and has a distinguished node as its root. A *Graph* has nodes as "vertices" and links as "edges".

One purpose of modeling composite structures is to build multiple views from a composite node based on its internal structure.

5.2   Links

Links represent relationships among nodes. As with nodes, links may have semantic link types (e.g., "supporting evidence for" or "criticism of") and associated keywords. Many hypertext systems display semantic link types as labels near the links marker in nodes, and within maps and overviews. This can orient users by showing how nodes are interrelated, and can help them decide whether to traverse particular links. Users also may specify link types and keywords within structural search queries for nodes with specific relationships.

Links can have link behaviors attached. Traversing them executes an associated action. For example, traversing a "display code" link to a computer program would list its code, whereas traversing an "execute" link would run the program. Hypertext webs also can represent processes to take advantage of hypertext features, representing process steps with nodes and transitions as links. Traversing these "transition" links enacts the process steps.

Many hypertext systems treat links as first class objects in their own right, i.e., giving links identifiers and attributes, and storing them in external linkbases or link databases so they can be reasoned about. Systems keeping links in external linkbases often include an anchor identifier with no link information in anchors. When the

user selects a link marker, the system looks up its anchors corresponding link, often using an independent link service. External linkbases facilitate bi-directional linking (traversing a link from all endpoints), structural search, link evaluation (e.g., checking that all link endpoints exist), and maintaining links over documents for which one does not have access permissions to embed an anchor in the departure node. In the latter case the system embeds link anchors in node content as it passes the node to a browser for display. Links between nodes not only provide the basis for semantic relationships and knowledge representation, but also provide the primary device for control and navigation of a hypermedia-based intelligent system. In this sense, they are the glue that holds the system together, and their definition and implementation within a system is critical for success.

Besides common entity properties, a HyTEXPROS link has property *LinkType* representing six link categories. A link consists of a sequence of link specifies which specify the link endpoints. This helps to direct access explicit and implicit relationships among underlying HyTEXPROS entities.

5.2.1 Typed Link

Typed links provide an easier and clearer mechanism for both the readers and authors to understand a hypertext information network. Link typing enhances the power of two navigation tactics: *filtering* and *zooming*. Filtering occurs when the user is presented by the system with a subset of links which can be followed. With untyped links, however, the user could be overwhelmed by the cognitive overhead of dealing with the whole set of links outgoing from nodes. Filtering on link types restrict his or her navigation to link types of interest while disabling others. Links in HyTEXPROS have a property *LinkType*, representing six categories of links based on different roles they are playing.

$LinkType = "StructureLink" \mid "ReferenceLink"$

$\mid "Annotationlink" \mid "AssociationLink"$

$\mid "NavigationLink" \mid "OperationLink"$

Structure links represent the underlying structure inter-object relationships. In a well-organized information system, among the various types of inter-object relationships, there might be distinguishable relationships which dominate the overall information organization and can be represented as structure links. For example, in TEXPROS, the folder organization can be mapped as structure links which allow direct access from a parent node to a child node and vice versa.

Reference links depict cross-reference relationships among nodes, which can be generated automatically by the system according to predefined inference. The link is that from one user to another user with the same research interest.

Annotation links connect nodes to their annotations. An annotation is a commentary document attached to a node. We separate a node from its annotation by placing the annotation in a separate atomic node and connecting it to the node through an annotation link. Unlike structure and reference link, they are static links created manually.

Association links are user-declared *ad hoc* links representing semantic relationships among nodes. Users can add such links to or delete them from a hypertext network at will. Association links are non-automatable (otherwise they would be reference links). Instead, they are defined manually based on semantic explanation in the user's mind which is not interpretable to the system. The user can define any links among nodes and give them semantic labels. In HyTEXPROS, an association link could be a link from a specified user to a special research group.

Navigation links are system-generated links for navigation purposes. Such links are used to construct navigation structures (e.g. guided-tours). Navigation links do not reflect inter-object relationships. They are dynamic links and generated

automatically by the system according to the user's navigation requests. Navigation links are transparent to users. Users might have no knowledge about the existence of these links. Another example is that in HyTEXPROS, we give different user groups different navigation paths composed of navigation links.

Operation links model operational commands and queries over HyTEXPROS. They are dynamic links defined by inference rules. An operation conducted on a node can be modeled as an operation link from the node pointing to the operation results (which might be generated as destination nodes). Operations invoked from an interface menu item can be modeled as an operation link with no departing node.

5.3   Navigation Structure

The associative nature of a hypertext network structure enables hypertext users to manage and access data stored in a hypertext database with great flexibility. It is this flexibility, however, that frequently causes users cognitive overhead and disorientation during navigation over the hypertext network. This classic hypertext navigation problem, "user disorientation", has been identified and discussed extensively in hypertext literature [37]. Arbitrary linking even has been compared to the abuse of GOTOs in non-structured programming.

Browsing describes the activity of navigating around HyTEXPROS, moving from node to node via the links or with the aid of a graph-type structure. Typically, the reader will read as much or as little of each node as necessary, and on the basis of what is read will decide what link to follow from that node. Browsing is divided into two categories: (1) purposeful browsing; (2) aimless browsing. Purposeful browsing describes the task of browsing when there is some specific piece of information in mind that is being searched. An example of systems that are made primarily to support this type of browsing are help systems, in which the user browses through the hypertext structure in search for the answer to some problem. Aimless browsing

means that the user moves from node to node via the links between nodes, choosing direction on the basis only of what takes their fancy, and without any specific goal in mind. However, there may still be a higher-level goal to this sort of browsing, for example the user may be trying to get an idea of what sort of things are held in HyTEXPROS, or they may be trying to establish how the information is arranged in HyTEXPROS.

Cognitive load is the effort the user has to face to understand and learn the "cognitive model" that has been used in designing the hypertext application and the hypertext system itself. The information base managed by the hypertext system will be structured during the authoring process with a specific cognitive model in mind and if this model is not made clear to the user, the cognitive load of the user can be too large and the user can become overloaded. The ability of the user to use hypertext is directly related to his knowledge of the hypertext structure being used. The simplicity and consistency of the hypertext structure help in reducing the load on the user and also in reducing the learning time necessary to reach a sufficient level of knowledge of the hypertext.

Disorientation is a user's feeling when he or she does not fully understand what are the navigation and browsing facilities made available by the hypertext system; in this case the user is "lost" in the hypertext and does not know in what way to make use of the navigation and browsing features. Most hypertext systems have browsing aids to assist the user in navigation; all systems have a backtrack capability to assist the user in the backward reconstruction of the exploration path constructed using the hypertext.

Efforts have been made to alleviate the disorientation associated with hypertext's nonrestrictive linking and direct user-access features. Navigation via graphical maps and overviews has been proved a useful tool in many hypertext systems such as Intermedia[87], gIBIS[14], NoteCards [38], PlaneText and Neptune[20]. Query-based

filtered browsers, history list, bookmarks and Web view are also helpful mechanism towards disorientation reduction. Navigation via guided-tours[36], combined with other techniques, reduces both disorientation and user cognitive overhead.

Benefiting from the experience of other hypertext researchers, HyTEXPROS provides a comprehensive level of navigation structures including bookmarks, network overviews and guided-tours. One major contribution of HyTEXPROS on navigation modeling is the introduction of the four guided-tour categories(query-based guided-tours, default guided-tours, user-defined guided-tours and navigation-based guided-tours) which are not found in any other hypertext literature. This section focus on the representation of navigation structures regarding bookmarks, network overviews and guided-tours.

5.3.1 Bookmarks

Some nodes in HyTEXPROS may be of special importance to the user. It is helpful to provide a direct access to these nodes from any navigation position. These nodes are called *bookmarks*. Navigation links are maintained by the system to allow direct access to bookmarks. Bookmarks are special nodes in HyTEXPROS which are directly accessible from all other nodes. HyTEXPROS models bookmarks as a *Set* composite node with an index link pointing to it. This index link is a unary link of type "NavigationLink" with only an endpoint directed as "TO" indicating this is a node accessible from all nodes(usually through a menu bar item). Users are allowed to manipulate(add or delete a bookmark) the bookmark *Set*. The content of this *Set* is a set of node specifications.

5.3.2 Navigation Overview

Users often get lost when exploring hypertext networks. A network overview(or simply called an *overview*) is a vision of a substructure of a hypertext network. Overviews help to alleviate the network disorientation by giving the user a sense of

context. HyTEXPROS models overviews on composite nodes. An overview node is constructed as a virtual node based on the node's internal structures, which could be a Set, List, Tree or Graph, depending on the complexity of the original entities. Global overview diagrams provide an overall picture; local overviews provide a fine-grained picture of nodes local neighborhood. Both provide spatial context and reduce disorientation. Readers can traverse directly to a node by selecting its icon.

5.3.3 Guided-Tour

From a control point of view, navigation over a hypertext network can be user-controlled or system-controlled. In a user-controlled navigation, all paths are determined by the user through navigation commands provided by the system. In system-controlled navigation, the navigation paths are prepared by the system following some user input commands. By default, the user is not allowed to use navigation commands to choose his or her own paths once getting on a system-controlled navigation path. One typical example of system-controlled navigation is a *guided-tour*(GT) which is a navigation structure built from a sequence of nodes as a linear path. When navigating on a guided-tour, the user must follow the guided-tour to access information. No branch links are available unless an explicit request is applicable to overwrite the prepared paths. The user can get on or off a guided-tour from any navigation pattern. At any stop of a guided-tour, the user is allowed to invoke other links by pausing the tour and returning later.

Guided-tours are navigational aids that provide a set path through a hypermedia information space that is additional to the usual built-in linking mechanism. A tour will partially take over the navigational function from the user, in order to display a pre-defined set of information. Guided-tours may be developed to adapt the information to individual user characteristics. Different users have different information needs, different backgrounds and different information gathering styles. A different

path may be developed for each of these different types of users. A lot of other hypertext applications use guided-tour. *Banker* generated guided-tours of software components matching specific criteria, which programmers then browse for reuse [6]. Isakowitz discussed automatically generating guided-tours as part of the RM design methodology [44]. In [32], the authors discussed many sophisticated forms of guided tour navigation.

HyTEXPROS models a guided-tour as a *List* composite node consisting of a sequence of nodes and a set of links [80].

HyTEXPROS classifies guided-tour into four categories: default guided-tour(DGTs), query-based guided-tour (QGTs), user-defined guided-tour(UGTs) and navigation-based guided-tours(NGTs).

- **Default Guided-Tours (DGTs)**: DGTS are derived from the structural information of a composite node. They are created automatically by the system and made directly available to the user. A DGT of a composite node is a *List* over links of type NavigationLink automatically derived from structure links of the composite node. One way to obtain a DGT from a composite is to expand the breadth-first search tree on the original structure links level by level and order the resulting nodes in a linear manner.

- **Query-based Guided-Tours (QGTs)**: QGTs are created by the system representing query results.

- **User-defined Guided-Tours (UGTs)**: The user is able to define a UGT on a set of nodes in the same way as defining *ad hoc* association links. In this case, the resulted links would be *ad hoc* navigation links which group participating nodes into a *List* composite node as a UGT. Once defined, a UGT can be invoked arbitrary times until deleted by the user. The user can manipulate

a UGT(e.g. annotating, deleting or adding new nodes, etc.) as a normal composite node.

- **Navigation-based Guided-Tours(NGTs)**: The user can define an NGT based on his or her individual navigation history stored in the history list. The user can select events from the *history list* to construct an NGT. Once constructed, a NGT exists in the linkbase until the user deletes it explicitly. As with UGTs, the user can also manipulate NGTs at run time.

The navigation structures presented in this section help to reduce user disorientation and provide the user a flexible, comprehensive and well structured mechanism to customize individual navigation environment over a hypertext network.

5.3.4   Backtracking

Backtracking serves four purposes: to return to a prior position in a web (which allows users to safely take "detours" from their main task), to review the content of a previously visited node, to recover from a link chosen in error, and as part of undoing. Backtracking differs from undoing, however, in that backtracking returns the reader to a previously visited node in its current state. Parametric backtracking, written as "go-back(X)", allows the user to specify a value for a node attribute in parameter X, which causes the system to backtrack to the most recently departed node with that attribute value. Conditional backtracking, written as "go-back(query-expression)" evaluates an arbitrary query-expression and returns the reader to the most recently departed node satisfying it. Displaying a history list of previously visited nodes, and then selecting one of these backjumps the user to that node. When implementing the history list as a stack, backjumping removes all nodes in-between from the history list. When implementing the history list as a session log, backjumping adds the prior node to the list (though some systems then remove it from its prior position in the log). Authors can construct guided tours and trails directly from session logs.

The user could be interested in the possibility to come back to a previous situation generated during the user's interaction with the environment. Depending on the level and on the specific function the user has just used, the backtracking function implements different ways of returning to the previous situation or possibility to navigate through the previously selected set of terms or documents.

5.4   Annotation

Annotation is a fundamental aspect of HyTEXPROS. In theory, HyTEXPROS grows and changes by way of addition. Readers respond to HyTEXPROS with commentary, make new connections and create new pathways, gather and interpret materials. In doing so, they crucially augment an existing body of interrelated materials. As mention in [53] readers are not only as navigators but also as annotators. Annotation enables a user to add to the HyTEXPROS system. It is the linking of a new commentary node to an existing one. If readers can annotate nodes, then they can immediately provide feedback if the information is misleading, out of date or plain wrong. Thus the quality of the information in the HyTEXPROS can be improved. It is the essence of a collaborate hypermedia. An annotation does not modify the next necessarily. One can separate protection against writing and annotation.

Annotation could be constructed in many ways: as link making (which is used in HyTEXPROS), as path building, as commentary, as marking in or around existing text, as a decentering of authority, as a record of reading and interpretation. They raise some questions concern about annotations: The form annotations take (Are they formal or informal?); the functions of annotation from a reader's point of view (the degree to which the reader has become a writer, the kind of reading that the reader is engaged in, and the permanence of the marks); and the role of annotations as they are used to communicate with others (Are they published or private? who is

the audience?) On the summary, annotation function should be established to benefit other readers without changing the mission or practices of the original annotators.

5.5 History

The history function keeps details on the history of the user-system interaction during an interactive session. Since one of the major usability problems connected with tools that permit navigation in an information space is the risk of disorientation, the history function has been designed to be used in case the user loses the sense of context. Another useful aspect of this function is to keep details of all steps of the interaction, so the user can read through the interaction history to review the different interaction steps and to formulate new interaction strategies for future usage of the tool.

5.6 External Linkbase

In the hypermedia literature, the notion of linkbase has often appeared. Links are stored outside of the documents they refer to [42], in one or more databases of links that are queried by the browser just before showing the document. This architectural choice is extremely flexible and powerful, and many currently unavailable functionalities become possible. For instance, this enables different link sets over the same content for different audiences and tasks. Third party individuals can provide links and create guided tours over documents owned by others. Individuals and groups could maintain their own personal linkbases.

The advantage of separating links from document contents is the following:

- Linking write-protected documents. Because of security concern, users can not create or edit links in documents to which they have no write access. With external links, users can interlink those write-protected documents.

- Linking read-only media. With external links, users can interlink documents stored in read-only media like CD-ROMs or online collection.

- Reduce maintenance requirements. Changes to link information can be made to linkbases instead of documents and will be valid wherever links are available.

- Support structured-based search. Suppose that employees of a department are working on a project. An employee can retrieve all links that some other employee created through a database query.

- Enhance collaborative authoring. Team members can create links to the same documents synchronously.

HyTEXPROS imposes embedded linking which prevents a user from creating links in a document for which he or she has no right to access. That is the idea of separating links from the document contents. Users can store their links in public or private link base.

CHAPTER 6

DESCRIPTION OF HYTEXPROS

Information embedded in HyTEXPROS should be described in terms of a set of pre-defined generic objects [31], called *domain objects*, such as nodes and buttons. All the actual objects are called *information objects*, such as folders and frame instance in HyTEXPROS. An information object is referred as an instance of a domain object and inherits the properties of the domain object. In the remaining section, we shall formalize the HyTEXPROS.

6.1   Basic Definitions and Preliminaries

An *object* is a general concept of primitive type that describes a concrete thing such as a folder in a folder organization. Objects may be referenced by object identifiers, such as the use of frame instance identifiers in the browsing subsystem.

Definition 6.1.1 (Set) A set of objects (also referred as **Set** object) has the following operations:

Let X, Y be **Set** objects, and A be one of the elements belonging to a **Set** object.

- The predicate SET(X) holds iff X is a **Set** object.

- The predicate MEMBERSHIP(A, X) holds iff A is one of the elements belong to X.

- The predicate EQUALITY(X, Y) holds iff X and Y have the same elements.

- The predicate SUBSET$_{OF}$(X, Y) holds iff every element that belongs to X also belongs to Y.

- The union operator '$\cup$' on two **Set** objects yields a new **Set** object, whose elements are all the elements of the two original **Set** objects.

- The intersection operator '∩' on two **Set** objects yields a new **Set** object, whose elements are the common elements of the two original **Set** objects.

- The difference operator '- 'on two **Set** objects yields a new **Set** object, whose elements are in the first original **Set** object but not in the second original **Set** object.

- The function INSERT(A,X) updates X to X∪{A} by adding A to X, if A is not an element of X originally; otherwise the function doesn't change X.

- The function DELETE(A,X) on X, updating X to X-{A}, if A is an element of X originally; otherwise the function doesn't change X.

- The **power set** of X is a new **Set** object, whose elements are all the subsets of X.

Definition 6.1.2 (Sequence) An ordered n-tuple (n is bigger than or equal to 2) over n objects is defined as:

$$\langle A_1, A_2, ..., A_n \rangle = \langle \langle ... \langle \langle A_1 \rangle, A_2 \rangle, ...\rangle, A_{n-1} \rangle, A_n \rangle.$$

When n equals to 2, the special sequence can also be called **Ordered Pair**.

Let X, Y be **Sequence** objects, A be an object which belongs to a **Sequence** object:

- The predicate SEQUENCE(X) holds iff X is a **Sequence** object.

- The predicate MEMBERSHIP(A, X) holds iff A is one of the objects which belongs to X.

- The predicate SUBSEQUENCE$_{OF}$(X, Y) holds iff eliminating zero or more objects from Y yields the set X.

- The function LENGTH(X) yields the number of objects in X.

- The function CATENATION(X, Y) yields a new **Sequence** object consisting of the objects of X followed by the objects of Y.

6.2 Definition of HyTEXPROS

Definition 6.2.1 (Predicate) The set π of predicates consists of mutually exclusive subsets π_1, π_2 and π_3, which are defined as follows:

- π_1 is a set of one-argument predicates (such as SET(X)).

- π_2 is a set of two-argument predicates to assert a certain relationship between two objects (such as IS-A(X, Y)).

- π_3 is one of the following two 3-argument predicates:

    1. **PROPERTY**(X, Y, Z) holds iff Z is a value of the property Y of object X.

    2. **LINKED**(X, Y, Z) holds iff there is a link type Z which goes from object X to another object Y. Z is also referred to as a link identifier.

Definition 6.2.2 (Hypertext) A hypertext system η satisfies the following axiom: $\mathbf{D_0}$ is a set of the domain objects of η. $\mathbf{I_0}$ is a set of information objects. Objects containing information: text, graphics, sounds *etc.*.

$$(\forall X)\ (X \in I_0 \Rightarrow (\exists Y \in D_0)\ (INSTANCE_{OF}(X) = Y))$$

This captures the notion that all information objects have their domain objects defined. Here, I_0 and D_0 could be functions of time t if it is necessary.

Definition 6.2.3 (Information-chunk) An information chunk is an ordered triple of the form $\langle$ position, set-of-special-symbols, content $\rangle$ (also denoted as $\langle P, S, C \rangle$). The position is an integer assigned once and uniquely by the system. The set-of-special-symbols consists of some reserved symbols which denote whether this information chunk is a **keyword**, or the beginning or end (or both) of a **button**. The content is actual information data contained in this information chunk.

Definition 6.2.4 (Keyword) is an instance of an information chunk, containing the special symbol in its set-of-special-symbols field.

Definition 6.2.5 (Button) A button is defined as a pair: ⟨ Node-id, Button-Value ⟩ (also denoted as ⟨ N, B ⟩), where Node-id is the object identifier of a node and button-value is a set of consecutive information chunks. The information chunk with the smallest integer in the position field contains the special symbol → in its set-of-special-symbols field, denoting the beginning of the button, while the end of the button is denoted with the special symbol ← in the set-of-special-symbols field of the chunk with the highest position value. If the two symbols coexist , the information-chunk represents the button.

Button Axiom: Every Button belongs to a specific node:

$(\forall X,N,B)$ $(X \in I_0 \wedge \text{BUTTON}(X) \wedge ((\text{LIST}(X) = \langle N, B \rangle)$

$\Rightarrow \text{NODE}(N) \wedge (N \in I_0) \wedge \exists Y(\text{PROPERTY}(N, \text{INFO},Y) \wedge \text{SUBSET}_{OF}(B, Y)))$

Definition 6.2.6 (Node) is an information-object(the actual information container of hypertext), which has a special property INFO, whose value is a set consisting of consecutive information chunks.

Definition 6.2.7 (Link) Links are to be considered as objects. We define a domain object of typed link LINK(X) is a one-place predicate which holds iff X is an object of typed link.

The establishment of a link from the user creates such an object and asserts an instance of the predicate: **LINKED**(X, Y, Lid), where X and Y are nodes or buttons, Lid is the object identifier of the link between X and Y.

By this definition, we allow links between nodes, and links between arbitrarily small or large information units (down to the size of a word).

Link Axiom L1: Introduce links of various kinds, let I_0 be a set of information objects, X, Y, Z$\in I_0$

$\forall X,Y,Z(\text{LINKED}(X,Y,Z) \Rightarrow$

$$(((NODE(X) \land NODE(Y) \land LINK(Z)) \lor$$

$$(NODE(X) \land BUTTON(Y) \land LINK(Z)) \lor$$

$$(BUTTON(X) \land NODE(Y) \land LINK(Z)))$$

Link Axiom L2: No two different links stem from the same anchor point. Let I_0 be a set of information objects, and X, Y1, Y2, Z1, Z2 $\in I_0$

$$(\forall X,Y1,Y2,Z1,Z2) \; (LINKED(X,Y1,Z1) \land LINKED(X,Y2,Z2) \Rightarrow ((Y1{=}Y2) \land$$

$$(Z1{=}Z2))$$

CHAPTER 7

IMPLEMENTAION: THE HYTEXPROS PROTOTYPE

In this chapter, we present a description of an implementation prototype to prove the correctness and robustness of the HyTEXPROS. We discuss the implementation architecture and its individual components. We decided to design HyTEXPROS ensuring an object-oriented approach. This object oriented approach, if used for design purposes, could support the direct representation of two essential data abstraction: classification and generalization/specialization. We use the Java language (JDK v1.1.6) [25, 57] to code the prototype. The prototype runs on the Sun Solaris platform, but it can also run on Microsoft Windows 95 and Windows NT.

7.1 JAVA

Java is an object-oriented programming language developed by Sun Microsystems. It is an interpreted language with built in multi-threading, dynamic loading, and automatic garbage collection. Java is currently being positioned as a language to develop small applications called applets that can be transported across the Internet through the use of an Internet browser. The Java AWT is a user-interface toolkit bundled with the Java development environment. It is a cross-platform user-interface toolkit for Java applets providing access to a standard suite of user-interface widgets.

We use Java[40] as a programming tool. Because run-time library provides platform independence, you can use the same code on Solaris, Windows 95, UNIX, Macintosh and so on. Everything in Java is an object. The key points about Java are that:

- Java is Object-Oriented. A Java program is made up of classes and objects. Each class describes a kind of object and provides operations on those objects. Classes form an inheritance hierarchy.

- Java is simple. It is based on C and C++. Many problematic features of C++ removed such pointers , multiple inheritance and templates. Garbage collection simplifies programming.

- Java is robust. It is a strongly typed language. It has many runtime checking such as array bounds and null pointers. Java Should never dump core. Garbage collection means no memory allocation errors. Exception handling encourages robust programming.

- Java is secure. There is no way to forge pointers. Java code is verified. Access to the file system and network can be restricted.

- Java is portable. Size and representation of primitive data types is defined by the language. A standard library is also specified. Bytecodes are easy to interpret or compile on almost any cpus and operating systems.

- Java is distributed. Java classes can be transmitted over the net. Library classes for network I/O are provides. Remote method invocation and CORBA are compatible.

- Java is Multithreaded. Thread and synchronization primitives are built into the language. Threads are common and relatively easy to use.

- Java is free. Everyone could download the Java Development Kit (JDK) from http://www.javasoft.com. JDK includes compiler, interpreter, JIT compiler, applet viewer, debugger, Java libraries, library source code and so on.

7.2 Guided-Tour Component

The Guided-Tour Frame is composed of four parts. The first one is a frame container. It has all the basic functionality of a frame. The second one is menu bar. There are three menu items "File", "Edit", and "Help". The third one is a button group. There are five buttons: "START", "NEXT", "PREVIOUS", "JUMP", and "RESET". Last one is a panel container with a label in it, it indicates what users should do in the next step. In Figure 7.1 shows the class hierarchy of this part.

The Guided-Tour frame, like other frame containers, can be dragged arbitrarily or to be resized by the user with the mouse. It also has a system menu button, with which the user can do operations such as minimizing, maximizing, refreshing, closing the window. Using the attached menus, users can create new tabletop cards and links or open a default guided tour.

The menu bar consists of three menu items: "File", "Edit", and "Help". All the editing work can be done with it. Users can make use of "File" to open guided-tour or exit from the current window. The user can create new tabletop cards and links, delete tabletop cards and links, or modify a card name and comments. Finally, the user can get information about the component with "Help".

To operate or "Run" a guided tour, the users can make use of the five buttons arranged at the bottom of the guided tour frame. The operations are invoked by these buttons. In each operation, the current node is automatically closed down before bringing up the text.

In Figure 7.2 and Figure 7.3, are shown the screen dumps from the implementation of the prototype. Figure 7.2 shows the default path, and Figure 7.3 shows the user-defined path.
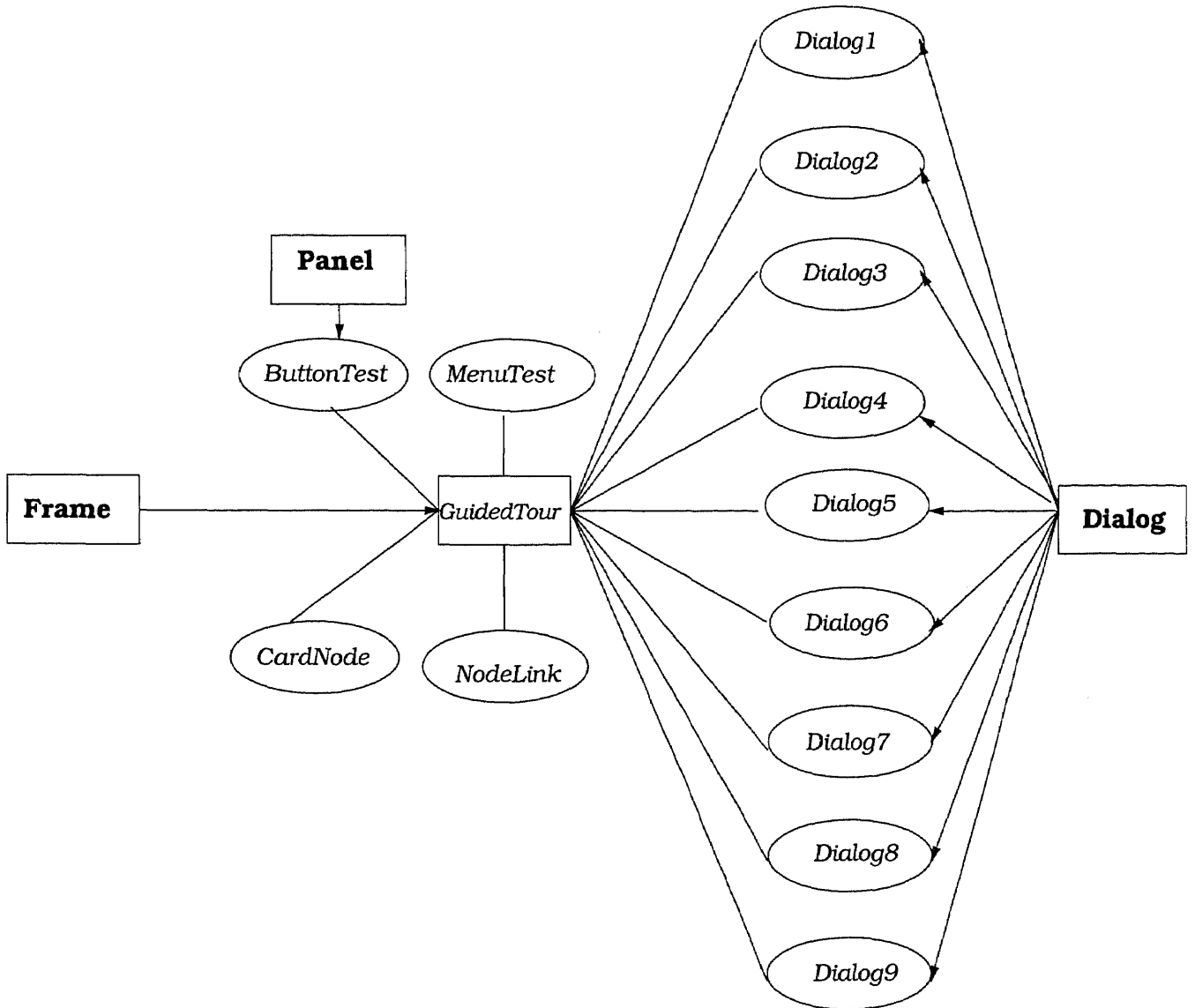
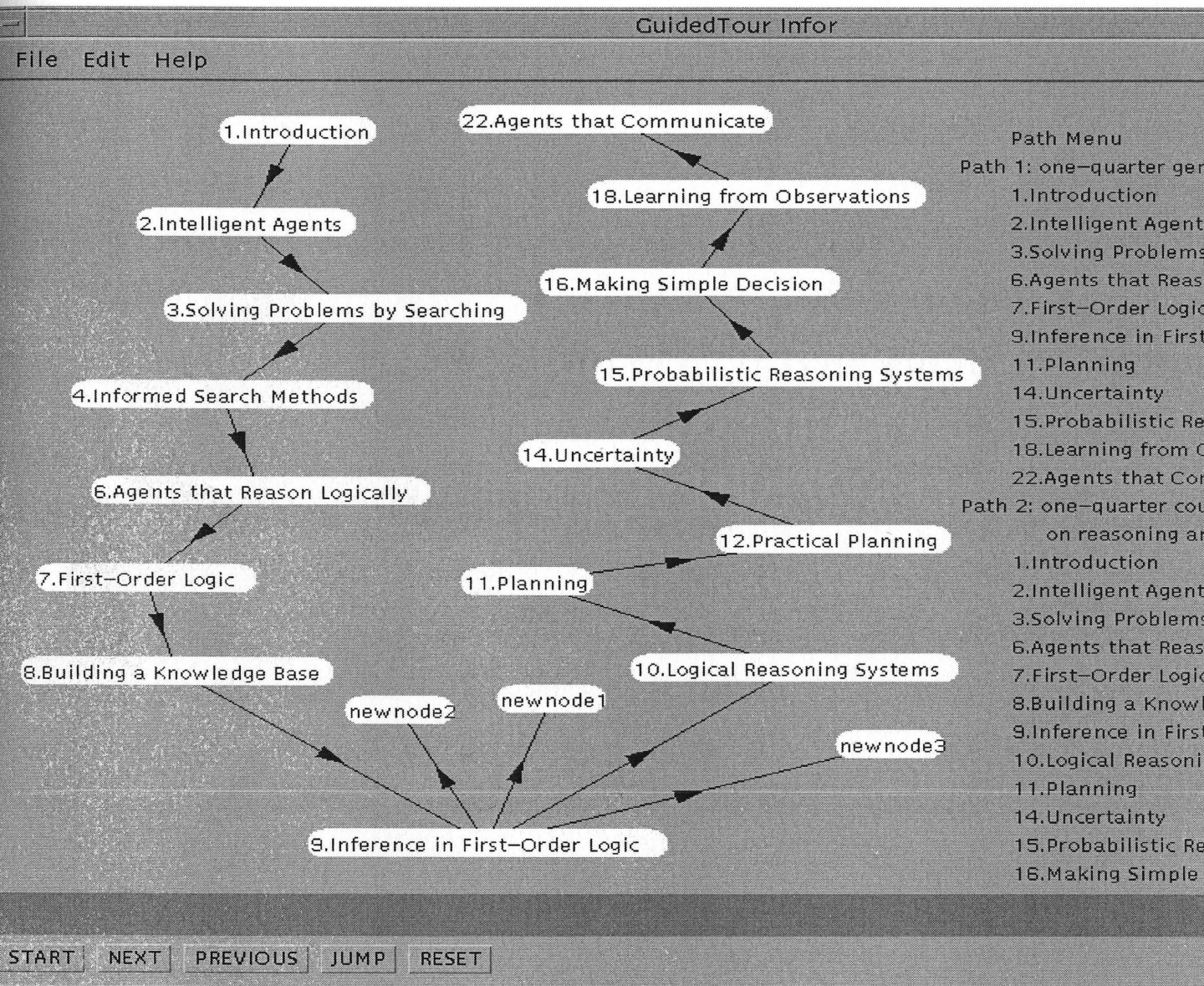**Figure 7.1** Class Hierarchy of Guided-Tour Component

Figure 7.2 Guided-Tour Component Implementation with Default Path
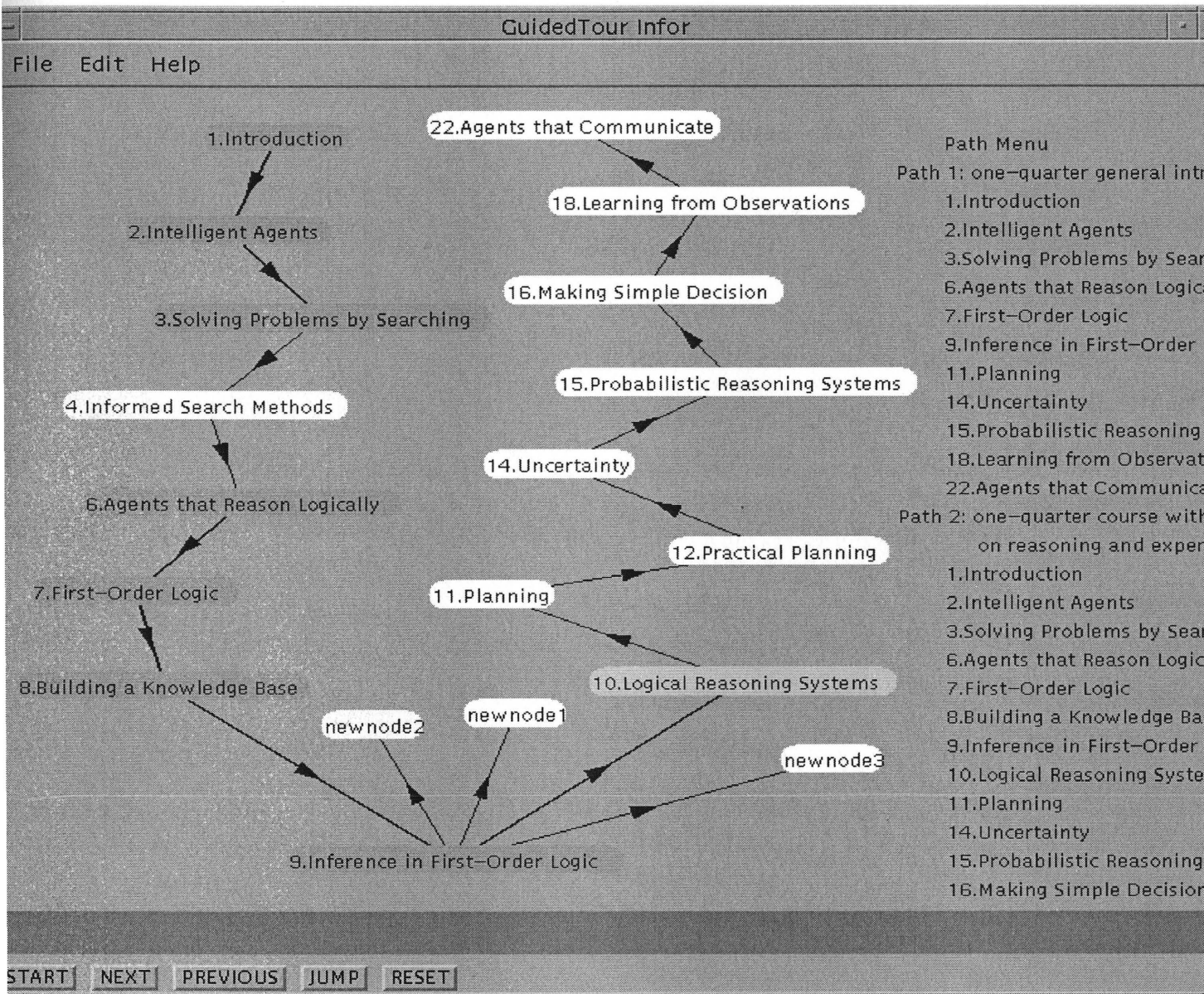
Figure 7.3 Guided-Tour Component Implementation with User-defined Path

7.3   Conceptual Architecture

This part uses some inherent packages of JAVA language, such as java.awt, java.util, and java.io. It provides a convenient and friendly graphical user interface. There are 23 classes in this part. They are CommentFrame, EdgeHelpFrame, EdgeHelpPanel, EdgeInfoFrame, ExitFrame, ExitDialog, NodeHelpFrame, NodeHelpPanel, NodeInfoFrame, SameLevelFrame, SameLevelPanel, ThreeLevelFrame, ThreeLevelPanel, TwoLevelFrame, TwolevelPanel, MainFrame, DrawPanel, and NodeMenu.

The main window composes by four parts. First is a window frame with a caption bar. It has all basic functionality of a window frame. The caption bar shows the user "Main Frame". Second is a menu bar. There are total five menu groups and more menu items. Most of the work is accomplished via these menu items. The third is a label that shows the users how to perform an activity and how they should do it. The fourth is a panel that displays the graph. It also provides some facilities to user. There are four rectangles in the panel, which contain four types of nodes: document, entity, template and concept.

The menu groups are shown in Figure 7.4

The main GUI classes includes MainFrame, drawPanel, and NodeMenu. The menu window screen dump are shown in Figure 7.5, Figure 7.6, Figure 7.7.

7.4   Evaluation of HyTEXPROS

Information retrieval evaluation techniques like computing precision and recall are not directly usable in the evaluation of characteristics of hypertext information retrieval systems like HyTEXPROS. It is therefore necessary to develop new procedures and tools that establish a relationship between present evaluation efforts to previous evaluation work in information retrieval, and at the same time be able to effectively evaluate new system capabilities.

MENU	SUBMENU
Document	New, Open, Close, Save, Save As, Print, Exit
Edit	Reset, Demo
View	Reference, Citation
Node	Add, ShowInfo
Link	Add, ShowInfo
Help	About HyTEXPROS, Help Topics

Figure 7.4 Menu Groups

Dunlop in [21, 22] proposed a model tested by carrying out two experiments that use a text document collection to relate the results to previous findings in the information retrieval area. Some insights into the development of evaluation techniques for hypermedia systems are given together with some more general results from a combination of query-based and browsing-based retrieval capabilities. Croft and Turtle in [18] also dealt with the problem of evaluating HIR systems. A comparison of performance of the strategies used in two retrieval models was made; a probabilistic retrieval model incorporating inter-document links with strategies that ignore the links versus a heuristic spreading activation strategy [12, 16, 26].
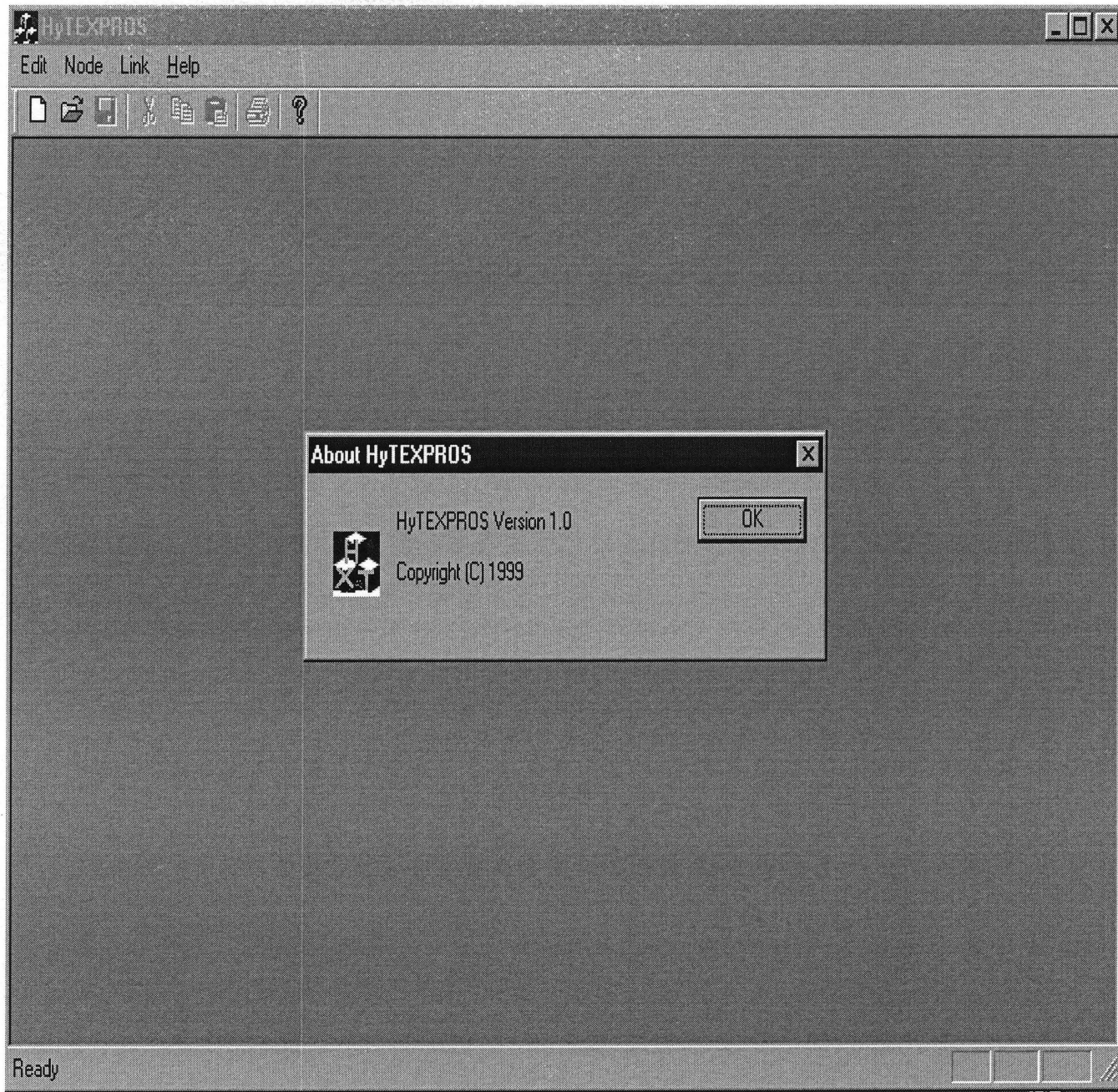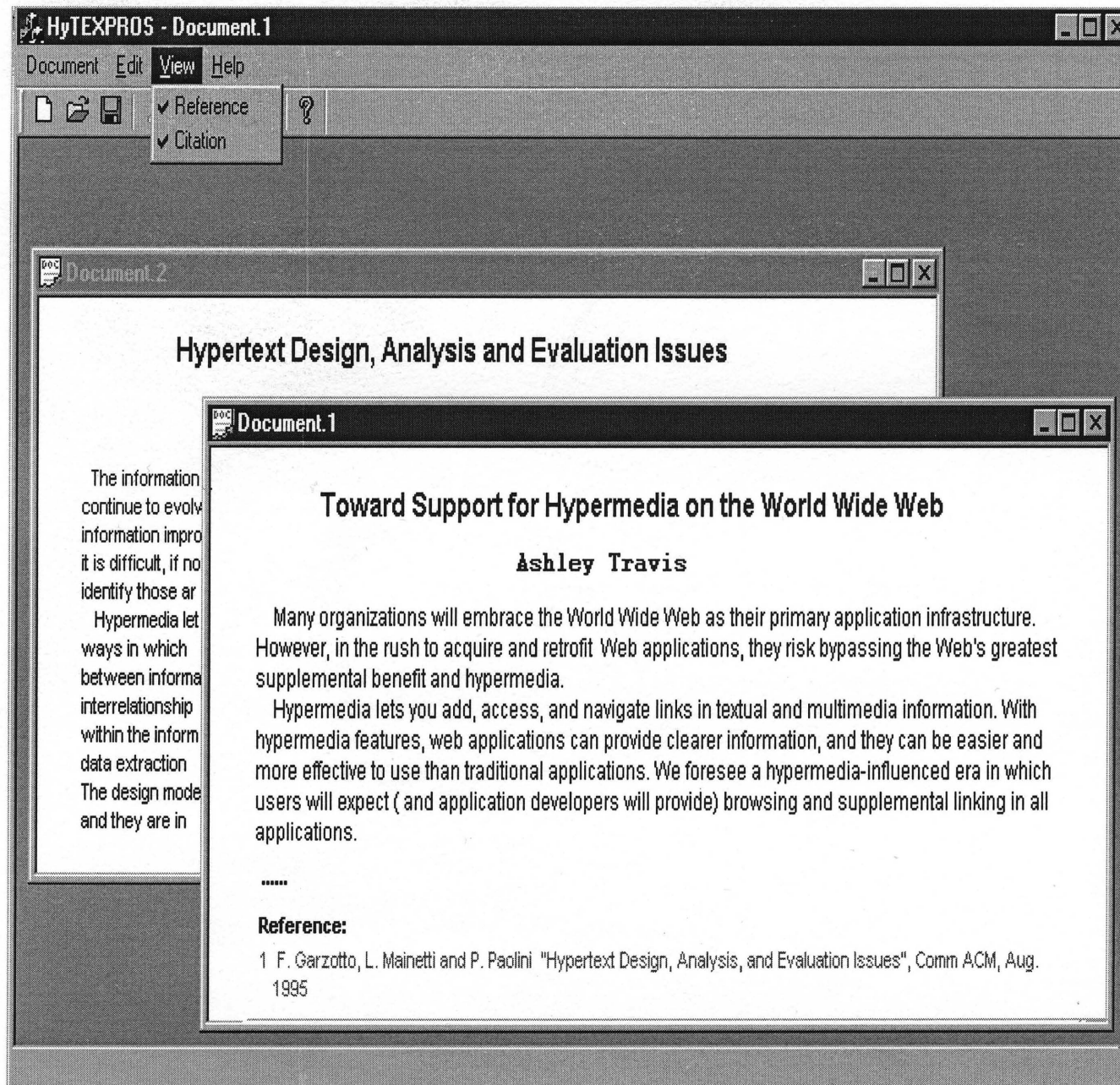
Figure 7.5 HyTEXPROS About Window
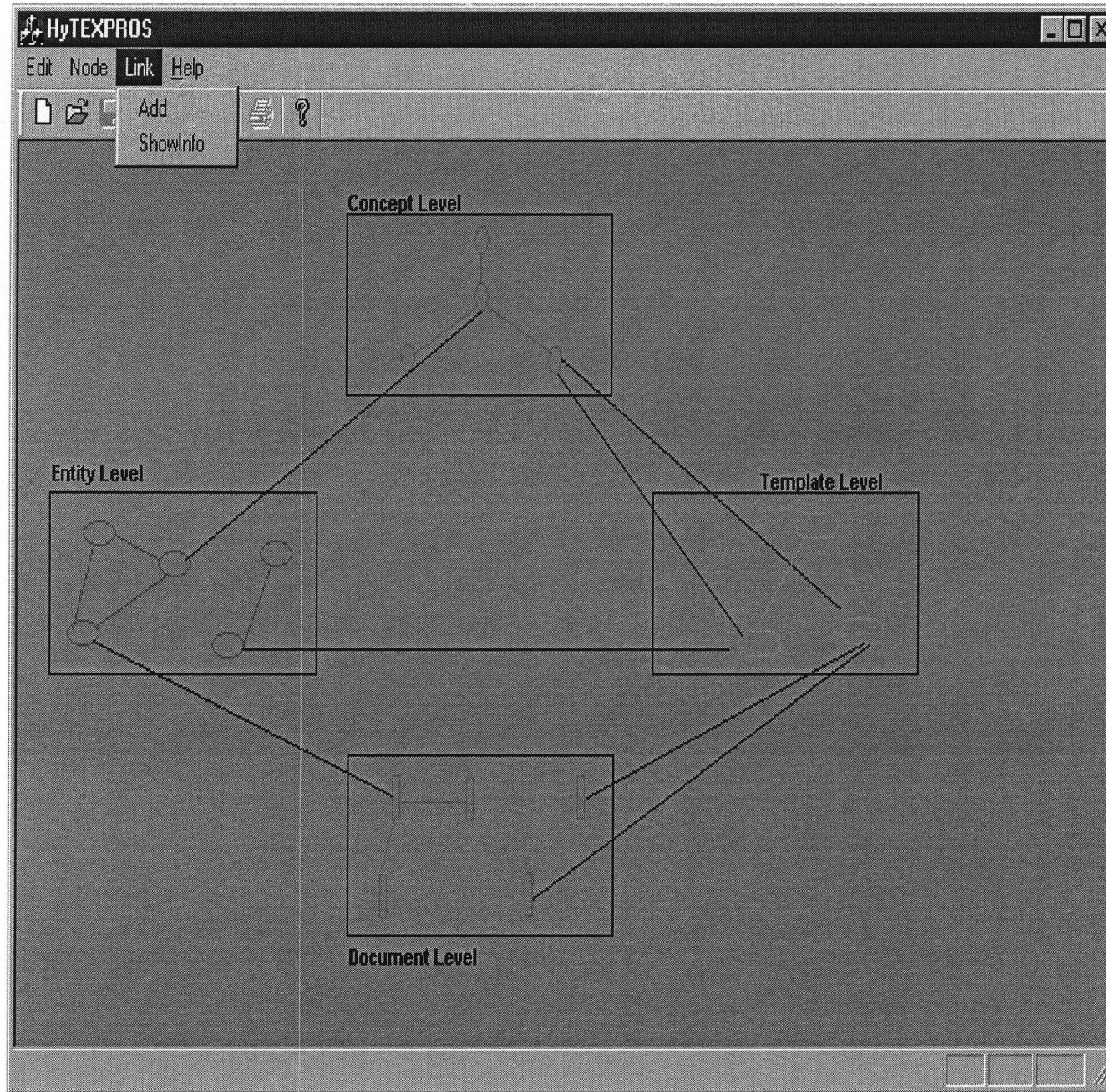
Figure 7.6 Document1 References Document2

Figure 7.7 Implementation of Conceptual Architecture

CHAPTER 8

CONCLUSION AND FUTURE WORK

In this chapter, we compare HyTEXPROS with other systems and models, identified both HyTEXPROS's major contributions and limitations, and briefly outlined future research that could emanate from HyTEXPROS.

8.1   Comparison with Other Systems and Models

HyTEXPROS and its prototype share ideas and common constructs with other systems developed by hypertext researchers, especially in the field of hypertext information retrieval.

We compare HyTEXPROS with other hypermedia information retrieval systems including EXPLICIT, BRAQUE, TACHIR. In Figure 8.1 , it shows the difference between these four systems on the aspects of typed links, composite nodes, annotation and external database. HyTEXPROS is better than any of other systems because it has all the components and all the other systems only have a few of the components.

8.2   HyTEXPROS Contributions and Limitations

Hypermedia information retrieval system makes use of the specific capabilities of hypermedia systems together with information retrieval operations and provides a new kind of information management tools. It combines both hypermedia and information retrieval together to offer end-users the possibility of navigating, browsing and searching a large collection of documents to satisfy an information need. TEXPROS is a perfect application to apply hypermedia information retrieval techniques.

Components / Systems	Typed Links	Composite Nodes	Annotation	External Database	Guided Tour
HyTEXPROS	YES	YES	YES	YES	YES
EXPLICIT	NO	NO	YES	YES	NO
BRAQUE	YES	NO	NO	YES	YES
TACHIR	NO	YES	YES	NO	NO

Figure 8.1 The Comparison of Different Systems

HyTEXPROS aims at enhancing TEXPROS by adding hypertext functionalities. such as node, typed and weighted links, anchors, guided-tours, network overview, bookmarks, annotations, and external linkbase.

We view HyTEXPROS's major contributions from the following five aspects: (1) It describes the whole information base including the metadata and the original documents as network nodes connected by links. Through hypertext functionalities, a user can construct dynamically an information path by browsing through pieces of the information base. By adding hypertext functionalities to TEXPROS, HyTEXPROS is created. (2) It changes the working domain of the system from a personal document process domain to a personal library domain accompanied with citation techniques to process original documents. (3) A four-layer conceptual architecture is presented as the system architecture of HyTEXPROS, which retains the Document Type Hierarchy and Folder Organization properly. Such an architecture is also referred to as the reference model of HyTEXPROS. (4) A detailed description of HyTEXPROS,

using the First Order Logic Calculus, is proposed. (5) A prototype of HyTEXPROS is constructed and tested. The HyTEXPROS prototype successfully embeds hypertext functionalities into the original TEXPROS system and it proves the correctness of the architecture and model of HyTEXPROS.

8.3 Future Research

HyTEXPROS is a robust hypertext information retrieval system. Extensions in several directions can be made to enhance the current version of HyTEXPROS. This section outlines the future work we plan to pursue.

In the conceptual architecture of HyTEXPROS, links between concept only have the relationship is-a, we have not include AND-link and OR-link here in the concept level. Within the entity level, the direct relationship between entities are not defined, we need to define them in the future work.

We plan to continue implementing those features defined in HyTEXPROS but not included in the prototype, such as an external database.

On the other hand, multi-user access represents future research work because it is usually realized through a distributed network of computers. In recent years, delivering electronic information via computer networks has had significant growth. There are numerous hypertext systems operating in a distributed environment. [56].

There are existing hypertext models and systems supporting multimedia[10, 50]. Examples of multimedia components include a CAD picture, a raster image, a short video clip, a short audio tape, a short animated sequence. Currently HyTEXPROS only considers text components. The component framework in HyTEXPROS can be extended to support multimedia. We could use structured composite nodes to model multiple types of media.

HYTEXPROS will also consider more ideas from digital libraries research field, such as : In the university of Illinois at Urbana-Champaign, the Digital Library

Initiative [8] project provides digital library and knowledge desegregation, which means a scientific journal article is comprised of standard components, such as author names, an abstract, figures, a bibliography, and sections describing methods and results, and these components can be identified mobilized, and used by students and faculty members based on the preliminary analysis of data.

8.4 Conclusion Remarks

TEXPROS is an intelligent document processing and retrieval system that supports storing, extracting, classifying, categorizing, retrieval and browsing enterprise information. TEXPROS is a perfect application to apply hypermedia information retrieval techniques.

In this dissertation, we extend TEXPROS to a hypermedia information retrieval system called HyTEXPROS with hypertext functionalities, such as node, typed and weighted links, anchors, guided-tours, network overview, bookmarks, annotations and comments, and external linkbase. It describes the whole information base including the metadata and the original documents as network nodes connected by links. Through hypertext functionalities, a user can construct dynamically an information path by browsing through pieces of the information base. By adding hypertext functionalities to TEXPROS, HyTEXPROS is created. It changes its working domain from a personal document process domain to a personal library domain accompanied with citation techniques to process original documents.

A four-level conceptual architecture is presented as the system architecture of HyTEXPROS. Such architecture is also referred to as the reference model of HyTEXPROS. Detailed description of HyTEXPROS, using the First Order Logic Calculus, is also developed. An early version of a prototype is briefly described.

REFERENCES

1. I. Aalbersberg, "Incremental relevance feedback," in *Proceeding of 15th SIGIR Conference*, Copenaghen, Denmark, pp. 11–21, 1992.

2. M. Agosti, F. Crestani, and M. Melucci, "Automatic authoring and construction of hypermedia for information retrieval," *Multimedia System*, vol. 7, no. 3, pp. 15–24, 1995.

3. M. Agosti, F. Crestani, and M. Melucci, "Design and implemention of a tool for the automatic construction of hypertexts for information retrieval," *Information Processing & Management*, vol. 32, no. 4, pp. 459–476, 1996.

4. R. Akscyn, D. McCracken, and E. Yoder, "Abstraction mechanisms in hypertext," *Communications of the ACM*, vol. 31, no. 7, pp. 820–835, 1988.

5. J. Allan, "Relevance feedback with too much data," in *Proceedings of the 18th ACM SIGIR Conference*, Seattle, U.S.A., pp. 337–343, 1995.

6. R. Banker and T. Isakowitz, *Software Engineering Economics and Declining Budgets*, ch. Tools for managing repository objects, Springer-Verlag, Korn, German, 1993.

7. N. Belkin, P. Marchetti, and C. Cool, "Braque: Design of an interface to support user interaction in information retrieval," *Information Processing & Management*, vol. 29, no. 3, pp. 325–344, 1993.

8. A. P. Bishop, "Digital libraries and knowledge disaggregation: the use of journal article components," in *Digital Library 98 Proceeding*, Pittsburgh, PA, pp. 29–39, 1998.

9. P. Bruza and T. van der Weide, "Stratified hypermedia structures for information disclosure," *The Computer Journal*, vol. 35, no. 3, pp. 208–220, 1992.

10. D. Bulterman, "Specifying and support of adaptable networked multimedia," *ACM Multimedia Systems*, vol. 1, no. 2, pp. 68–76, 1993.

11. A. Celentano, M. Fugini, and S. Pozzi, "Knowledge-based document retrieval in office environments: The kabina system," *ACM Transactions on Information Systems*, vol. 13, no. 3, pp. 237–268, 1995.

12. P. Cohen and R. Kjeldsen, "Information retrieval by constrained spreading activation in semantic networks," *Information Processing & Management*, vol. 23, no. 4, pp. 255–268, 1987.

13. J. Conklin, "Hypertext: An introduction and survey," *IEEE Computer*, vol. 20, no. 9, pp. 17–41, 1987.

14. J. Conklin and M. Begeman, "gibis: A tool for all reasons," *Journal of the American Society for Information Science*, vol. 20, no. 1, 1989.

15. S. Cozzens, "What do citation count? the rhetoric-first model," *Scientometrics*, vol. 15, no. 2, pp. 437–447, 1989.

16. W. Croft and H. Turtle, "A retrieval model for incorporating hypertext links," in *Hypertext 89 Proceeding*, Southampton, UK, pp. 15–25, 1989.

17. W. Croft, "Effective text retrieval based on combining evidence from the corpus and users," *IEEE Expert*, vol. 10, no. 6, pp. 59–63, 1995.

18. W. Croft and H. Turtle, "Retrieval strategies for hypertext," *Information Processing & Management*, vol. 29, no. 3, pp. 313–324, 1993.

19. B. Cronin and H. Snyder, "Comparative citation rankings of authors in monographic and journal literature: a study of sociology," *Journal of Documentation*, vol. 53, no. 3, pp. 263–273, 1997.

20. N. Delisle, "Neptune: A hypertext system for cad applications," in *ACM SIGMOD International Conference on Management of Data*, Washington, D.C., pp. 132–143, 1986.

21. M. Dunlop, *Multimedia Information Retrieval*, PhD thesis, Department of Computer Science, University of Glasgow, Glasgow,UK, 1991.

22. M. Dunlop and C. Rijsbergen, "Hypermedia and free text retrieval," *Information Processing & Management*, vol. 29, no. 3, pp. 287–298, 1993.

23. X. Fan, *Knowledge-Based Document Filing for TEXPROS*, PhD thesis, Department of Computer and Information Science, New Jersey Institute of Technology, Newark, 1998.

24. X. Fan, Q. Liu, and P. Ng, "Knowledge-based document filing: Texpros approach," in *Proceedings of the 13th International Conference on Advanced Science and Technology in conjunction with the 2nd international conference on Multimedia Information Systems*, pp. 58–67, April 1997.

25. D. Flanagan, *JAVA in a Nutshell*, O'Reilly, 101 Morris Street, Sebastopol, CA, 1997.

26. M. Frisse and S. Cousins, "Information retrieval from hypertext:update on the dynamic medical handbook project," in *Hypertext 89 Proceeding*, Southampton, UK, pp. 199–212, 1989.

27. R. Furuta and B. Schneiderman, "Automatically transforming regularly structured linear documents into hypertext," *Electronic Publishing*, vol. 4, no. 2, pp. 211–229, 1989.

28. E. Garfield, *Citation Indexing: Its Theory and Application in Science, Technology, and Humanities*, 1979.

29. E. Garfield, "The concept of citation indexing: A unique and innovative tool for navigating the research literature," *Current Contents*, January 03 1994.

30. E. Garfield, "Where was this paper cited ?," *Current Contents*, January 31 1994.

31. P. Garg, "Abstraction mechanisms in hypertext," *Communications of the ACM*, vol. 31, no. 7, pp. 862–870, 1988.

32. F. Garzotto, L. Mainetti, and P. Paolini, "Navigation in hypermedia application," *Journal of Organizational Computing*, vol. 6, no. 3, pp. 373–403, 1995.

33. C. Giles, K. Bollacker, and S. Lawrence, "Citeseer: An automatic citation indexing system," in *Digital Libraries' 98*, Pittsburgh, PA, pp. 89–98, 1998.

34. A. Ginige, D. Lowe, and J. Robertson, "Hypermedia authoring," *IEEE Multimedia*, vol. 30, no. 7, pp. 24–35, Winter 1995.

35. G. Golovchinsky, "What the query told the link: The integration of hypertext and information retrieval," in *Hypertext 97*, Southampton, UK, pp. 33–39, 1997.

36. C. Guinan and A. Smeaton, "Information retrieval from hypertext using dynamically planned guided tours," in *ACM Conference on Hypertext*, Milan, Italy, pp. 43–52, 1992.

37. F. Halasz, "Reflections on notecards: Seven issues for the next generation of hypermedia systems," *Communications of ACM*, vol. 31, no. 7, pp. 836–852, 1988.

38. F. Halasz and R. Trigg, "Notecards in a nutshell," in *1987 ACM Conference of Human Factors in Computer Systems*, Totonto, Ont., pp. 45–52, 1987.

39. X. Hao, *Document Classification and Information Extraction*, PhD thesis, Department of Computer and Information Science, New Jersey Institute of Technology, Newark, 1995.

40. C. Horstmann and G. Cornell, *Core Java Volume 1 and Volume 2*, Sun Microsystems Press, Mountain View, CA, 1997.

41. J. Hu, *Knowledge Management for TEXPROS*, PhD thesis, Department of Computer and Information Science, New Jersey Institute of Technology, Newark, 1999.

42. D. Hugh, "Referential integrity of links in open hypermedia systems," in *Hypertext 98 Proceeding*, Pittsburgh, PA, pp. 207–216, 1998.

43. P. Ingwersen, "Information retrieval interaction," tech. rep., Royal School of Librarianship, Copenhagen, Denmark, 1992.

44. T. Isakowitz and E. Balasubramanian, "Rmm: A methodology of structured hypermedia design," *Communications of the ACM*, vol. 38, no. 8, pp. 34–40, 1995.

45. S. Lawrance, C. L. Giles, and K. Bollacker, "Digital libraries and autonomous citation indexing," *IEEE Computer*, vol. 31, no. 20, pp. 17–41, 1998.

46. X. Li, *Automatic Document Classification and Extraction System(ADoCES)*, PhD thesis, Department of Computer and Information Science, New Jersey Institute of Technology, Newark, 1999.

47. Q. Liu and P. Ng, "A browser of supporting vague query processing in an office document system," *Journal of System Integration*, vol. 5, no. 1, pp. 61–82, 1995.

48. Q. Liu and P. Ng, *Document Processing and Retrieval: Text Processing*, Kluwer Academic Publishers, Norwell, MA, 1996.

49. D. Lucarella and R. Morara, "First: Fuzzy information retrieval system," *Journal of Information Science*, vol. 17, no. 3, pp. 81–91, 1991.

50. D. Lucarella, S. Parisotto, and A. Zanzi, "More: Multimedia object retrieval environment," in *Hypertext 93 Proceeding*, Washington, DC, pp. 39–50, 1993.

51. D. Lucarella and A. Zanzi, "Information retrieval from hypertext: An approach using plausible inference," *Information Processing & Management*, vol. 29, no. 3, pp. 299–312, 1993.

52. D. Lucarella and A. Zanzi, "A visual retrieval environment for hypermedia information systems," *ACM Transaction on Information Systems*, vol. 14, no. 1, pp. 3–29, 1996.

53. C. Marshall, "Toward an ecology of hypertext annotation," in *Hypertext 98 Proceeding*, Pittsburgh, PA, pp. 40–49, 1998.

54. C. Marshall and P. Irish, "Guided tours and on-line presentations: How authors make existing hypertext intelligible for readers," in *Hypertext 89 Proceeding*, Southampton, UK, pp. 15–25, 1989.

55. C. Marshall and C. Shipment, "Spatial hypertext: Designing for change," *Communications of the ACM*, vol. 38, no. 8, pp. 88–95, 1995.

56. J. Nielsen, *Multimedia and Hypertext: The Internet and Beyond*, AP Professional, Mountain View, California, 1995.

57. P. Niemeyer and J. Peck, *Exploring Java, second Edition*, O'Reilly, 101 Morris Street, Sebastopol, CA, 1997.

58. R. Pollard, "A hypertext-based thesaurus as a subject browsing aid for bibliographic databases," *Information Processing & Management*, vol. 29, no. 3, pp. 345–357, 1993.

59. R. Rana, "Converting a textbook to hypertext," *ACM Transactions on Information Systems*, vol. 10, no. 3, pp. 294–315, 1992.

60. G. Salton, "Automatic indexing using bibliographic citations," *Journal of Documentation*, vol. 27, no. 3, pp. 98–110, 1971.

61. G. Salton, *Automatic Text Processing*, Addison-Wesley, Reading, MA, 1989.

62. G. Salton and C. Buckley, "Automatic text structuring experiments," in *Text-based Intelligent Systems: Current Research and Practice in Information Extraction and Retrieval*, Lawrence Erlbaum Associates, Hillsdale, NJ, 1992.

63. G. Salton and C. Buckley, "Term weighting approaches in automatic text retrieval," tech. rep., Tech Report 87-881, Department of Computer Science, Cornell University, Ithaca, New York, 1987.

64. G. Salton and M. McGill, *Introduction to Modern Information Retrieval*, McGraw-Hill, New York, NY, 1983.

65. S. Schiminovich, "Automatic classification and retrieval of documents by means of a bibliographic pattern discovery algorithm," *information Storage Retrieval*, vol. 6, no. 2, pp. 417–435, 1971.

66. N. H. Shen, *HyTEXPROS: A Hypermedia Information Retrieval Systems*, PhD thesis, Department of Computer and Information Science, New Jersey Institute of Technology, Newark, 1999.

67. N. H. Shen, J. Lin, T. Chu, R. Curtis, and P. A. Ng, "Automatic authoring in hytexpros," in *Proceeding on the Fourth World Conference on on Integrated Design and Process Technology (IDPT)*, Kusadasi, Turkey, June 1999.

68. N. H. Shen, P. A. Ng, and Q. Liu, "Hytexpros: A hypermedia information retrieval system," in *Proceeding of Intelligent Information Systems Conference*, Bahama, pp. 399–404, December 1997.

69. B. Shneiderman, *The Society of Text*, MIT Press, Cambridge, MA, 1989.

70. H. Small, "Co-citation in the scientific literature: A new measure of the relationship between two documents," *Journal of the American Society for Information Science*, vol. 29, no. 3, pp. 265–269, 1973.

71. H. Small, "Cited documents as concept symbols," *Social Studies of Science*, vol. 8, no. 3, pp. 327–340, 1978.

72. A. F. Smeaton, "Building hypertext under the influence of topology metrics," in *International Workshop on Hypermedia Design(IWHD)*, Montpellier, France, pp. 102–110, 1992.

73. M. Snoeck and G. Dedence, "Generalization/specialization and role in object-oriented conceptual modelling," *Data and Knowledge Engineering*, vol. 19, no. 2, pp. 171–195, 1996.

74. P. Srinivasdan, *Information Retrieval: data structures and algorithms*, ch. Thesaurus construction, Prentice Hall, Englewood Cliffs, NJ, 1992.

75. N. Streitz, "Design hypermedia: A collaborative activity," *Communications of the ACM*, vol. 38, no. 8, pp. 70–77, 1995.

76. R. Trigg, "Guided tours and tabletops: Tools for communicating in a hypertext environment," *ACM Transaction on Office Information Systems*, vol. 6, no. 4, pp. 398–414, 1988.

77. H. Turtle and W. Croft, "A comparison of text retrieval models," *The Computer Journal*, vol. 35, no. 3, pp. 279–289, 1992.

78. N. Uda, "Personal librarian system in digital library of specific fields," in *International Symposium on Digital Libraries 1995*, Tsukuba, Ibaraki, Japan, pp. 285–286, 1995.

79. C. van Rijsbergen, *Information Retrieval(2nd Ed)*, Butterworths, The United Kingdom, London, 1979.

80. J. Wan, *Integrating Hypertext with Information Systems Through Dynamic Mapping*, PhD thesis, Department of Computer and Information Science, New Jersey Institute of Technology, Newark, 1996.

81. C. Wang, *The Intelligent Browser for TEXPROS*, PhD thesis, Department of Computer and Information Science, New Jersey Institute of Technology, Newark, 1998.

82. C. Wang, Q. Liu, and P. Ng, "Browsing in an information repository," in *Second World Conference on Integrated Design and Process Technology*, Austin, TX, pp. 48–56, 1996.

83. J. Wang and P. Ng, "Texpros: An intelligent document processing system," *International Journal of Software Engineering and knowledge Engineering*, vol. 2, no. 2, pp. 171–196, 1992.

84. J. Waterworth and M. Chignell, "A model for information exploration," *Hypermedia*, vol. 2, no. 4, pp. 35–58, 1990.

85. C. Wei, *knowledge Discovering for Document Classification Using Tree Matching in TEXPROS*, PhD thesis, Department of Computer and Information Science, New Jersey Institute of Technology, Newark, 1996.

86. D. Wolfram, A. Volz, and A. Dimitroff, "The effect of linkages structural on retrieval performance in a hypertext-based bibliographic retrieval system," *Information Processing & Management*, vol. 29, no. 3, pp. 529–541, 1993.

87. N. Yankelovich, B. Haan, N. Meyrowitz, and S. Drucker, "Intermedia: The concept and the construction of a seamless information environment," *IEEE Computer*, vol. 21, no. 1, 1988.

88. Z. Zhu, Q. Liu, J. McHugh, and P. Ng, "A predicate driven document filing system," *Journal of Systems Integration*, vol. 6, no. 3, pp. 373–403, 1996.

89. Z.Zhu, *On Document Filing Based Upon Predicates*, PhD thesis, Department of Computer and Information Science, New Jersey Institute of Technology, Newark, 1997.