

**INFORMATION DISCOVERY IN DISTRIBUTED
SYSTEMS – A NOVEL CONCEPT BASED ON
MODELLING OF INFORMATION ELEMENTS**

A THESIS SUBMITTED BY

SRINIVASA NARASIMHA KINI

IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE AWARD OF THE DEGREE OF

DOCTOR OF PHILOSOPHY

UNDER THE FACULTY OF TECHNOLOGY



DEPARTMENT OF COMPUTER SCIENCE

COCHIN UNIVERSITY OF SCIENCE AND TECHNOLOGY

KOCHI – 682 022, KERALA, INDIA

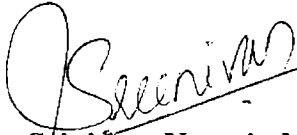
2002

DECLARATION

I hereby declare that the work presented in the thesis entitled “**INFORMATION DISCOVERY IN DISTRIBUTED SYSTEMS – A NOVEL CONCEPT BASED ON MODELLING OF INFORMATION ELEMENTS**”, is based on the original work carried out by me under the supervision and guidance of Dr. K. Poulouse Jacob in the Department of Computer Science, Cochin University of Science and Technology, and that no part thereof has been presented for the award of any other degree.

Kochi – 682 022

31st May, 2002.



Srinivasa Narasimha Kini

CERTIFICATE

This is to certify that the thesis entitled “**INFORMATION DISCOVERY IN DISTRIBUTED SYSTEMS–A NOVEL CONCEPT BASED ON MODELLING OF INFORMATION ELEMENTS**” is a bonafide record of the research work carried out by *Srinivasa Narasimha Kini* under my supervision and guidance in the *Department of Computer Science, Cochin University of Science and Technology* and the result enclosed in this thesis or part thereof has not been presented for the award of any other degree.



Dr. K. Poulouse Jacob

(Supervising Teacher)

Professor and Head

Department of Computer Science

Cochin University of Science and Technology

Kochi – 682 022

31st May, 2003.

ACKNOWLEDGEMENT

I express my profound gratitude to my guide Prof. Dr. K. Poulouse Jacob for his invaluable guidance and support throughout this work.

I am grateful to Prof. Dr. K. Babu Joseph, former Vice Chancellor, Cochin University of Science and Technology for helpful suggestions.

I express my sincere thanks to Dr. (Mrs.) M.D.Baby, Librarian, University Library, Cochin University of Science and Technology and Mr. David Peter S., Reader, Department of Computer Science, Cochin University of Science and Technology for his timely help and inspiration.

I am thankful to Dr. (Mrs.) Sumam Mary Idicula, Reader, Department of Computer Science, Cochin University of Science and Technology for helpful suggestions.

Sincere thanks are due to Prof. Dr. A. Neelameghan, Prof. Dr. G. Bhattacharyya and Prof. Dr. I.K. Ravichandra Rao for all their help and for giving me an opportunity to utilize the facilities at ISI, Bangalore.

I am thankful to all the members of the faculty, laboratory, library and non-teaching staff of the Department of Computer Science for their co-operation and help during the tenure of this research work.

I wish to thank all my family members for their whole-hearted co-operation and encouragement in completing this work. Without their support, inspiration and motivation, I could not have imagined venturing into research.

I am indebted to one and all who have enlightened me, in this crucial period of life.

Srinivasa Narasimha Kini

TABLE OF CONTENTS

TABLE OF CONTENTS	I
TABLE OF SCENARIOS	V
TABLE OF TABLES	VI
TABLE OF LISTINGS	VI
ABSTRACT	VII
1 INTRODUCTION	1
1.1 BACKGROUND	1
1.2 MOTIVATION	3
1.3 SCOPE	5
1.4 OUTLINE OF THE WORK	6
2 INFORMATION DISCOVERY - A REVIEW	8
2.1 INTRODUCTION	8
2.2 DISTRIBUTED SYSTEMS: CHARACTERISTICS AND CHALLENGES	9
2.3 INFORMATION MODELLING	12
2.3.1 TYPES OF INFORMATION MODELS	15
2.3.2 COGNITIVE MODELS	16
2.3.3 CONCEPTUAL MODELS	19
2.3.4 SEMANTIC DATA MODELS	24
2.3.5 OTHER ASPECTS OF MODELLING	28
2.4 INFORMATION RETRIEVAL TECHNIQUES	35
2.4.1 FEATURE-BASED TECHNIQUES	37

2.4.2	STRUCTURE-BASED TECHNIQUES.....	43
2.4.3	NETWORK OF INTERRELATED DOCUMENTS.....	47
2.4.4	FEEDBACK METHODS.....	50
2.4.5	PREDICTING APPROPRIATE RETRIEVAL TECHNIQUES.....	50
2.4.6	ARCHITECTURES AND TECHNIQUES FOR INTEGRATED SYSTEMS	51
2.5	SEARCH ENGINES AND THEIR PROBLEMS.....	52
2.6	CURRENT STATE OF RESEARCH IN THE FIELD.....	54
2.7	SUMMARY.....	61
3	ARCHITECTURE OF SOFTWARE SYSTEMS.....	63
3.1	INTRODUCTION.....	63
3.2	INFORMATION DISCOVERY PARADIGM.....	64
3.3	ARCHITECTURE OF DISTRIBUTED SYSTEMS.....	69
3.3.1	BACKGROUND.....	70
3.3.2	SYSTEM ARCHITECTURE	71
3.4	ECRS SYSTEM ARCHITECTURE	80
3.5	SUMMARY.....	86
4	INFOTRONS AND IDS DESIGN	87
4.1	INTRODUCTION.....	87
4.2	INFORMATION THEORY	88
4.3	INFOTRONS.....	91
4.4	DEFINITION OF INFORMATION.....	92
4.5	INFOTRON DICTIONARY	95
4.6	IDENTIFYING INFOTRONS IN A DOMAIN AND DEVISING AN INFOTRON DICTIONARY.....	97

4.7	AN EMPIRICAL STUDY FOR INCORPORATING INFOTRONS.....	100
4.7.1	EXPERIMENTAL SETUP.....	104
4.7.2	SOFTWARE.....	114
4.7.3	HARDWARE.....	114
4.7.4	RESULTS OF THE EXPERIMENT AND CONCLUSION.....	115
4.8	INFORMATION DISCOVERY SYSTEM.....	121
4.8.1	REQUIREMENTS.....	122
4.8.2	SPECIFICATIONS.....	122
4.8.3	ARCHITECTURE.....	125
4.8.4	IMPLEMENTATION ASPECTS.....	131
4.8.5	PERFORMANCE RESULTS OF THE IDS COMPONENTS.....	131
4.9	INFORMATION DISCOVERY IN LIBRARY INFORMATION SYSTEM (IDLIS).....	133
4.10	SUMMARY.....	139
5	CONCLUSION AND FUTURE DIRECTIONS.....	140
5.1	MAJOR CONTRIBUTIONS OF THE WORK.....	141
5.2	POSSIBLE APPLICATIONS OF IDS.....	143
5.2.1	INTERNET INFORMATION DISCOVERY (IID).....	143
5.2.2	WEB SERVICES DISCOVERY.....	143
5.2.3	FORENSIC SYSTEMS.....	144
5.2.4	MACHINE TRANSLATION AND NATURAL LANGUAGE UNDERSTANDING.....	144
5.3	FUTURE DIRECTIONS.....	145

APPENDIX A	A-1
REFERENCES.....	148
AUTHOR INDEX.....	167
SUBJECT INDEX.....	169
PUBLICATIONS OF THE AUTHOR.....	176

TABLE OF FIGURES

Figure 2-1	An information model continuum.....	16
Figure 2-2	Classification of IR techniques for a single individual document	36
Figure 2-3	Classification of IR techniques for a network of interrelated documents.	37
Figure 3-1	Information Discovery Paradigm.....	65
Figure 3-2	A System Architecture with five complementary technologies	73
Figure 3-3	ECRS System Architecture.	82
Figure 4-1	Schematic diagram of general information discovery system	91
Figure 4-2	Schematic component interaction diagram for the experiment	104
Figure 4-3	A snapshot of the input screen	106
Figure 4-4	Algorithm for analysis of infotrons in HTML documents.....	108
Figure 4-5	Alogorithm for Analysis of infotrons in XML documents	109
Figure 4-6	Chart of time taken for analysis for 25 queries in HTML & XML.	119
Figure 4-7	Architecture of Information Discovery System.....	126
Figure 4-8	IDS components in Penta-tier Architecture.	130
Figure 4-9	Web-based GUI for IDLIS.....	134
Figure 4-10	Results for Q.No 1	137
Figure 4-11	A corresponding XML file got as result.	138
Figure 4-12	Results for Q.No 3 from IDLIS	138

TABLE OF SCENARIOS

Scenario 4-1	Sending a Telegram	89
Scenario 4-2	Drinking tender coconut	90

TABLE OF TABLES

Table 2-1	Comparison of prominent existing search engines on Internet.....	53
Table 3-1	Modules used in monolithic architecture.....	81
Table 3-2	Faults injected, their description and classification.....	83
Table 3-3	Summarised results of the fault tolerance experiment.....	85
Table 4-1	A sample infotron dictionary for Library.....	99
Table 4-2	A table comparing HTML and XML.....	103
Table 4-3	Infotrons and operators furnished in the experiment.....	111
Table 4-4	Hardware platforms used for testing purpose.....	114
Table 4-5	Results of infotron measures for 25 queries in HTML.....	115
Table 4-6	Results of timing measures for the 25 queries in case of HTML.....	116
Table 4-7	Results of infotron measures for 25 queries in XML.....	117
Table 4-8	Results of timing measures for the 25 queries in case of XML.....	118
Table 4-9	Summarized result of the experiment.....	120
Table 4-10	Resource utilization of components of IDS.....	132
Table 4-11	Details of LIS Database.....	134
Table 4-12	Infotrons furnished to IDS system.....	136

TABLE OF LISTINGS

Listing 4-1	A sample partial listing of HTML document containing infotrons.....	101
Listing 4-2	A sample partial listing of XML document containing infotrons.....	102
Listing 4-3	A listing of 25 queries used in the experiment.....	110

INFORMATION DISCOVERY IN DISTRIBUTED SYSTEMS – A NOVEL CONCEPT BASED ON MODELLING OF INFORMATION ELEMENTS

ABSTRACT

Sharing of information with those in need of it has always been an idealistic goal of networked environments. With the proliferation of computer networks, information is so widely distributed among systems, that it is imperative to have well-organized schemes for retrieval and also discovery. This thesis attempts to investigate the problems associated with such schemes and suggests a software architecture, which is aimed towards achieving a meaningful discovery. Usage of information elements as a modelling base for efficient information discovery in distributed systems is demonstrated with the aid of a novel conceptual entity called infotron.

The investigations are focused on distributed systems and their associated problems. The study was directed towards identifying suitable software architecture and incorporating the same in an environment where information growth is phenomenal and a proper mechanism for carrying out information discovery becomes feasible. An empirical study undertaken with the aid of an election database of constituencies distributed geographically, provided the insights required. This is manifested in the Election Counting and Reporting Software (E CRS) System. E CRS system is a software system, which is essentially distributed in nature designed to prepare reports to district

administrators about the election counting process and to generate other miscellaneous statutory reports.

Most of the distributed systems of the nature of ECRS normally will possess a "*fragile architecture*" which would make them amenable to collapse, with the occurrence of minor faults. This is resolved with the help of the penta-tier architecture proposed, that contained five different technologies at different tiers of the architecture.

The results of experiment conducted and its analysis show that such an architecture would help to maintain different components of the software intact in an impermeable manner from any internal or external faults. The architecture thus evolved needed a mechanism to support information processing and discovery. This necessitated the introduction of the novel concept of infotrons. Further, when a computing machine has to perform any meaningful extraction of information, it is guided by what is termed an infotron dictionary.

The other empirical study was to find out which of the two prominent markup languages namely HTML and XML, is best suited for the incorporation of infotrons. A comparative study of 200 documents in HTML and XML was undertaken. The result was in favor of XML.

The concept of infotron and that of infotron dictionary, which were developed, was applied to implement an Information Discovery System (IDS). IDS is essentially, a system, that starts with the infotron(s) supplied as clue(s), and results in brewing the

information required to satisfy the need of the information discoverer by utilizing the documents available at its disposal (as information space). The various components of the system and their interaction follows the penta-tier architectural model and therefore can be considered fault-tolerant. IDS is generic in nature and therefore the characteristics and the specifications were drawn up accordingly. Many subsystems interacted with multiple infotron dictionaries that were maintained in the system.

In order to demonstrate the working of the IDS and to discover the information without modification of a typical Library Information System (LIS), an Information Discovery in Library Information System (IDLIS) application was developed. IDLIS is essentially a wrapper for the LIS, which maintains all the databases of the library. The purpose was to demonstrate that the functionality of a legacy system could be enhanced with the augmentation of IDS leading to information discovery service. IDLIS demonstrates IDS in action. IDLIS proves that any legacy system could be augmented with IDS effectively to provide the additional functionality of information discovery service.

Possible applications of IDS and scope for further research in the field are covered.

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND

Continued miniaturization of microelectronics and higher level of software performance have produced computers with immense power within their hood. Present day microprocessors can churn numbers at an incredible speed and at great accuracy. Their capacity has surpassed teraflops and they have already started operating at Giga Hertz (GHz) clock. Their basic word length has been increased from 32 to 64 bits; apart from using numerous parallel computing components incorporated right at the hardware level. Modern computer systems contain often than not, more than a single processor housed within it. Any reasonable server available on the network today, for instance, is based on at least a dual processor architecture.

The goal of engineers in the software field has been to devise methodologies to develop the software easily, faster, and in a maintainable way throughout the software life cycle and even beyond. Recently, the focus has shifted from just solving the problem once, to think whether the components so developed for one purpose can be reused with little or no modification to a similar problem environment in other domain or an organization with a different environment. This trend is mainly due to the following reasons.

The software components, which have been proved reliable, when reused in a new environment/context increases the level of confidence in the new system developed. This in turn results in high reliability and quality of the deliverable software system. In addition, there is a reduction in the total developmental time. The independent nature of each component makes it suitable to be coupled with the running system dynamically, resulting in high availability, fidelity and utilization of the software system. Other problems often faced are the maintenance of software, and avoiding prohibitive developmental cost, especially so, for large software systems.

Despite being expensive to develop large software system, there is no guarantee that the estimated operational life cycle of the software system is reasonable enough to justify the system development in the first place. This is largely due to the ever evolving, dynamic nature of the environment in which the large software system operates. The dynamic nature of the problem environment calls for such software architectures, which can face the test of time and scale with the change in specification of the software system. A well-developed architectural framework can concoct understanding of the software system plausible to flourish its feasibility and extend its operational life cycle at least to justify development.

As the basic computing technology developed from mechanical to electronics and further from analog to digital, computing power has increased in order of magnitude, with a decrease in cost and size of machines. This transition of the machines has led to

the development of many distributed systems (DSs). DSs make it possible for the computers to find usage in new application that did not exist earlier.

1.2 MOTIVATION

The major challenge faced by humanity today, is the **tremendous growth of information** [1,2]. This is more so in a distributed system (DS). The trend of information getting accumulated in a DS is expected to continue and grow further. This trend will lead to “*eternal information space*” in a DS. For instance, in the universe of Internet – the largest distributed system – finding the required information amounts to an expedition. Without being able to keep in pace with the current growth of information on the Internet, the present search engines are seeking alternative technologies [3,4].

A book by David Shenk [5] titled “Data Smog coping with the information glut” elucidates the problems of information explosion/overload that is invading mankind, as a result of the technological revolution, that took place in the past few decades. As observed by Neil Postman in his review comments [available in Internet at <http://www.amazon.com>] on Shenk’s book, “*Over the past 150 years, humanity solved the problem of information scarcity. In solving it, we created the problem of information glut, incoherence, and meaninglessness*”. Accordingly, humanity appears to get escorted from an information-scarce society to an information-loaded one. Neil Postman himself has written a book in similar vibes [6].

Steve Kirsch the founder of Infoseek search engine has pointed out the way in which the information overload on the Internet is growing day by day. He presented how and at what rate the information content and the users of the Internet are growing worldwide.

Radical growth in Internet,¹ associated with the swift growth of the World Wide Web (WWW), and other technologies (such as Space technology, Geographical Information System (GIS) etc.), forced everyone to generate information in the digital form. The data so generated was in terms of terabytes, as is rightly observed by Sadananda in [7]. The author therein has even made suggestion and pointers to choose from one among the several alternative tools and techniques.

The Reuters started a project [8] to cope up with the “Information explosion”, in relation to the staggering information producing capability they had themselves built, with more than 27,000 pages of reports per second. Their study report suggests, an important need to measure and incorporate sound information management techniques at all levels.

Daniel Drelinger et. al., [9] designed a meta search engine, which they named SavvySearch that was capable of producing results from various other search engines. Meta search engines automate and produce results, which otherwise could have been achieved - albeit independently - using search engines by human beings.

¹ Internet and WWW are two different entities. The former represents the network of computer networks, whereas the latter is a collection of HTML pages with hyperlinks connecting them to form a web within the former.

The issues discussed and the problems mentioned are some of the basic motivating factors for conducting the present research.

1.3 SCOPE

This work is the manifestation of the need to support the needs of information scientists, software architects, knowledge engineers, and the like who constantly endeavor for improvement in the information discovery process. The concept of infotrons was the outcome of investigations towards quantifying and modelling information and information systems, which is expected to get refined in due course. It is hoped that infotron would ultimately represent the basic unit of information storage at its barest form.

Software systems find themselves constrained, and unable to cope up with the growth of networks. It was often predicted that the IPv4 address space would soon be exhausted, paving way for the new generation IPv6; this however has not happened yet. The accepted language support on computer networks namely Hyper Text Markup Language (HTML), eXtensible Markup Language (XML) [10] etc., were considered for adaptation of the concept of infotrons. Some advantage in the processing has been observed using XML; it also appears to be friendlier to the class who create solemn and fundamental information.

The prototype IDS developed is useful in legacy systems². IDS would in a way enhance legacy systems' functionality. It would also equip the legacy systems with the capability to adapt to the emerging new technology.

1.4 OUTLINE OF THE WORK

This thesis is organized as follows:

Chapter 1 gives a brief account of the evolution of the problems of information explosion. A brief description of the motivating factors for conducting research in this topic is also given along with the scope. Chapter 2 investigates in more detail the problems and issues faced by DS along with a review of the relevant literature. The approach to modelling of information is examined. The basic concepts and techniques pertaining to Information Retrieval (IR) are reviewed. An overview of previous work done in the field is also presented. Chapter 3 focuses on the architectural aspects of a large distributed software system. It starts with information discovery paradigm and goes on to list various issues related to Information Discovery. The penta-tier architecture introduced as a result of the experiences gained in "Election Counting and Reporting Software" (E CRS) system, is also presented in this chapter. Chapter 4 focuses on the approach taken by Shannon and others towards the concept of information and investigates their suitability in the context of ID. The need to evolve the concept of "INFOTRONS" is highlighted. A formal definition of information, which involves

² Legacy Systems are applications and data that have been inherited from languages, platforms, and techniques earlier than current technology e.g. those that do not utilize the Internet.

infotrons, is given. This Chapter further discusses the proposed Information Discovery System (IDS) and the design methodology followed, along with a brief description of the hardware and software platforms used. The focus is on the alternatives available in the context of Internet and WWW for creation and/or dissemination of information. Further, a front end has been developed for the Library Information System (LIS) resulting in Information Discovery in Library Information System (IDLIS). Inherently IDLIS uses the IDS. The thesis ends with chapter 5 where observations and inferences brought out in the earlier chapters are summarized. Suggestions for further work are also incorporated. A list of references is appended.

CHAPTER 2

INFORMATION DISCOVERY - A REVIEW

2.1 INTRODUCTION

The information space in the universe is growing at an unbelievable rate. To capture all this space in digital form and work on this space for efficient and effective usage of information with the hope to become more powerful and competitive has been a strong desire of everyone. The inability to keep in pace with this explosion of information makes it imminent for an individual/organization to use fast, accurate and effective devices like computers. Therefore, any single individual/organization may become incompetent to look beyond the smog of information to derive any potential benefit from the vast information space at their disposal. Locating the potentially beneficial information at the right time and retrieving it, requires capturing information at the point of its creation. This capturing and storing of information, in turn, results in information being fragmented between various sources. Hence this leads to a distributed system.

Distributed systems have their own characteristics and present challenges to engineers, researchers and scientists. These are described in the next section.

2.2 DISTRIBUTED SYSTEMS: CHARACTERISTICS AND CHALLENGES

The distributed system is defined by [11] as:

“A system in which hardware or software components located at networked computers communicate and coordinate their actions only by message passing”.

According to [12],

“A DS is a collection of independent computers that appear to the user of the system as a single computer”.

Clearly then, the largest and best-known DS is the set of computers, software, and services comprising the [13, 14] Internet / World Wide Web (WWW), which is so pervasive that it coexists with and connects to most other existing DSs.

The reason for going distributed are many, major ones among them being

- Functional distribution
- Inherent distribution
- Load distribution/balancing
- Replication of processing power
- Physical separation
- Economics

Thus most of the distributed systems today have essentially the following properties/characteristics –

- Multiple computers
- Resource sharing
- Concurrency
- Message passing
- Heterogeneity
- Multiple protocols
- Openness
- Extensibility
- Reliability
- Fault tolerance
- Flexibility
- Persistence
- Security
- Isolation
- Decentralized control
- **Information sharing capability**
- Cost saving

The software systems written for such DSs - which may possibly have multiple processors - have features of multi tasking, multi processing, and multi threading, which will be used and accessed by multiple users. Thus necessarily all large software systems are distributed in nature. For example, enterprise-wide business systems must support multiple users executing common applications across different sites.

DS presents many challenges to the scientists and researchers.

- They cannot be *modeled* adequately as Turing Machines (TMs). Unlike TMs, DSs may be *open*, arbitrarily *extensible* by adding new nodes or functionality, and *reactive*, continuously responding to changing environments.
- Prescribing the *specification* for a DS is not an easy task. This is so as DSs do not merely compute a single function or perform a single action. Instead, they perform a never-ending stream of diverse operations. The present specification techniques available from discrete mathematics such as pre-conditions and post-conditions that describe inputs and outputs of sequential programs will not be sufficient to describe the ongoing properties of the system as a whole. Hence one has to rely on temporal logic or related modal logics that provide at least two forms of specifications - properties that must

- invariably hold true
 - eventually hold true.
- *Systems engineering and design*, in case of DS is non trivial. Building a successful DS requires adherence to sound design principles and engineering practices, and reliance on an increasingly standardized set of decomposition and structuring rules. Adequate care must be taken to identify components used (for example – Domain Naming System (DNS), which provides a basis for the naming scheme used on the WWW). Further, care must be bestowed in clearly defining interfaces and implementations. Component management is also an issue in system engineering of DS especially if they have to be reused in an unforeseen environment later.
- *Architectural style used / developed* for a DS involves trade-offs and hence varies considerably between different DSs or each of the subsystems of DS. System or subsystems may be based on one or more design patterns and the variation mainly depends on the nature of the system itself and the components used in system establishment.

Foregoing observations indicate that the main challenges are modelling, specification, engineering, and architectural aspects of a DS. Hence, in the next section a review of the issues in modelling of large information systems is given.

2.3 INFORMATION MODELLING

One trend in computer science and cognitive psychology has been a parallel evolution towards abstraction mechanisms that allow a more generic conceptualisation of the way information is structured, stored, retrieved, and processed, through the formalism of information modelling.

Modelling offers a number of advantages. The very discipline of constructing a model requires a clear view of who or what is being modelled and forces the modeller to select and articulate a particular point of view. A well-formed model will also help anyone understand the endeavour being modelled, and even a poor model will at least indicate to the model builder the need for further clarification of idea because of the sheer exactitude insisted upon. Also, even though specific models may be idiosyncratic with regard to their content, the concepts and techniques used to build them may well be useful in other, very different contexts. In fact, sometimes the most revealing insights arise from taking a perspective that is very far removed from the usual way of looking at things.

Selected literature of the past decades pertaining to various models and modelling techniques within information science are reviewed. Since many terms are not consistently defined, a working definition of major terms is provided. The selected literature can be identified under three major categories: 1) cognitive models, 2) conceptual models, and 3) semantic data models. A related concept is the idea of user

models developed in Artificial Intelligence (AI) research. It appears that many AI applications would work much better if the system incorporated a model of the user. It is worth noting, however, that AI approaches to user modelling have attracted considerable attention in the IR research community. Daniels [15] and Brajnik et. al.,[16] discuss user models in an IR context.

Information system(s) (IS): An information system is considered to be any combination of information source(s) together with any access and/or retrieval/discovery mechanisms provided for its manipulation or use. The purpose of any IS is to connect a human user to an information source relevant to that user's current need, with the expectation that the user will be able to retrieve/discover and internalise that portion of the information required to achieve knowledge that will satisfy the need. Note that this definition does not necessarily imply computer-based IS, although the literature predominates in that area, nor does it necessarily imply document or other text-based systems.

Information model(s) (IM): A model of any kind is first and foremost a representation of a particular aspect of reality. As a representation it is both like and unlike the reality it models. It has its own form and structure, and the correspondences or divergences, between the model and the reality should both be expressible and actually expressed. A model consists of components, links and relationships among the components, and any operations or constraints concerning the behaviour of the

components. A model also requires at least one mode of expression, which may be graphic, procedural, or discursive, among others.

An information model means any representation of information for the acquisition, organization, or manipulation of information. Oberquelle [17] presents a good discussion of the nature of models and modelling and addresses the problem of definition, particularly in regard to articulating the purpose of any individual model. Murray [18] also presents a useful comparison of existing definitions. A model can serve many purposes, but a primary purpose is to communicate something about what it models in order to generate a better understanding of the reality. It is also important to know the perspective of any model. Whitefield [19] suggests an approach that classifies models according to who is doing the modelling and who or what is being modelled. This approach would be useful in identifying possible gaps or biases.

Information modelling: Information modelling refers to the identification of the model's components and their links, to the mode(s) of expression, to the process of model building, and also to the underlying paradigm(s) that affects the type of model being constructed. Model paradigms reflect both the purpose and the boundaries of the model. For example, Lyytinen [20] talks about two paradigms for modelling. The first is "reality mapping," which is essentially a descriptive technique for representing anything that is clearly understood and that which exhibits unambiguous behaviour. The need for certainty in this approach is a severe limitation in its applicability. Lyytinen prefers the second paradigm, "formal language development", which concentrates on representing

the structure, the content, and the use of linguistic messages, because it can deal more accurately with the essentially ambiguous nature of most of the reality. Other paradigms indicate the diverse points of view of the modeller.

2.3.1 Types of Information Models

In considering the entire possible range of information models, a continuum (see figure 2-1) is used. At one extreme is a user with some internal, personal, probably idiosyncratic actuality; at the other extreme, is an information system with an internal system-dependent actuality. Between these extremes lie representations of one or the other of these actualities that attempt to explain or bridge some or all of the gaps in between them. Terminology is a problem in any attempt to define and articulate regions on the continuum. Different authors use the same terms differently, even when those authors share a common discipline.

Those models that lie closest to the user, with focus on what goes on inside an individual's brain are termed cognitive models; those that lie closest to the system end and describe what goes on inside the system are termed data models. Low-level data models are tied very closely to the design of specific database structures. The higher-level group (refer figure 2-1) known loosely as semantic data models attempts to incorporate a user's viewpoint. In the middle of the continuum are models that look at either users or systems or the interactions between them in a relatively objective way; these models collectively form, conceptual models. Thus the distinction between individual's own view of reality - i.e., cognitive models - and someone else's idea of how

a group of individuals should be viewing an IS – i.e., conceptual models can be recognized.

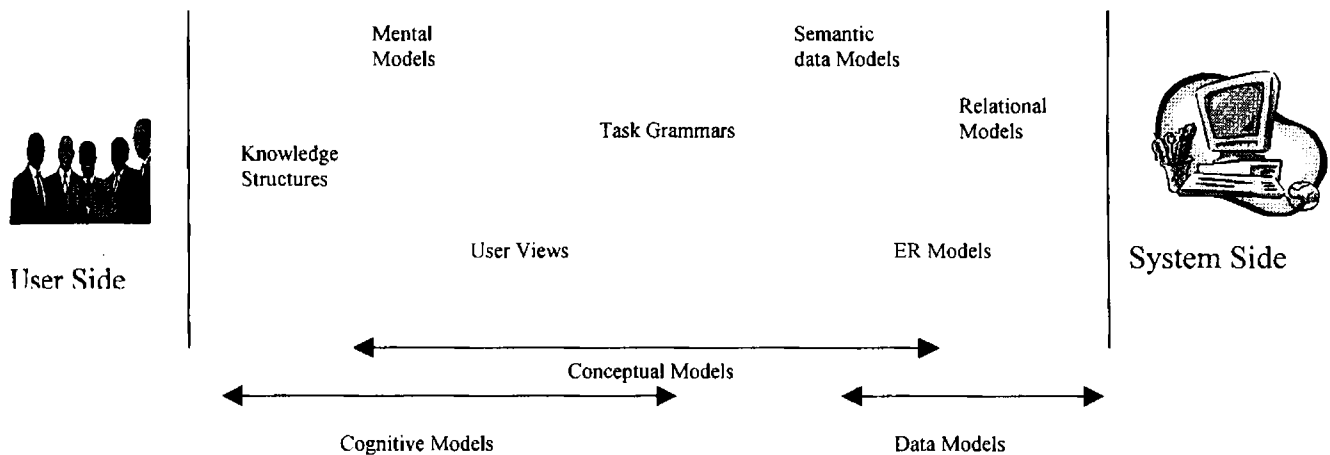


Figure 2-1 An information model continuum

Within the categories of cognitive models and conceptual models, one can identify a number of important subgroups. There is no clear demarcation between any of these types of model, but there is hope that the relative distinctions are useful in providing an insight into the rich variety of models in the field. Despite so many models, there is hardly any model available which can bridge the large gap between the user and the system in an effective and efficient manner.

2.3.2 Cognitive Models

Semantic Memory: Semantic memory is the term used in cognitive psychology to refer to the general knowledge that people have in contrast to the knowledge or the memory of specific events. For many years, cognitive psychologists have been studying how concepts are formed in the human mind. Much of their early work, such as that of

Bruner [21] involved artificial constructs invented by the researchers, but since the mid-1970s there has been a shift towards studying people's conceptions of naturally occurring entities. Rosch & Lloyd [22] and Smith & Medin [23] provide overviews of modern approaches to the study of concepts. Much of this work has dealt with the question of categorization - i.e., how people group objects in their minds. Categories have been typically viewed as being comprised of objects that are similar to each other because they share many features or attributes. However, an object can be missing one or more features and still be considered a member of the category. For example, an ostrich though cannot fly is still considered as a bird. Barsalou [24], among others, has shown that people do not view membership in a category as an all-or-none proposition but think of it (category membership) as a continuum. Vosniadou & Ortony [25] present a compilation of research in this area.

Along with the realization that members of a category can vary in their similarity to each other, comes the awareness that some of the objects in a category exemplify that category better than others. Rosch [26] theorized that people think of categories in terms of prototypes, which are the most typical examples of the category.

Research by Keil [27] and by Medin [28] suggests that a more complete understanding of how people organize and use concepts will have to take into consideration the theories that people have about how the world works. To illustrate, Medin & Soben [29] found that most people judge grey hair to be more similar to white hair than to black hair, but they judge grey clouds to be more similar to black clouds than

to white clouds. They argue that an explanation of these ratings must use the subjects understanding of aging and weather. In an information context Humphrey [30] looked at how work roles influence both perceptions and behaviour in organizations. He found that the individual's view of their position in an organizational structure and their accompanying motivational factors biased their evaluation and selection of the information available.

Category membership is one of several kinds of semantic relations. As noted by Vickery [31], researchers in many different fields, including linguistics, AI, cognitive psychology, and information science (particularly indexing and classification), have identified different types of relationships among concepts. Notable among these are semantic structure and roles as well as the notion of category, both in general ways, as in part-whole relationships, and also in specific ways, as in instance-of relationships. Chaffin & Herrman [32-33] grouped semantic relations into five different classes:

- 1) category membership,
- 2) contrast,
- 3) similarity,
- 4) part-whole relations,

and 5) case relations (such as the relation between agent and action). Within each of these classes they distinguished from four to eight different and specific kinds of relations. Winston & others [34] took a closer look at part-whole relations and observed that everyday use of relation terminology is quite imprecise. This indicates that the kind of distinctions made in some models, are difficult for most people to articulate, even if they can recognize the distinctions.

People learning something new commonly use metaphor models or analogies. Perhaps “analogies” would be a more preferable term in a modelling context than “metaphors”, however, the terms are interchangeably used here to mean reference to any concept other than the one being modelled.

As observed by Schumacher & Gentner [35], an observer who is learning by analogy compares the target system with another system with which the observer is already familiar. As understanding grows, additional metaphors may be added to explain new features, such that advanced learners and experts will have richer mental models made up of composite metaphors and simulation analogies. The process of understanding is iterative as outlined by Miyake [36], who shows that an individual obtains successively deeper insights by adjusting or discarding a succession of mental models. The book edited by Vosniadou & Ortony [25] contains several discussions of learning by analogy.

2.3.3 Conceptual Models

Not surprisingly, the middle range of models in the continuum (refer figure 2-1) embraces the greatest number of models and is the most varied with regard to the user concerns being addressed.

Different viewpoints: Notable literature among the area of conceptual modelling is the review by Lyytinen [37]. It contains an extensive section on the user and includes the user’s environment and the use process, discussing information models for the

purpose of IS design. Lyytinen highlights two points: 1) problems with IS arise from the changing social environment within which any system functions, and 2) that designers need to recognize the complexity of that environment by using more than one approach or strategy in modelling both user and system requirements.

Another interesting viewpoint is by Belkin [38-39], who discusses the entire range of models possible in the context of information seeking in bibliographic IR systems. Belkin enumerates seventeen such models but then focuses on the process of model building, which he sees as the essence of mediated information retrieval. In an interactive, iterative process, this cooperative venture between the user and the intermediary aims at consensus on both the nature of the problem and the appropriate strategies for retrieving information relevant to the resolution of that problem. Key points in this work are that the models should be produced jointly and should reflect what each of the participants understands about the other.

In a third viewpoint Daniels [15] discusses user models in IR systems from an evaluative perspective. She concludes that most models are reasonably effective, only because they operate in restricted domains in which strong simplifying assumptions have been made. She notes the lack of comprehensive models and calls for greater attention to evaluation and resolution of ambiguities.

User Views: Many models are concerned with what might be characterized as user views. These models present an interpretation of an actual or theoretical IS from the

perspective of a user or group of users with specific characteristics that the model builder deems to be relevant to the use of the IS. Generally speaking, models that address user views tend to look at differences between and among specified groups of users or at a single, stereotypical user. Many of these groupings tend to be stripped-down, preconceived categories created from the perspective of optimum system performance. The most common dichotomy is that of “novice” v/s “expert” users, where the expertise being modelled is computer use rather than domain or application knowledge.

Task or Function Models: In these models we analyse either the functionality of the IS or the details of the task that the user wishes to carry out. This type of modelling tends towards quite formal representations and includes rules for both the inclusion of components and the mode of expression of the model. A distinction may be made between models whose expression takes the form of a created language or grammar and models whose expression takes the form of procedural, behavioural, or algorithmic specifications.

Grammars: Several approaches, reviewed by Farooq & Dominick [40], are based on tools used to describe the syntax of natural or programming languages. Most of them are labelled as grammars. For example, Reisner [41] developed a grammar of computer tasks using a notation scheme called Backus-Naur Form (BNF), which she later extended [42] to include the cognitive activity of the user. Moran, on the other hand, acknowledged the need to represent both the semantics and the syntax from the outset. His Command Language Grammar (CLG) consists of descriptions of system use at four

levels: 1) the task level, 2) the semantic level, 3) the syntactic level, and 4) the interaction level. The interaction level deals with the physical manipulation of the interface eg. keystrokes. At the task and semantic levels, CLG consists of definitions of the objects in the system and the operations on those objects. The objects and operations are organized into semantic hierarchies. At the syntactic and interaction levels the grammar provides definitions of the actual commands used to carry out the intended operations. Moran's development of CLG has thus a strong psychological basis.

Noting that BNF has no facility for linking rules that have a similar but not identical structure, Payne & Green [43] developed a grammar that was more sensitive to "family resemblances" among commands. Payne and Green organized their approach, which they called "task-action grammar" (TAG), on different levels (similar to Moran). The lowest level consists of simple tasks, which are combined into rule-schemas for more complex operations. Payne and Green's approach is heavily semantic. Simple tasks are viewed as concepts, and family resemblance among task concepts is represented through shared features. In the works mentioned above there is no due consideration given to the *pragmatics*, which we feel, is most essential if a clear understanding has to be achieved.

Procedural and other models: Some of the models belonging to this group are the GOMS model (Goals, Operators, Methods, Selection) and the KLM (Keystroke-Level Model), developed by Card & others [44] and the BASIS model (Behavioural Approach to the Specification of Information Systems) of Leveson & others [45]. An interesting analysis of some of these models is given in Green & others [46].

The GOMS model contributed both to Moran's CLG and to the work of Kieras & Polson [47], who used production systems to model the user's representation of the system interaction task. These authors then introduced Generalized Transition Networks (GTNs) to model the operation of the system itself. A GTN is a graphic way of representing a system that can exist in a finite number of states (a finite-state machine). In a transition network, nodes represent states of the system and arcs represent transitions between those states. The operation of the system is visualized as a traversal of the network that ends when a goal state is reached. In a GTN, individual networks can be nested, enabling different networks to represent activities at different levels.

Payne & Green [48] conducted a learning experiment with a novel retrieval system to test the psychological validity of TAG, CLG, BNF, and Kieras and Polson's production system approach. The results supported TAG, although some of the contrasting predictions they ascribed to the other schemes were questionable. To test the usefulness of formal specifications in system design, Frohlich & Luff [49] used a formal method devised by Foley & Van Dam [50] to design a form-filling program and to convey the design to the programmers. Foley and Van Dam's approach is multi-layered. Frohlich and Luff's instantiation of the method included a textual description of the conceptual design of the program, frame-based descriptions of the interaction semantics, and a finite-state diagram representation of the interaction syntax. Finite-state diagrams are essentially the same as the GTNs used by Kieras and Polson. Frohlich and Luff found that the specification technique was cumbersome at times and that specification of the semantics and the syntax had to proceed simultaneously, rather than sequentially

(semantics first, then syntactic), as recommended. Nevertheless, the specification was precise enough to enable the entire program to be implemented from it. Here again we see that *pragmatics* has not been emphasised by anyone.

The original work by Bobrow [51] is worth mentioning as it identifies “dimensions of representation,” which should be considered for any model, notably the concepts of domain and range, inference, and access, and the probability that multiple views of the concept being modelled.

2.3.4 Semantic Data Models

Data models represent the area of the continuum most closely associated with the system part of an IS and tend to be designed with a view to mapping the model into the internal actuality of the system. Certainly the traditional data models, such as the hierarchical model, the network model, and the relational model of Codd [52], travel only the most modest distance from the actual data structure. Even the entity-relationship model of Chen [53] merely provided a graphical expression for the relational model, although this was of considerable benefit for relational database users. Davis [54] found that beginning users supplied with an entity-relationship diagram or a data-structure diagram made fewer query errors than users who were given only a tabular description of the database contents. Jarvenpaa & Machesky [55] obtained similar results with a data-modelling task.

The traditional data models concentrate on the syntactic and structural aspects of the data but do little to illuminate the underlying meaning of the data or to articulate any inherent, logical, relationships among them. With the development of more sophisticated IS, there has been an accompanying impetus for models that facilitate user understanding of each system while obviating the user's need to know anything about the physical structure of the computer database. Predominant in the extensive research on data models is an increasing concentration on the problem of making these models reflect more of the semantic complexity of real-world information. Models that attempt to do this are known collectively as semantic data models.

Much of the literature on data modelling are directed towards database designers and therefore emphasize problems important in the context of creating an operational computer system under a set of specific application constraints. While this is certainly a valid viewpoint, not everyone agrees with this system design perspective. For example, in their basic text on data models Tschritzis & Lochovsky [56] state that the proper role of a data model is to serve as a medium of communication *for people* in general, such that the structure imposed by the model will not disagree with the natural structure of reality as perceived by the human user. They discuss the need for at least two types of data model -one dealing with what they call the "infological realm," to map real-world information into basic human concepts, and the other dealing with the "datalogical realm," which would map these basic concepts into their computer representation. With this understanding, traditional data models would be "datalogical," and semantic data models would be at least a step towards the "infological."

Since most semantic data models represent specific applications created to meet specific contextual requirements, there does not seem to be any recognized, generalized model yet in existence. The basic review articles, by Hull & King [57] and Peckham & Maryanski [58] conclude that the literature does seem to agree on the path of evolution of semantic data models, but a major difficulty here is the lack of a precise definition. Semantic models evolved mainly through attempts to extend the relational model by enriching it with semantic abstractions adapted from linguistic research. A working definition in most instances is simply that semantic data models represent more semantic complexity than relational models. As a functional definition, this is clearly unsatisfactory. However, although overall consensus has not yet been attained, the underlying issues are the same even though the proffered solutions differ widely in their implementation details. Also, the literature does indicate general agreement on the most common components of semantic models, especially those that were adopted from linguistic research.

Semantic Abstractions: Semantic abstractions are ways of specifying relationships among linguistic concepts that articulate the sometimes-subtle differences in meaning (see the discussion under semantic memory, above). Four such abstractions are commonly used in most semantic models: generalization, aggregation, classification, and association.

Generalization: This abstraction occurs when objects or entities (i.e., any concept or thing) are grouped in a hierarchical relationship, in which objects at a lower level are

seen as subtypes of the higher levels. For example, the objects “biographies” and “novels” can be seen as more specific instances of the objects “books.” This is also known as the “IS-A” relationship.

Aggregation: This abstraction occurs when objects are grouped in a component relationship, wherein each object contributes to the makeup of the larger object. For example, the objects “pages,” “covers,” “bindings,” and “ink” can be grouped to form one view of the object “book,” while “title,” “author,” and “publisher” can be grouped to form another view. This is also known as the “IS-PART-OF” relationship.

Classification: This abstraction occurs when objects are grouped because they are specific examples of a more general object type e.g., “*Kalam*” and “*Agnisakshi*” are specific instances of “novel” or “book_title.” This is also known as the “IS-INSTANCE-OF” relationship.

Association: This abstraction occurs when objects are grouped by virtue of satisfying some criterion or criteria. For example, the set object “Science-fiction” may consist of all books available in paperback, illustrated with fluorescent or lurid covers, and selling for less than Rs 100.00. This is also known as the “IS-MEMBER-OF” relationship.

Individual authors have slightly different views, which at times seem to be merely a matter of terminology, while at other times they seem to be more substantive. For example, Peckham & Maryanski [58] use the terminology noted above, while Hull &

King [57] also speak of *fundamental* object types exhibiting “INSTANCE-OF” or “IS-A” relationships (i.e., classification and generalization, respectively) and *constructed* object types (i.e., aggregation and association). However, Hull and King distinguish between semantic models that “encapsulate structural aspects of objects” and object-oriented models that “encapsulate behavioural aspects of objects.” The object-orientation approach simply shifts the emphasis from the relationships among model components to the behaviour of individual components or component groups, wherein the fewer the constraints on object behaviour, the more “strongly typed” the object is considered to be. The need for such a distinction is unclear since it implies that models must necessarily take one approach or the other but not both. There does not seem to be a reason for this restriction. Other authors, such as Brown [59], take the view that object-oriented models are used in the implementation of semantic data models to bring the concepts closer to the internal, physical level necessary for data storage and retrieval.

In another view, Potter & Trueblood [60] use the term “semantic” models to refer to the use of generalization and aggregation mechanisms only, while they refer to models exhibiting additional constructs, including classification and association (they use the term “membership” rather than association), as “hyper-semantic.” These hyper-semantic models, described further in [61] also borrow concepts of inference and heuristics.

2.3.5 Other aspects of modelling

Although every semantic model deals with the concepts of objects, entities, attributes, and relationships in one way or another, there is no rule or restriction on how

these concepts may be applied in specific situations. Thus, in one model a concept can be categorized as an attribute, while in another model that same concept might be considered as an object or a functional relationship. For example, the concept of “publishing” may be considered: 1) as an attribute of an entity (“DOCUMENT” - a document has a publisher); 2) as an object (“PUBLISHER”) associated with another object (“DOCUMENT”); or 3) perhaps as a relationship (“publishes” or “is-published-by”) between two objects (“DOCUMENT” and “PUBLISHER”).

However, these techniques and components are certainly not enough to represent completely the links between and among the objects in any model. Other problems must be addressed. While not all models deal with all problems - indeed no single model attempts to do so - those most frequently discussed are described as follows.

Inheritance/derivation: Most models achieve economy of expression by linking some attributes to a more general object type and permitting lower-level objects to inherit those attributes. In a journal, for example, each volume and each issue of that journal would inherit the “attribute” JOURNAL.TITLE. The problem, of course, is how far to carry inheritance. Clearly a distinction needs to be made among JOURNAL.TITLE, ISSUE.TITLE (if one exists), and ARTICLE.TITLE. Other attributes may not be specifically defined but can be derived or constructed from information that is available in the model.

Multiple values: Some instances of multiple values can be handled by allowing objects to be set-valued. In this way, for example, each instance of a VOLUME.ISSUE can accommodate a number of different articles, or each instance of an ARTICLE can have more than one AUTHOR. It is not so easy to avoid redundancy when an object may have multiple relationships with other objects. For example, it is clear that the same AUTHOR may be associated with a number of different ARTICLES, each of which may be associated with a JOURNAL_ISSUE and more than one REFERENCE_LIST. Further, an AUTHOR may be associated with another AUTHOR in a number of different ways, as in a co-author, citing, cited, reviewing, or reviewed relationship. These multiple values may not always be obvious since a relationship may be explicitly defined in the model or may be implicitly derived through some type of matching function. In addition, if a model supports multiple views of the same data, each view will have a value that must be retained and distinguished from other views. The Semantic Database Model (SDM) of Hammer & Macleod [62] is directed towards this concern in multiple views.

Temporal issues: A specific kind of multiple value occurs with time-related values. For example, JOURNAL_PUBLISHER may change, but each value would be valid in a given time frame. Similarly, objects may exist only in a temporal sequence, as in a progression from MANUSCRIPT to REVISION to PUBLICATION. This example also represents a possible instance of the problem of recursion and the question of version control.

Constraints: Constraints cover a number of points. It may be necessary for the model to permit restrictions on operations or relationships, either established a priori or added later, as in the temporal values noted above. Set membership is a prime candidate for this. In addition, constraints would be required for insertion, deletion, or modification of model elements. One of the earliest semantic models to address relations and integrity rules is the RM/T model (Relational Model/Tasmanian version) of Codd [63]. Bowers, [64] discusses the need for a significant number of constraint mechanisms to be represented in a semantic data model if that model is to be useful in system design.

Static / dynamic aspects: With a static model, all objects and so forth would be determined in advance, and only those operations specifically defined in the model could be carried out. A more realistic representation of the world would permit each user to define objects and operations as required, but that means that the model must somehow provide mechanisms to do this. To illustrate with a journal example, a user might wish to define an object called ARTICLE_CITATION, which would be made up of JOURNAL.TITLE, VOL_#, ISSUE_#, ISSUE.DATE, ARTICLE_TITLE, ARTICLE_AUTHOR, and ARTICLE.PAGE_RANGE, or the values NUMBER_OF_PAGES and NUMBER_OF_REFERENCES might be desired. The problem of providing a generic mechanism for user-defined operations is not trivial. The SHM+ model (Semantic Hierarchy Model extended version) developed by Brodie [65] is one attempt to deal with static/dynamic issues.

It appears that the consensus is simply that any semantic model should indicate all the relationships between and among data objects necessary to represent fully that view of reality chosen by the model builder and that the model should provide semantics that specify the valid states, operations, and reactions of the system. The lack of a generic structure is both strength and a weakness because it permits each model to present a highly subjective view but almost guarantees divergences of opinion. Specific models select and extend existing techniques and introduce new ones tailored for the given situation. One model worth looking at for its extensions to semantic modelling techniques is SAM* (Semantic Association Model-enhanced version), which was created for statistical databases by Su [66] and later was adapted for integrated manufacturing applications [67].

Recently object oriented modelling is gaining momentum and is largely deployed in practice [68]. Nevertheless many critics feel that they are not a panacea in their usage at every step of IS development though they help programmers in easily conceptualising their coding efforts for a software system [69]. Moreover, recently UML is widely used as an alternative. UML though inherently object oriented focuses on the association with situation and scenario as is evident in use case modelling. UML is being used extensively within the unified approach for developing complex systems [70].

To conclude, the area of information models and modelling is characterized by its lack of boundaries. Nevertheless, this is an important area and likely to become more so because people are becoming more sophisticated in the need for and use of

people-oriented (i.e., user friendly) systems. There are however major problems of definition here. Too many people are using the same terms to mean different things, and not enough authors define what they mean. Although it is not necessary to achieve consensus on terminology in order to work in the area, a clear agreement would definitely facilitate discourse. What is necessary, however, is for researchers and practitioners to make explicit the underlying assumptions, purposes, and point of view of any model as well as the definition of components and structure. Accordingly observations state that, too many models in the literature are ill formed and fuzzy.

Another difficulty in trying to achieve a sense of the area as a whole is that it is broad and fragmented. Some regions of continuum (refer figure 2-1), particularly, those in which there has been a long tradition of research, are well developed while others have hardly been touched. The tractable, more easily resolvable areas have the richest number of models and the most coherent “theories” that tie some or all of them together. Certainly the whole idea of information modelling as an integrated discipline is in need of a holistic approach, which could serve two purposes: 1) it could offer a provisional theory for the entire information-mediation process, and 2) it could provide a framework to direct future research efforts.

Conceptual models are moving closer to cognitive models and away from pure data structures. Cognitive models are moving out of the laboratory and into the real world. Collaboration between the two strengthens both. Semantic data modelling techniques provide a level of terminological precision and model definition that most of

the work on semantic memory lacks. One contribution of cognitive psychology has been the notion of empirical research to evaluate and compare models. Psychologists can contribute empirical data about how people naturally categorize concepts and the semantic relationships they identify. However, empirical research in any area dealing with models must be carefully designed to ensure useful results. As Carroll [71] notes, there is a need for “rich analysis of real users working on real tasks.” The problem with the real world however is that one cannot easily abstract and simplify as is needed in carefully controlled experimental design. This does not mean that empirical work cannot be done. It only means that it must be done with greater care and precision. Well-defined models will make this process easier. By modelling one’s techniques and knowledge, information scientists should be able to assist more fully in the design of better systems, which include better interfaces. They must also keep in mind the impact of “*pragmatics*”.

Modelling is just one aspect; to really reap the benefits provided by different models one has to devise techniques to harness the models to their fullest ability. Scientists and the researchers in the field of information retrieval have been doing just that, over several years. The following section reveals their approach and uncovers how any technique is suitable for the underlying model adopted.

2.4 INFORMATION RETRIEVAL TECHNIQUES

Information retrieval (IR) traditionally deals with information in the form of text alone. There are differences among research communities in conceptualizing information. Despite differences, IR techniques have been found useful and hence a review of these techniques is presented below.

A retrieval technique is defined as a technique for comparing the query with the document representations. Broadly IR techniques can be classified into operational (eg boolean or full text, string searching retrieval) and experimental (eg probabilistic retrieval) based on their underlying algorithms in IR system.

Classifying the retrieval techniques is also possible based on the query matching for documents, resulting into exact match or partial match. Partial match also includes documents from the exact match category; hence exact match techniques can be treated as a special case of partial match techniques. Yet another way to classify IR techniques is based on whether the query is matched with a single individual document or a networked collection of interrelated documents. The figure 2-2 and 2-3 shows the classification hierarchy according to this scheme of classification.

The individual category breaks down into retrieval techniques that use a feature-based representation of queries and documents as well as techniques that use a structure-based representation. In a feature-based technique, queries and documents are

represented as sets of features, such as index terms. Features can be weighted and can represent more complex entities in the text than single words. The structure-based category is divided into representations based on logic, that is, those in which the meaning of queries and documents are represented using some formal logic, and on representations that are similar to graphs, in which documents and queries are represented by graph-like structures composed of nodes and edges connecting these nodes. Natural Language Processing (NLP) or statistical techniques can produce such graphs (eg semantic nets and frames).

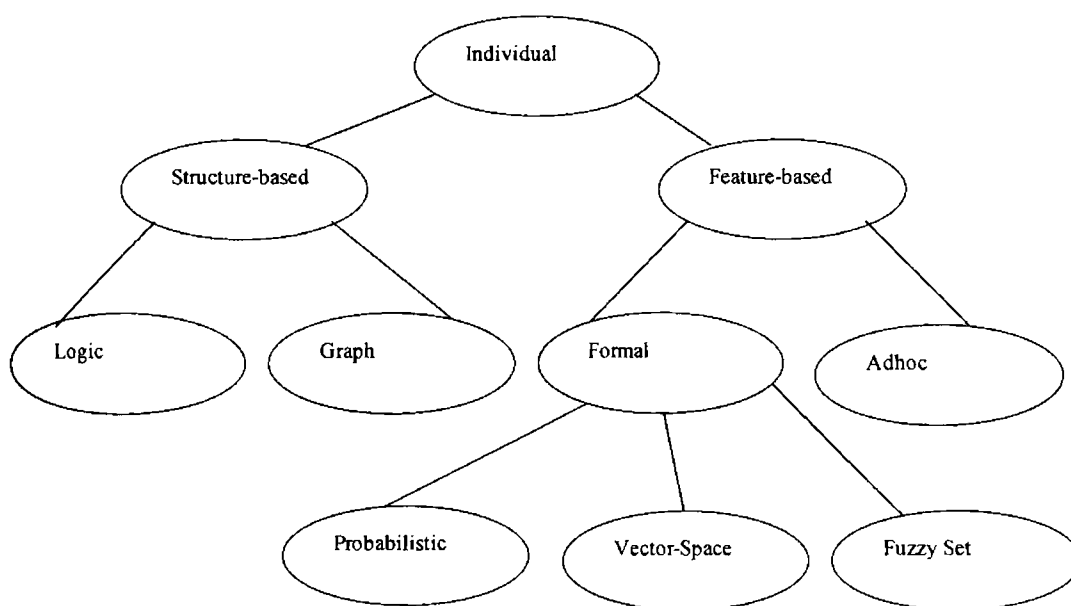


Figure 2-2 Classification of IR techniques for a single individual document

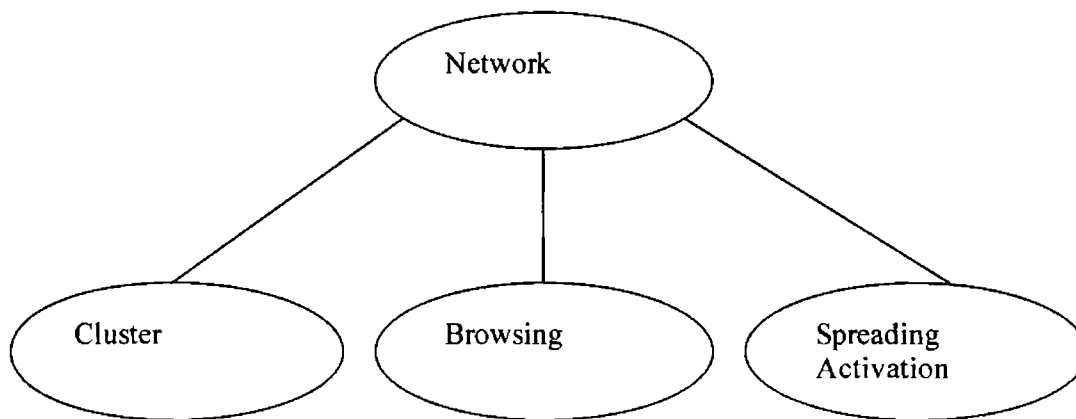


Figure 2-3 Classification of IR techniques for a network of interrelated documents

The feature-based category includes techniques based on formal models (including the vector space model, probabilistic model, fuzzy set model and others) and techniques based on ad-hoc similarity measures.

2.4.1 Feature-based techniques

These techniques are used to compare queries with documents represented as sets of features or index terms. The document representatives are derived from the text of the document, either manually or by automated indexing [72-74].

Features can represent single words, stems, phrases, or concepts and can have weights associated with them. The weights are typically derived from the way the feature is used in an individual document, such as a within document frequency weight (tf), or the way it is used in the document collection, such as the inverse document frequency weight (idf)³ [73-74]. The interpretation of the weights, the way they are used, and the way they are calculated depend on the particular retrieval technique and the underlying retrieval model selected.

In discussing retrieval techniques in this category, it is assumed that a document has a representation consisting of a vector of terms (d_1, d_2, \dots, d_m), where d_i indicates the presence or absence of term i and has the value 1 or 0. Weights associated with these terms can be introduced as needed. A query has a similar representation, with q_i referring to the i^{th} query term.

Formal Techniques: These retrieval techniques are based on formal models of document retrieval and indexing. Here, major modelling approaches that have been used for information retrieval include vector space, probabilistic, and fuzzy set [75-78].

Vector space model: In the vector space model, documents and queries are vectors in an n -dimensional space, where each dimension corresponds to an index term. The model has intuitive appeal and has formed the basis of a large part of IR research,

³ The within-document frequency is the number of times an index term (usually a word stem) occurs in the document text (usually the abstract). The inverse document frequency is the inverse of the relative frequency of a term in the collection. The logarithm of this ratio is often used, giving a weight of $\log N/n$, where N is the number of documents in the collection and n is the number of documents that contain the term.

including the SMART system [79,73]. Although this was one of the first models proposed, modifications to it are appearing [80-81]. From the point of view of retrieval techniques, there have been very specific recommendations about how the model should be applied in operational systems [82,72]. Notable points about this retrieval technique are, firstly, the exact formula for calculating term weights, such as *tf. idf*, may vary from one system or set of experiments to another. For example, it is common in calculating the *idf* weight to normalize the document collection frequency with the maximum collection frequency rather than simply the number of documents in the collection; this is done to expand the range of *idf* weights that result. Similarly, in some experiments the *tf* weight is normalized with the maximum within-document frequency, and in others it is not normalized. These details can have a significant impact on the effectiveness of a system, and it is essential to check their validity. Secondly, when similarity measures are utilized, the query is not directly compared with every document in the collection to produce a ranking. Typically, an inverted file is used to exclude documents that have no terms in common with the query. Refinements of this technique have been devised to reduce search time further [83-85]. The weights used follow directly from the retrieval models, and if the collection is dynamic, they can be more accurately calculated during retrieval. The identification of important relationships among words and the expansion of terms to include thesaurus classes can also be part of retrieval since it is those relationships and terms that are relevant to a particular query that should be identified [86].

An important extension of the retrieval techniques based on the vector space model is the extended boolean retrieval [87-91]. This technique overlaps the structure-based category because it uses structured (boolean) queries. That is, the query is formulated with index terms and the boolean operators AND, OR, and NOT. The prevalence of systems that use boolean queries has led researchers to consider the problem of producing ranked output from boolean specifications. In this approach to the problem, a similarity measure is defined that ranks documents, giving precedence to those that match all or part of the boolean specification. For example, consider a situation in which document D1 contains terms A and B, document D2 contains terms B and C, and the query is A AND (B OR C). Ignoring the effect of term weights, the standard cosine correlation would rank documents D1 and D2 equally because both have two query terms. The extended boolean similarity measure would rank document D1 higher because it matches the boolean query specification. The addition of term weights makes the calculation more complex, but the general effect is the same.

Probabilistic model: As an alternative approach, the probabilistic model was introduced by Robertson & Spark Jones and Van Rijsbergen [92,78]. Different forms of this model are discussed by [75], but these have not contributed significantly to the development of retrieval techniques. The advantages of the probabilistic model are the insight it gives into techniques that have been used in previous research and that it is a powerful framework for developing new techniques.

The retrieval techniques based on the probabilistic model are very similar to those developed from the vector space model. The basic aim is to retrieve documents in order of their probability of relevance to the query [93]. If we assume that document term weights are either 1 or 0 and that terms are independent of each other, this can be shown to be achieved by ranking documents according to the formula given below:

$$\sum d_i q_i \text{ ----- (1)}$$

where, d_i is the *tf.idf* weight and q_i is a weight equal to $\log \frac{pr_i (1-pnr_i)}{pnr_i (1-pr_i)}$ where pr_i is the probability that term i occurs in the relevant set of documents, and pnr_i is the probability that term i occurs in the non relevant set of documents.

The problem in applying this ranking function is to estimate the probabilities in the query term weights. Experimental results have shown that pnr is best estimated from the entire collection [94]. This means that the formula, $pnr_i = n_i/N$ (where n_i is the number of documents that contain term i and N is the number of documents), is a reasonable estimate. If a set of relevant documents is available (for example, after feedback), the best estimate for pr_i is r_i/R , where r_i is the number of relevant documents that contain term i and R is the total number of relevant documents. The actual estimation formula that provides the best results is $(r_i + 0.5)/(R + 1)$ [95]. When r_i is 0, this estimate is too high and a value of 0.05 for pr_i is used. If a set of relevant documents is not available, as in an initial search, the retrieval technique suggested by the simple probabilistic model is approximately equivalent to using the *idf* weight in formula 1 [96].

The probabilistic model can be extended to use within document frequency information [97]. In this case, the term weight (d_i) used in formula 1 is $ts \cdot idf$, where ts is a term significance weight that measures the importance of a term to a particular document. The ts weight is best estimated with the normalized within document frequency. This form of the ranking function is virtually identical with the one that was developed from the vector space model [98]. Another approach to incorporating term weights is described by Fuhr [99].

A number of proposals have been made to remove the term independence assumption of the probabilistic model. Harper, Van Rijsbergen, and Yu et. al., [94,100-102] describe retrieval techniques that involve calculating correlations between terms (or term dependencies) in the document collection. These dependencies are then used to expand queries and change estimates of the relevance of documents. It has been shown that if there is sufficient information about the occurrence of terms in relevant documents, these retrieval techniques could significantly improve effectiveness. In practice, however, estimation problems make it difficult to realize any of the potential benefits [103,102].

Fuzzy set: A fuzzy set approach to information retrieval has been discussed in many papers [75]. To summarise, the main contribution of the work in terms of retrieval techniques has been the integration of boolean queries with ranking techniques. This integration is limited, however, when compared with extended boolean retrieval based on the vector space model or the use of term dependencies in probabilistic models.

Ad hoc: A number of similarity measures for comparing queries and documents have been proposed in the literature [104-105,79]. Many of them were developed in the context of numerical taxonomy [106].

2.4.2 Structure-Based Techniques

In this category of retrieval techniques, either the query or the documents or both are represented by more complicated structures than the sets of terms used in feature-based techniques.

Logic: It is theoretically possible to represent the information conveyed by the text in documents as sentences in a formal logic. For example, the statement, "CMC sells computers." could be represented in first-order predicate calculus as (sells CMC computer). Similarly, the statement, "If a company sells computers, it is financially viable," could be represented as (forall (x) (if (sells x computer) (viable x))). More complex sentences require more complex logic representations [107]. Given a logic representation of document content, a query in the same logic could be answered by inference using the rules associated with that logic. For example, the query (viable ?) can be answered by forward chaining (a form of *modus ponens*) from the sentences given. This approach to information retrieval has been studied by Walker & Hobbs and Simmons [108-109] and is related to the natural language research by Schank & others [110-111]. The critical problem with this approach is the translation of the text into logic. In current experimental systems, this is done manually. Van Rijsbergen, [112] describes retrieval as a process of determining if a query (expressed in logic) can be

inferred from a document's content (expressed in logic). In many cases, this inference cannot be made directly because information is missing in the document; in these cases the inference is uncertain. The notion of uncertain inference is also the basis of the RUBRIC system [113-114]. A part of this system provides standard full-text document retrieval. Queries, however, are represented as rules that describe how pieces of evidence in the document text can be used to infer the relevance of the document. Numbers are attached to the rules to represent the certainty of the statement. For example, a query may be represented by the rules -

“information” AND “retrieval” → information-retrieval (0.6)

“information” ADJACENT “retrieval” → information-retrieval (0.9)

“probability” → probabilistic-model (0.5)

“information-retrieval” AND “probabilistic-model” → probabilistic-information-retrieval (0.9)

If a particular document contains the sentence, “retrieval of information with high probability of relevance is desirable”, the rules above will be used to infer that the document is about information retrieval with a certainty of 0.6, probabilistic models with a certainty of 0.5, and probabilistic information retrieval with a certainty of 0.45 (0.5×0.9). This type of rule-based representation has also been used for thesaurus information to infer concepts that are related to terms in a query [86,115]

Graph: The general characteristic of a graph-like representation is a set of nodes and edges (or links) connecting these nodes. Specific examples include semantic nets and frames [107], which are typically produced by NLP. Simpler network structures can

also be produced by statistical techniques, such as those used in the ASK (Anomalous States of Knowledge) project [116-117]. Retrieval techniques in this category must look for similarities in the structures of query and document graphs. This similarity can be used directly to determine if a document should be retrieved or to modify a document ranking.

Belkin and Kwasnik [117], for instance, describe the identification of regions of interest in graphs generated by a co-occurrence analysis of narrative “problem statements” (ASK representations). The areas are identified as specific kinds of structures within the graph, such as groups of highly interconnected nodes at high association strengths or two such groups weakly connected with each another. The structural nature of the ASK graph is then used to determine first the terms that will be matched against the database of documents, then the ranking of the retrieved documents. The first retrieval stage takes little account of the relationships of the terms within the document structures (computed in the same way as the ASK structures), using instead the ASK structure to identify terms that either must or may appear in the text structures. In the second stage, the candidate set of retrieved documents is then ranked according to how well each satisfies desirable criteria of term position, importance, and relationship to other terms as established by rules associated with the structures identified in the ASK representation. This retrieval method has not been tested in a formal experiment, but it appears to have some possibilities of providing a way to use graphical representations to choose different retrieval stratagems. It may also be of use in specifying term dependencies that can then be used by other retrieval techniques.

Most of the search engines available on the Internet today use Boolean techniques for specifying the queries. Bookstein [75] mentions several undesirable characteristics of boolean retrieval. In a simple case of exact match searching, the following disadvantages may be identified. It

- misses many relevant texts whose representations match the query only partially.
- does not rank retrieved texts.
- cannot take into account the relative importance of concepts either within the query or within the text.
- requires complicated query logic formulation.
- depends on the two representations being compared having been drawn from the same vocabulary.

Even with all these disadvantages the technique is still predominantly used in operational IR systems due to the following two reasons:

- structure of boolean statements represent important aspects of user's queries or problems.
- investment in current IR systems is so high that changing them is economically not feasible.

The efficiency improvements have been achieved with techniques such as, file organization [118], specialized hardware [119-120], and text compression [121]. Perhaps the most interesting approach to the question of extending the logic of exact matching has been the effort to deal with boolean techniques as a special case of either vector or probabilistic models. Salton and colleagues have developed an extended vector model, and Croft has proposed a method for making use of the relations established in a boolean

query within a probabilistic search model. Other attempts include Fox's relational thesaurus for boolean retrieval [122], which is equally relevant for probabilistic or vector retrieval as are suggestions for a "user thesaurus" [123] by Bates. The problem of reconciling the query vocabulary with the document or with the index vocabulary in matching is prevalent in these attempts.

2.4.3 Network of interrelated documents

A distributed system contains, more often than not, documents which are interrelated and spread across the network. Three major variations are identified and are discussed below. Among these, cluster behaviour is common in Intranets as opposed to the browsing, which is used in Internet.

Cluster: A cluster is a group of documents whose contents are similar. A particular clustering method gives a more detailed definition of a cluster and provides a technique for generating them. The use of clustering for information retrieval was a major topic in the SMART project [79]. The approach used was to form a cluster hierarchy using an ad-hoc clustering technique. The cluster hierarchy was formed by, dividing documents into a few large clusters, dividing these clusters into smaller clusters, and so on. A top-down search of the cluster hierarchy is performed by comparing (using a similarity measure) the query to cluster representatives of the top-level (largest) clusters, choosing the best clusters, comparing the query with representatives of lower-level clusters within these clusters, and so on until a ranked list of lowest-level clusters is produced. The documents in the top ranked clusters are then ranked individually for

presentation to the user. A cluster representative can be generated in various ways, but in general it represents the average properties of documents in the clusters. The cluster hypothesis was introduced as a basis for using cluster searches to improve retrieval effectiveness relative to ranking individual documents.

The emphasis on small, well-defined clusters has led to the development of retrieval techniques based on the generation of the document's nearest neighbours [84,124]. A document's nearest neighbours are those most similar to it, and a cluster of nearest neighbours is very similar to the lowest-level single-link clusters [125]. Given a network of documents connected to their nearest neighbours, it is possible to generate clusters and their representatives at search time with considerable storage savings [84].

Other research on clustering techniques has compared the relative effectiveness and efficiency of different types of clusters [126].

Browsing: If the documents, terms, and other bibliographic information are represented in the system as a network of nodes and connections, the user can browse through this network with system assistance. Browsing is an interesting retrieval technique in that it places less emphasis on query formulation than do other techniques and relies heavily on the immediate feedback provided by user browsing decisions. The THOMAS system [127] uses index terms as starting points in a simple network of documents and terms. Through dialog with the user, the system uses the network to build a model of the user's information need that includes relevant documents found during the

process. The browsing component of the I³R system [86] contains nodes that represent documents, index terms, domain knowledge, authors, and journals. The links represented include indexing information, thesaurus information, nearest neighbours, citations, and authorship. The system makes browsing recommendations based on the number and types of connections between and among documents but allows users to choose any path.

Other research in browsing concentrates on the use of visual representations of the document database to acquire information from the user interactively [128].

Spreading activation: Spreading activation is a retrieval technique that has some similarities to browsing. A query is used to “activate” parts of a network that describes the contents of documents and how they are related to each other. In the simplest case, the query would activate index term nodes that are connected to document nodes and other terms. In more knowledge intensive networks, the links and nodes represent concept from the subject domain and how they relate to each other as well as the documents that contain those concepts [129-130]. From the “start nodes” provided by the query, other nodes connected to those nodes are in turn “activated” (hence, the term “spreading activation”). Criteria, such as threshold values that decrease as the activation propagates through the network or, rules about the reasonableness of the inference implied by using a particular link, are used to control the spread of activation. Activation can converge on particular document nodes from a number of links. These highly activated nodes are retrieved.

2.4.4 Feedback Methods

Relevance feedback techniques cannot be considered as retrieval techniques. Rather they are used to refine the request model, which is then used for another search. Feedback techniques are, however, an extremely important part of ensuring that a IR system will be effective.

These techniques were primarily developed in the context of feature-based retrieval [79] although the principles apply to any retrieval technique. The main part of relevance feedback is the adjustment of weights associated with query terms. This adjustment is done on the basis of term occurrences in the documents identified by the user as relevant. The probabilistic model has a particularly strong motivation behind this weight adjustment in that the identified relevant documents provide a sample to estimate the pr values. The query (or request model) can also be changed by the addition of new terms from relevant documents. Some control is needed over the number of new terms added, and it seems that the most reliable method is to have users identify interesting terms in relevant documents.

2.4.5 Predicting appropriate retrieval techniques

A number of experimental results have indicated that although different retrieval techniques appear to have similar results, they often retrieve different relevant documents for the same query. The use of alternative content representations, such as citation information, can also result in the retrieval of different documents. Different techniques

also vary in their performance for different queries. If the best results from different techniques for individual queries could be selected, very high performance would be possible [131,124]. The problem is then to identify which technique (and representation) is appropriate for a particular query. Unfortunately, growing evidence shows that this is extremely difficult if not impossible [eg, 132]. The selection of a particular strategy and representation can be guided by rules in consultation with the user. Given such a system, retrieval can be viewed as a form of plausible inference [112] or gathering of evidence about the relevance of documents from various sources.

2.4.6 Architectures and techniques for Integrated Systems

It seems, therefore, that there are strong arguments either for using specific retrieval techniques for specific kinds of queries on the assumption that some techniques are more appropriate for some queries than for others, or for using many retrieval techniques on a single query in the hope that the combination will result in a satisfactory response when no single technique is adequate.

The SIRE system [133] is an example of a system that provides several retrieval techniques that can be used singly or in combination for any specific query. In this system the user chooses the technique. Systems that automatically choose the technique or that have special techniques or data structures associated with specific kinds of queries or problems are more problematic. To complicate the situation further, there are many different user populations with presumably different data needs.

The observations above suggest that a proper paradigm and architecture is necessary for a DS facing the information overload problem. Before dwelling into it therefore, a survey of the search engines and their problems and issues are discussed in the next section.

2.5 SEARCH ENGINES AND THEIR PROBLEMS

The tremendous amount of information available on Internet has made a plethora of search engines to be developed. Table 2-1 shows the details of prominent search engines on the Internet with their unique features and statistics. The table is neither exhaustive nor complete. It is only indicative of the present scenario that exists on the Internet. The search engines use spiders/robots –the specialized software agents- to index the resources available on the Internet. Most of these search engines use either an individual or a combination of the information retrieval techniques discussed above. Some even try to make use of their own heuristics and depend highly on the feedback mechanisms and the user behavior.

As is clear (from the table 2-1), no single search engine has indexed all the available web pages/sites on WWW into their indexes. Apart from the WWW, Internet contains so much of information in the form of files, news-archives in news groups, email-archives in mailing-lists, bulletin boards etc., that no search engine can ever imagine of all the available information to be indexed within its database.

The ease and economy of creating and publishing information (eg., using HTML) has inducted everyone to create and publish the information on the Internet without any review, authenticity, authority, responsibility and the like. Add to this the problem of information creation in one's own native language. Subsequently, Internet was a place where from any one user's point of view -at any given time it- contained noise more than the signal. It is this that has made scientists and researchers even to think of having an altogether new universe called Internet2 (I2).

Scope	AltaVista	Excite	Google	HotBot	Lycos
Content Size	250M pages	250M pages & media objects	1.25 billion sites	110M sites	50M pages
Full-text	Yes	Yes	Yes	Yes	No
Logic					
Default word	OR	OR	AND	AND	and
Phrase search	Quotation marks	Quotations marks	Quotation marks	Quotation marks	Quotation marks
Truncation	No, use *	No	Automatic	No, ? for one right-hand letter; * for left-hand	No
Case sensitive	Yes	No	No	Yes	No
Words included	Use +	Use +	Use +	Use +	Use +
Word elimination	Use -	Use -	Use -	Use -	Use -
Duplicate detection	Grouped under one title	Yes	Grouped Under Categories	Grouped under one title	Yes
Special features	Limit by date, language, or format field followed by a colon	Concept searching suggests terms	Search any language	Limit by date, language, location, page depth	Search for image and sound files

Table 2-1 Comparison of prominent existing search engines on Internet

Further, anyone and everyone could retreat at any time. Consequently, the Internet universe grows and shrinks at an almost unpredictable rate and hence an information seeker, is facing a universe constantly changing and highly dynamic in nature that can hardly ever be estimated. In addition, it is clear that in future, it is not possible to *search* for the information; rather one should *discover* information possibly the one and the only alternative for the information seeker. This leads us to an altogether new paradigm that we term as the Information Discovery Paradigm (IDP) (more details are provided in section 3.2). The IDP presented in [134] focuses on Information Discovery. The importance of information discovery is also discussed in [135]. There are other closely related terms that are at times used in the field for Information Discovery (ID) [136] like Information Retrieval (IR) [137-138] and Information Filtering (IF). For ID, IR and IF classification and categorizing of information is essential. This is analogous to document classification [139] in a library environment.

2.6 CURRENT STATE OF RESEARCH IN THE FIELD

In future, Information Discovery is inevitable. The current issues in network based information resource discovery are clearly presented by Lynch C.A. His work can be considered as the first serious approach in bringing the multitude, multi-dimensional issues bothering information discovery. However, in this paper the term networked information resource discovery is extensively used, which confines it only to discovering the sources of information.

Another important work in the field is an attempt by H.A. Proper and P.D. Bruza in which they stick to the term information discovery but mean only resource discovery with their concept of information carriers. Their work is important in the sense it attempts to devise a new model-theoretic basis for information discovery problem. Their approach to the fundamentals of the information discovery in terms of logic/information-based framework is influential. However, they acknowledge the shortcomings in characterizing the information carrier model. They are right in pointing out a lack of consensus in the field and a need for rigorous and active research in the field of information discovery. Their observation of the interrelationship of ID with various other fields and its user-centered nature is worth noting.

In an attempt to evaluate the state of the art in networked information resource discovery, Lynch has identified three generations in the short span of few decades of time on the research and development of resource discovery systems. These are explained below.

2.6.1 The First Generation: Manually Compiled Directories

The first generation of work consisted primarily of human-generated bibliographies, distributed both as electronic files and as printed works. These days go back to the late 1980's, when the Internet was small enough. One could then imagine compiling, comprehensive directories of networked information resources and updating them perhaps even monthly. Relevant work from this period includes the directories by Art St. George and Ron Larsen, Billy Barron, and BBN (under contract to NSF to provide

Network Information Center Services in the early NSFNET days). These were distributed via the network and updated periodically. The directories were sufficiently short, at least when originally conceived, that only relatively superficial attention was devoted to classification schemes and taxonomies. All of these efforts at directories were overwhelmed by the explosive growth in networked information resources that took place in the early 1990's.

Since the early 1990's, attempts at "living" (or at least frequently updated) comprehensive directories, and particularly those oriented for use as print directories, have been largely abandoned for their technical and economical unfeasibility.

2.6.2 The Second Generation: Data Collection Systems

The second generation of networked information discovery, tools - the first tools that were based primarily on automated information gathering - are best represented by the archie system developed by Peter Deutsch and Alan Emtage [140], [141]. Archie is a "pull-model" system that provides a searchable database of the files in FTP archives. Initially it was simply a system that polled a large number of FTP archives, requesting directory listings, and compiling them into a database. Since then, it has been extensively optimised and refined to use pre-computed listings stored on the FTP sites, to include additional information provided by FTP site operators to replicate the archie database at multiple sites to distribute the search load, and to regionalize the polling process to reduce network bandwidth. Searching in archie database is relatively simple, and the descriptive data is very sparse (basically file name, file size, and date of the last update).

There is no evaluative data, and archie is purely oriented towards discovery as opposed to selection among, candidate resources. The archie experience was central and seminal in identifying a number of requirements, ranging from the need for standard methods of identifying networked resources at various levels of abstraction (implemented through the efforts of the Internet Engineering Task Force (IETF) to standardize Uniform Resource Locators (URL's), Uniform Resource Names (URN's), and Uniform Resource Characteristics (URC's)), to the need to develop more efficient methods of data collection (as explored in the Harvest project, discussed below).

The archie work subsequently served as a model for a number of other networked information discovery services, which helped users navigate deliberately designed information spaces than the rather haphazard world of FTP archives, notably Veronica [142], which constructed a database of all of the menu entries in Gopherspace, and later various Web-walking systems that compiled directories of pages on the WWW [143]. In all of these systems, the basic strategy is the same.

During the time of the second generation, there were a number of research efforts to apply more traditional bibliographic cataloging practices to develop databases of key network resources. Unlike archie and its progeny, the idea here was to apply selection and intellectual cataloging. Examples included research work at OCLC [144] and the TOPNODE project sponsored by the Coalition for Networked Information (CNI) [145]. These efforts have not developed viable operational systems. Other efforts to develop directories based on externally provided "catalog" records continue to emerge. For

example, Government Information Locator Service (GILS) system provides distributed information server architecture, based on the Z39.50 information retrieval protocol, to a set of databases of locator records describing government information resources [146].

The Wide Area Information Server system (WAIS) was also developed around the time of the second-generation systems [147] [148]. WAIS was an open, extensible system for creating instances of a new class of networked information resources as part of its basic architecture in contrast to Gopher or the WWW. In WAIS system capabilities were incorporated as an internal component of the base system architecture. WAIS is, a distributed, client-server based information retrieval system. The interface between the clients and servers was based on an early version of the Z39.50 protocol, with some specialized extensions. However, the WAIS also included a distinguished WAIS database called the directory of servers, which contained entries for each (publicly advertised) WAIS database. These entries were created by the owners of the various databases and were based primarily on free-text database descriptions supplied by the database providers. The most notable aspect in WAIS is the inclusion of the discovery and selection service, at least on an architectural basis. The experience with WAIS was also a part of the motivation for the inclusion of a facility called - EXPLAIN - in the drafts of version 3 of the Z39.50 information retrieval standard in 1994-1995 [149]. EXPLAIN provides a standard way for information servers to incorporate and offer access to a collection of descriptive information about databases housed on the server which could provide a source of input to directories such as a WAIS directory of servers, and also guide direct client interaction with a given information server.

By the time of the second generation, the number of databases available, even as part of monolithic commercial services (such as DIALOG), or to patrons at some research libraries, had grown so huge that selection of appropriate databases was often a greater problem for the user than mastering the mechanics of search syntax. This led to a growing focus on methods for selecting appropriate databases *within* a system. For instance, DIALOG (offering access to numerous databases), or the “reference server” that was part of the architecture of the Carnegie-Mellon University Mercury Library System (intended to help library patrons determine as to which databases to search) [150].

2.6.3 The Third Generation: Distributed intelligent indexing systems

The third generation of networked information discovery and selection systems, such as the Harvest System [151] [152], incorporate significant improvements in the methods used for transferring information from source servers to indexing services. This is done by providing a much more flexible framework for distributed, cooperative computation of indexing terms between indexing services and source servers (see also [153], [154]). Rather than creating a system likearchie, which simply has to fit on top of an installed base of existing FTP sites, Harvest includes the ability to place an “agent” in residence at each source server. This scheme allows the server to participate in the Harvest system in an intelligent way. The architectural model explicitly recognizes the existence of multiple autonomous indexing services and even explores frameworks for cooperation among them.

Schwartz and his colleagues built Harvest on top of his earlier work on a system called Essence [155], which is essentially a sophisticated, heuristic-based, extensible indexing system for files stored on the network. It actually employed heuristics to identify the type of contents that a file contained (e.g., program code, TeX, various word processor formats, and Postscript), and then invoked an appropriate module to attempt to index that class of content intelligently. Lycos is another third generation system [156], developed at Carnegie-Mellon University, which applies complex indexing and analysis methods to WWW pages.

Another type of third generation system that has been deployed filters large numbers of newsgroups, mailing lists, or other information sources according to interest profiles that a user has registered [157]. Stanford University provides an example of such a system [158].

Lynch's classification of the three generations is essentially prior to the era of search engines and the sole interest at that time was just to classify the information resources and not to bother about information itself. As explained in section 2.5 earlier, the search engines have the problem of facing a dynamic environment. The dynamic environment they face is beyond any of the presently available estimation techniques. Nevertheless, Google, the largest among the search engines available has an architecture, which is much inferior to Harvest's architecture (as admitted by [159]). One major drawback with the Harvest's architecture is, it requires a server to be placed at each participating source of information, which is impracticable in Internet. Google's

popularity is *certainly not because of its architecture* but because it can deliver the results expected by a normal user. For a real scalable architecture, and getting the answers to the queries with just sufficient information (not necessarily exhaustive in nature with undesirable garbage in it), to precisely satisfy users information need, *one should work at a granularity that is much smaller*. The whole exercise of planning to have a semantic web [160] by W3C (World Wide Web Consortium) is precisely a move in this direction.

2.7 SUMMARY

Some of the important characteristics of distributed systems and the challenges presented by them for the researchers and scientists were examined. In doing so, the fact that information modelling is an issue in DS has emerged and a review of the same is presented with the necessary focus on information modelling. Modelling must be augmented with suitable techniques for efficient retrieval. Hence, various information retrieval techniques were discussed. Search engines are playing a major role in recent times and a survey of some of them, with the problems they face is therefore covered. The need for *discovering* information rather than *searching* is highlighted leading to the development of an information discovery paradigm. This paradigm is most common in any networked environment, which is meant for information services, whether it is an Internet, Intranet or Extranet. Several ID issues are identified and various solutions proposed/implemented in the largest DS are also discussed with appropriate classification of three generations. It is observed that tackling of issues concerned with architecture

and scaling in a DS environment requires, operating at information granularity much smaller than the existing ones.

CHAPTER 3

ARCHITECTURE OF SOFTWARE SYSTEMS

3.1 INTRODUCTION

Development of large software systems has always faced problems due to several reasons; one among them is the fact that not all systems are built with sound architectural framework. Unless a large software system is engineered by sound architectural basis, it is liable to failure as it is designed and developed by members belonging to different teams, countries and cultural backgrounds. Such large software system architecture needs to be flexible as these systems are built with many years of effort. In the mean time technology might change rapidly. More often it is so in computer science and software engineering. The number of computer architectures and programming languages developed, taught and utilized in the past five decades or so stands testimony for the rapid change in these fields. If a large software system commits to a particular algorithm or a hardware component at the architectural level then that surely will result in the failure of the system. This argument might be extended further to make the technology itself as a component in developing architectures for large software systems. Apparently there has been almost no effort on the part of software system architects to allow technologies themselves to become the components of the system architecture. Rigorous research in this direction is needed if one has to develop any reasonably useful system in future.

Before exploring architecture, information discovery that is essential in the networked environment and its associated issues are presented.

3.2 INFORMATION DISCOVERY PARADIGM

The information space can be near infinite considering the exponential growth of the networks around the world. The collection of such networks, which is now ubiquitous and pervasive in the form of Internet, manifests itself as the largest distributed system. Information discovery therefore, could be within a distributed system. Figure 3-1 shows essentially an Information Discovery Paradigm (IDP) proposed by Srinivas Kini & others [134]. The figure depicts an ocean of information at one end containing the information the discoverer wants -represented by a small white square-, and the discoverer himself at the other end along with information need being conceptualized in his mind. The conceptualized information-need is provided to the information discovery system (IDS) in the form of queries.

The penta-tier architecture of the software, (described in the section 3.4), helps in exploring information from information space available in distributed environment. This process of information exploration is nontrivial with complex activities. These are shown symbolically in the background of the penta-tier architecture in the form of a group of people discussing a point (refer figure 3-1).

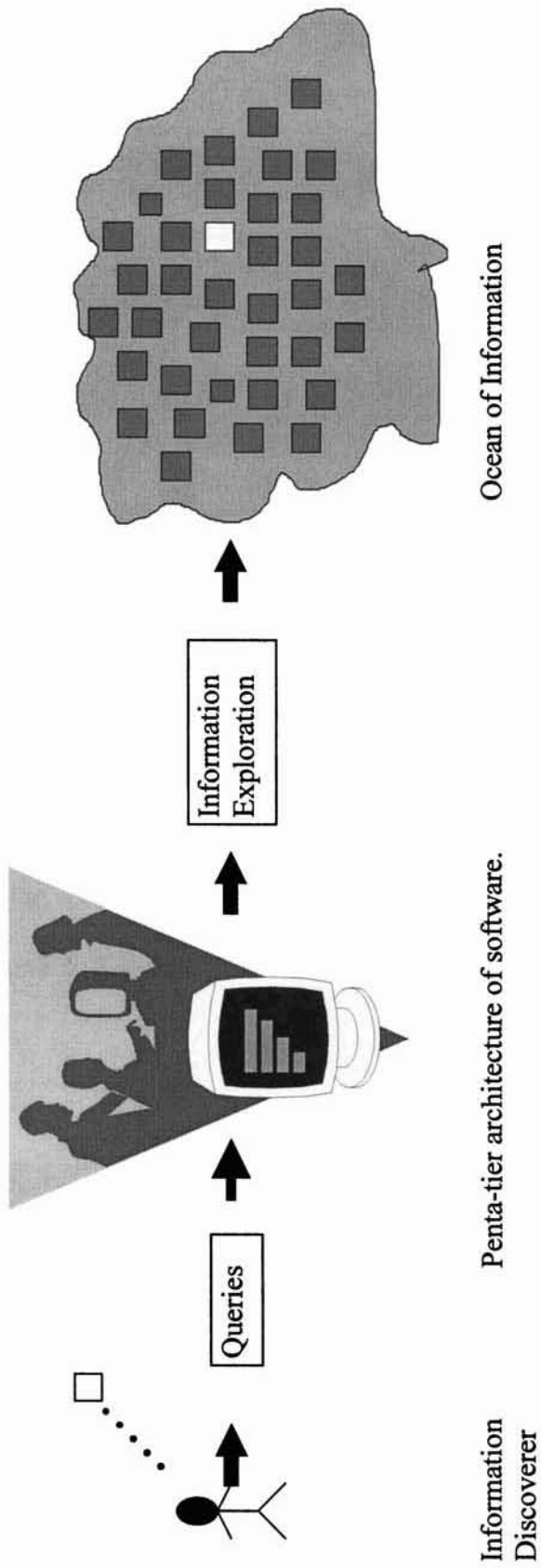


Figure 3-1 Information Discovery Paradigm

In order to facilitate better and quicker response it is essential to organize and classify information space. In order to do this in an efficient and expedient manner the granularity at which the information must be confronted should be the lowest possible. The need has been expressed by researchers and scientists in various fields including but not limited to information science, cognitive psychology, information theory, library science, computer science, signaling theory and the like. This is evident from the growing number of research papers by many authors in recent times that are trying to describe and define information (discussed in chapter 4 in more detail).

Information discovery in a networked environment is inevitable. Though in literature some authors use the term Resource Discovery (RD) to mean the information resource discovery, H.A. Proper & P.D. Bruza have preferred the term information discovery. This is appropriate, as users prefer to discover information of interest to them than just the sources of information. They agree with the definition provided by Lynch for information discovery. The definition for information discovery provided by Lynch is as follows

“a complex collection of activities that can range from simply locating a well-specified digital object on the network through lengthy iterative research activities which involve the identification of a set of potentially relevant networked information resources, the organization and ranking resources in this candidate set, and repeated expansion or restriction of this set based on characteristics of the identified resources and exploration of specific resources”.

This definition of ID clearly distinguishes itself from the “Information Retrieval” a term coined by C.N Moores in 1952 and popularized later by Faithhorne in 1961. The dissimilarity between ID, IR and (IF) is discussed by H.A.Proper & P.D. Bruza. To summarize, the IR term is usually utilized in a more or less restricted (closed) environment whereas ID occurs in a networked, uncontrolled, distributed (open) environment. This openness of the networked environment makes the discovery of information resources (which could be in any form viz., text, images, sound, movie, animated pictures etc.), a nontrivial activity of repetitive nature. The repetitive nature of the problem is mainly due to the following reasons.

First of all, the user of the networked environment (i.e. an information discoverer) may not have a concrete and clear idea of what he/she is trying to discover. This makes them often to reformulate their queries. They assume that the needed information exists somewhere in the information space and there exists a path to reach the corresponding information. Information discoverer views the problem of ID as that of exploring the path, which takes him/her to the information that best suits the conceptual idea they have had. As the users learn increasingly, they become more competent in describing their needs to the IDS.

Secondly, the information can be reached only by applying multiple characteristics one after the other. The biggest problem at times is to know when to stop expansion and based on what criteria to restrict the candidate set. There are a large number of alternatives available and consequently the present Internet has so many

varieties of search engines available that follow different strategies (See section 2.5 above.).

Thus, the ID problem is essentially to identify the right information from a vast information space in any network. The issues identified with the ID problem are:

- Behavior of users (information discoverers).
- Representation of information.
- Refining the documents for more detailed analysis.
- Selection of Information.
- Characterization and categorization of information based on content.

The behavior of the information discoverers is mainly constrained by the time available for them to discover the information. The larger the time available, the more queries can be formulated and in different ways. The reformulation of queries by expansion or restriction of certain characteristics is also possible. The reformulation and multiple queries become a necessity, as it is a nontrivial task to precisely formulate the information need of information discoverer.

Representation of information making them amenable for ID is a nontrivial task and many alternatives have been suggested among which the metadata and new formats (like RDF) are proposed in literature. Documents may have to be refined before really using them in ID context. Refining them for more detailed analysis could be more

effective if the user (information discoverer) in a networked environment has some control over selecting the resources and subjecting them for analysis, based on their content.

Classification of information based on certain characteristics leading to categorization is similar to that of document classification in library environment. The need for a classification and analysis of information sources is pronounced by Ellis David and Vasconcelos Ana.

Keeping in view that the information discovery paradigm is an issue involved with the distributed systems, the experience gained by the design and development of an Election Counting and Reporting Software (E CRS) system -that exemplifies a DS- is considered in detail in this thesis. The following section illustrates design and development of architecture for a DS, which in turn is composed of five different technologies.

3.3 ARCHITECTURE OF DISTRIBUTED SYSTEMS

In order to arrive at an appropriate architecture for a DS, investigations on certain typical application packages were carried out. It was observed that an application relating to the counting of votes in a general election to a parliamentary constituency in India, called Election Counting and Reporting Software (E CRS) system, would fit as an ideal candidate as DS. E CRS system is distributed in nature and covers all the problems discussed earlier in section 2.2. The observations recorded and the experience gained in

developing a monolithic architecture [161], has resulted in an architecture that further evolved into penta-tier architecture. This architecture is presented in the next section.

3.3.1 Background

India is the biggest democracy in the world. The data of recent general elections substantiate this fact. Elections in India generate enormous data. In order to publish the trends and statutory reports and such other details, which would help the district administration in managing the process of elections, it is necessary to compile data from different points within the system.

Elections are held for various constituencies; each constituency consisting of assembly segments. The number of assembly segments and the number of polling booths in each assembly segment are dependent on the population. The election counting process as it was planned and executed at the constituency where this ECRS system was implemented is explained below.

The counting process at Shimoga (a district head quarters in Karnataka State) constituency was carried out according to the following scheme: There were 30 counting tables for each assembly segments (there were 8 such assembly segments), and a total of 240 counting tables were used. The ballot papers are bundled, after mixing them properly from the ballot boxes. Each bundle contains 1000 ballot papers (except the residual ones). Bundles are served to each table. In one round thus 2,40,000 votes would be counted. The total number of rounds of counting depends on the votes to be counted.

The data was fed to the computers with software components specially designed to analyze election data, generate reports, etc., running on different systems. These computers were further networked in order to share the data and information.

3.3.2 System Architecture

The system architecture (refer figure 3-2) contains five distinct components [162]. These five components are identified based on observation and experience in dealing with any DS, which is amenable for ID.

- A. Networked computing machines and Network Operating System.
- B. Data warehouse
- C. Data mining
- D. OLAP
- E. Data visualization

A. Networked computing machines and Network Operating System

It contains the necessary hardware along with the network operating system and the network servers.

B. Data warehouse

The data warehouse is the repository for stored data and metadata. Data warehousing involves the storage and maintenance of the data specific to the domain in question. Any domain contains two types of data - static and dynamic. It is important to distinguish the data in a data warehouse as static and dynamic. It serves two purposes.

First, it helps in determining the state of the data warehouse at any given point of time. Secondly, such a classification of data (into static and dynamic) allows us to concentrate on the data that is to be maintained. Different approaches for classification of data into static and dynamic data can be had, depending on the vulnerability of data, intended usage of data, frequency of update, the source of data and its reliability. Such a classification helps us to increase the overall efficiency of the data warehouse. This also helps in building fault tolerance into the system [163].

It is observed that most of the data in the data warehouse would be static. Almost all data enters in to domain as dynamic data except in the initial phase of creating the data warehouse. However in the initial phase of the creation of data warehouse, it is left to the designer of the data warehouse to choose which data will act as static and which data will be dynamic. The importance of the need for such a distinction between the data items calls for a lot of attention of the designer. A failure at this point by the designer to classify data properly results in degraded performance of the data warehouse. In addition, mistakes committed can result at worst in total failure of the system to respond to the user queries later.

Slowly over a period of time, depending on the domain, dynamic data becomes static. It is an important issue for the architect of data warehouse to determine at what point of time a dynamic data becomes static. This is where the designer of the data warehouse must be careful. Fortunately most of the times the domain in question itself

will provide the answer to this question. In ECRS system, for instance, each and every round of counting data becomes static, once the competent authority approves the data.

Another way to classify data in a data warehouse as static and dynamic depends on the purpose of the usage of the data. For instance, historical data in a data warehouse constitutes static data and the metadata in a data warehouse constitute dynamic data. Since the concept of static, dynamic and supportive data is important for ID, it is discussed below.

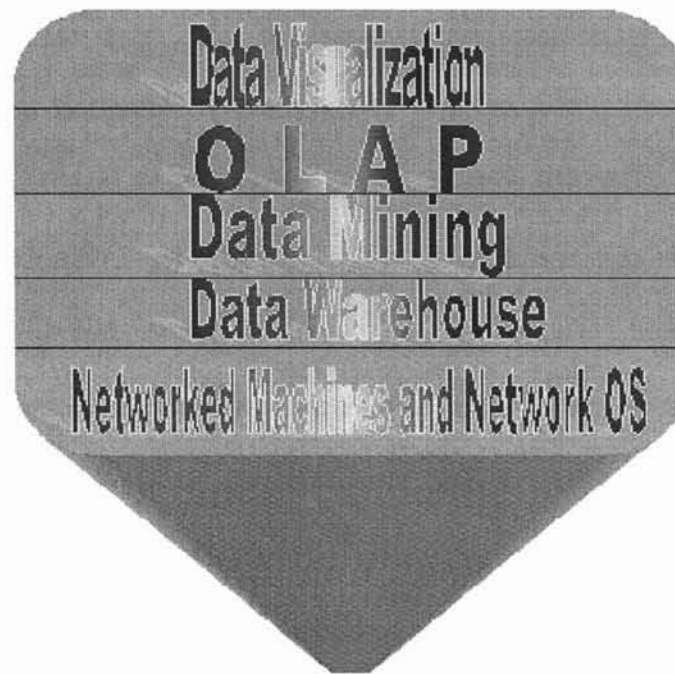


Figure 3-2 A System Architecture with five complementary technologies

Static (Historical) Data

Populating static data contains the following steps.

- Data Collection.
- Data preparation and validation.

Data Collection

In Shimoga constituency the general elections were held from 1952 onwards. All the data related to election results were in the form of paper documents, which had to be converted into electronic form. This involved a lot of interaction with the staff concerned for collecting all the data intended to be placed in the data warehouse. During discussions it was decided to maintain only records of the candidates and political parties who have secured more than 50,000 votes. Such a decision is important irrespective of the domain. Otherwise the data warehouse may be containing a lot of junk, which is not useful. For instance, consider an independent candidate who contests only once in an election and secures less than 50,000 votes. These records need not be maintained. In this way system user determines the decision as to which data is to be maintained in the data warehouse. This is so as the user is the one who wants to query on the data warehouse and generally more competent in domain knowledge. But the designer has to educate the user of the consequences of storing the data, which may not be very useful in the long run and is negligible in its magnitude to disturb the final results expected by the system.

Data preparation and Validation

The data so collected was entered into suitable place within the data warehouse. Automatic validation checks were performed on this data at the time of data entry. Only the “clean data” was allowed to enter the data warehouse. A strict and stringent policy has to be developed to make sure the data is clean. One must remember that query results are directly affected by the cleanliness of the data warehouse. It is not only necessary to see to it that one gets the clean data in the data warehouse, but also to maintain the data warehouse itself clean is an issue that needs to be emphasized.

Dynamic (Election Counting) Data

In a special questionnaire form, the data was collected from each election vote counting table. After entering the details of each counted bundle of ballot papers, these special forms were duly signed by the authorized person. Once these special forms are signed, they arrive at the desk of the computer data entry operator. Operator enters all the details in the form including the assembly segment code and the counting table code into the computer. Cross verification of the data entered by the data entry operator can be automatically done with already available information in the system⁴ about the assembly segment. The data about the number of ballot papers in one bundle fed earlier was also used to check the entered data. In addition, the total number of votes counted and categorized as belonging to different candidates was then summed up and it was verified with the other data maintained by the system. Thus even before populating data into the

⁴ The historical data and other static data that represent the assembly segment could be used to support validation and verification checks.

data warehouse, cleanliness of data entering the data warehouse afresh was doubly ensured by above procedure. In case of any discrepancies, they were handled in an intelligent way. Discrepancies were also logged into a special log file. System even drew the attention of data entry operator, who was instructed to report the matter to his immediate supervisor to verify and take proper action. The above procedure was followed to make sure the quality of the data and data consistency are not compromised at any cost, which might otherwise result in wrong interpretations by the software used for its analysis later. It is advocated and emphasized that preserving data consistency and integrity should be invariably done as it plays a vital role in the overall success of the system.

Supportive (Meta) data

Metadata basically refers to the data of data. When the size of the data warehouse is extremely large, metadata helps in retrieving the data in the data warehouse in an easy and faster manner. Since metadata has to reflect the current state of the data warehouse, it can be treated as dynamic data. This dynamic nature of metadata warrants at times replicating the information on metadata. Such replication of metadata provides fault tolerance to the data warehouse. Replication imposes additional overhead to the system but it is worth carrying this out.

C. Data mining

Data mining is an attractive technology for users looking to make sense out of a large volume of complex data. The fact that programs can analyze an entire data warehouse and identify the key relationships relevant to the users is interesting for all data analysts. Data mining uses algorithms employed to determine the key relationships in the data. Data mining can look at numerous multidimensional data relationships, highlighting those that are exceptional.

Different approaches that are available here include sequence-based analysis, clustering, classification, estimation and various other techniques. In ECRS system however simple IR techniques that are mentioned in section 2.4 earlier are used in addition to some of the simple statistical analysis on the data.

D. OLAP

OnLine Analytical Processing (OLAP) helps in providing the kind of information the end user is looking for, by making use of all of the data that is available in the data warehouse at the instance of query execution. The end user requirements can be implemented in the form of canned query. If an a priori knowledge of what the end user is looking for can be heuristically guessed, then the data warehouse can be tuned to meet the request of the user in the fastest possible way by using appropriate algorithms (refer section 2.4.5). In ECRS system, the online analytical processing was employed in trend finding, comparison of the new results with that of the old ones (by party, by candidate,

by assembly segments etc.). OLAP tools, process the user query for instantaneously providing the results from the data warehouse. The analysis can vary from simple aggregation, to complex modelling of the data. The OLAP tools can add some metadata to the data warehouse to facilitate instantaneous results to the queries anticipated. At this layer of the architecture, the interface with the end user must be capable of generating query results quickly [162].

E Data Visualization

“A picture is worth a thousand words” – end users would like to view the results in the form of graphs, charts, pictures and animations. The reason for such a demand from the user community is two fold. First of all the user is not having interest in seeing just numbers in the form of tables. Secondly the user’s time is precious and they can efficiently and effectively analyze the results by saving time, if the results are in a more convenient visual form. Rather than simply creating statistical printouts of the results, sophisticated online visualization for end users is appreciated. Features like the ability to display graphics dynamically, show related 3-D graphs on the screen, and rotate graphs so that they can be viewed from different perspectives can be therefore included here. Basically this layer acts as the presentation layer for the system.

To conclude, a system developed by following the above architecture is far more informative than simple tables that represent the same information. Comparison of the results of older elections with the current one and that too using special-effect visualization was well appreciated in informed quarters. One important attracting feature

of the ECRS system was to predict the result of the elections from the point when the second round of counting results were entered into the data warehouse. The technologies blended as discussed in the architecture are needed for full exploitation of the data warehouse and OLAP.

The concept of static and dynamic data becomes handy in creation and maintenance of the data warehouse. It also brings in the fault tolerance and reliability to the data warehouse, which is an important aspect for any system.

The visualization of the data helps users to understand the interrelationships among the data. This in turn helps them to get rid of conflicts and makes their understanding clearer by saving their precious time. The architecture of complementary technologies helps in making strategic decisions by the users themselves without botheration of how and where the data is stored.

All layers in the system architecture are complementary to the other adjacent layers. The five complementary technologies help in developing and deploying a better system. Since most of the organizations already have networked their computing machines for proper sharing of data and information, the lowest layer in the architecture is already present. What is required is the creation of a data warehouse, proper selection of data mining technique for the specific domain, using OLAP for necessary processing, and finally the data visualization for presenting the results to the user. Five complementary layers in the architecture can therefore be filled up with appropriate

subsystems within the distributed system. Each group of subsystems if assigned for a corresponding layer in the architecture, resulting in a tier, would end up in an architecture comprising five tiers that can conveniently be called penta-tier architecture.

This approach has mainly following advantages

- If any technology evolves, all the advantages it encompasses may be carried into the final system instantaneously.
- If any component (basic/composite) is found to be unreliable, faulty, and inefficient, due to underlying technology used in the development and/or deployment of that primitive basic component and/or composite derived components, it is easy and safe to revert back to previously used components, which were reliable, correct and efficient with least/no effect on the system.

3.4 ECRS SYSTEM ARCHITECTURE

The monolithic architecture that was used in a version earlier to ECRS system contained modules listed in table 3-1. Any single fault occurring in it could bring down the system to a grinding halt. This is due to the architecture being fragile. In monolithic architecture, modules were essentially developed and deployed with the assumption that the integrity of the database is intact. This assumption may not always be true. In a complex environment that any DS is supposed to function, all sorts of errors and faults are possible including a compromise of database integrity. The deficiency of the monolithic architecture was its fragileness with respect to these errors and faults.

Sl.No	Module Name	Module Function
1	Get_Data	To get the data from the user and populate database.
2	Update_Data	To modify the existing data in the database either using a programming interface or directly by tampering the database.
3	Analyse	Make analysis of Election database with appropriate algorithms, which includes statistical analysis.
4	Display	Display the information in the form of graphs, charts and tables making it attractive, appealing and easy to understand for the end user.
5	PrepareForDisplay	In order to display the information properly consolidating, classifying and assimilating related information may have to be done and this module does that.

Table 3-1 Modules used in monolithic architecture

The modules listed in table 3-1 were put in appropriate layers in the penta-tier architecture as shown in figure 3-3. Thus ECRS system is made to conform to the penta-tier architecture. This architecture is conveniently called “ECRS system architecture”. Further wrappers around each of these modules were developed to efficiently and effectively identify and block faults. These wrappers were designed in such a way that they can identify both internal and external faults, especially at the interface (layer) boundaries. In situations where the fault is detected two alternatives exist. In the first one, either the system on its own, changes system parameters (thus continues to function as if it is normal) or user is given a chance to review and restart computing from the previous layer with appropriate changes in the system parameters. In the other case a message is flashed mentioning the inability to continue further and the culprit module and/or data is identified and reported.



Figure 3-3 ECRS System Architecture.

An experiment was conducted to demonstrate that this layered approach makes architecture fault tolerant. In this experiment two types of faults (internal and external) were injected and the behavior with respect to these faults was compared between the monolithic architecture and the penta-tier architecture as applied in the modified ECRS system. The faults injected are carefully selected. This is done by keeping in mind that such faults should naturally be expected in a distributed system. The table 3-2 gives the description of the faults injected, their classification as internal and external and under which guise they may appear in a distributed system.

Sl.No	Description of Faults	Probable guise the faults may take in a DS environment	Classification of faults as internal/external
1	Database values changed to from positive to negative where it was expected to be positive always (Unexpected Value)	During transmission over network due to faulty network components that outside the boundaries of the system. Due to malfunctioning hardware under strained environmental conditions.	External
2	Out of range values given as input for modules. (Out of Bound)	Buggy software modules, which do not maintain a strict check on the data taken as input and also those software modules that may not verify their outputs and/or results for range confirmation.	Internal
3	Not null data fields containing Null data. (Unanticipated Data)	Integrity of the database is compromised and/or loss of data due to malfunctioning or faulty hardware.	External
4	Wrong type of input received like for instance just an integer in the place of a float. (Type Mismatch)	Faulty hardware, sensors and transducers that feed data to the modules and are internal to the system. Other software modules of the system that return wrong type of data.	Internal

Table 3-2 Faults injected, their description and classification.

At each instruction, by observing the snapshot of the system and its parameters, propagation of faults through each module in the system is monitored. This is a sort of glass box testing. The permeability of faults through each module is observed in both the cases. It is observed that faults could not permeate beyond the layer boundaries in the

case of ECRS system architecture. Further, the fault was either identified as belonging to a module or an intervention from the user is sought (in extreme cases only).

For each fault injected, the fault is monitored for its movement in the system, especially across the boundaries of the layers and a measure of how many modules they could permeate (without the fault being detected) is recorded. The permeability is inversely proportional to the reliability and fault tolerance. This means more the value of this measure worse is the systems reliability and/or fault tolerance capability. In the experiment this measure is calculated using the formula p/N where p is the number of modules permeated and N is the total number of modules in the system (5 in this experiment as is clear from table 3-1). In the table 3-3 given below in monolithic architecture for two of the faults, the number of modules permeated was 0 (zero) but the system crashed (exhibiting its fragileness and non fault tolerance), while for all the cases of faults in ECRS system architecture the measure was 0 (zero). Neither the system stopped functioning normally nor did it crash either. In the case of fault 1 (i.e., Unexpected Value fault) the system changed the parameters by itself to expected values (changing negative values to positive) and continued to function normally (bringing in fault tolerance to the system). In the case of fault 2 (i.e. Out of Bound fault), system changed the system parameters by itself and continued to function normally by bringing the values of parameters within the expected range. While in the case of Unanticipated Data and Type Mismatch faults (fault 3 and 4 respectively) system asked for user intervention (bringing in fault avoidance feature).

Type of Faults	Type of Faults	Classes of faults injected (refer table 3-2.)	Number of modules Permeated	% Permeability	Overall %
Monolithic Architecture	Internal	2 (Out of Bound)	5	100%	50%
		4 (Type Mismatch)	0 (System Crashed)	0%	
	External	1 (Unexpected Value)	5	100%	
		3 (Unanticipated Data)	0 (System Crashed)	0%	
ECRS System architecture following penta-tier model	Internal	2 (Out of Bound)	0	0%	0%
		4 (Type Mismatch)	0	0%	
	External	1 (Unexpected Value)	0	0%	
		3 (Unanticipated Data)	0	0%	

Table 3-3 Summarised results of the fault tolerance experiment

Table 3-3 shows the summarized results of the experiment. This table assumes that permeability through modules, for each module in the system is equally difficult. Based on this assumption the entries in the percentage columns (column 5 and 6 respectively) in the above table are derived.

It can be concluded from this experiment and the results in the tables that -

“A layered architecture if designed for handling faults with appropriate measures like wrappers can be fault tolerant and more reliable than a monolithic architecture.”

3.5 SUMMARY

This chapter has viewed the architectural issues concerned with any DS. Some important concepts like static, dynamic and supportive data are discussed. They constitute the basis for the necessity of a better architectural framework. It is illustrated that five complementary technologies can be blended together resulting in penta-tier architecture that is fault tolerant and hence reliable. The fault tolerance of the penta-tier architecture was justified with an experiment conducted using two versions of the ECRS system. A more detailed analysis of information, including its definition can support researchers in diverse fields. In the next chapter we shall throw some light on these issues.

CHAPTER 4

INFOTRONS AND IDS DESIGN

4.1 INTRODUCTION

Architecture as developed in the previous chapter provides a framework flexible enough to design any DS. But this architecture does not convey anything about the granularity incorporated within a DS at each tier. Most of the literature considers the document itself as of lowest granularity and attempts to solve the resource discovery (instead of information discovery) problem (refer section 2.7 for details). In resource discovery the major focus is on discovering the sources of information (viz., URI, URN, URL etc.) that points to interrelated collection of documents, as opposed to the information itself. This is the major reason why search engines (discussed in section 2.5 and 2.7) only attempt to produce a list of URLs of documents. The URLs so extracted are the links to the documents that might contain any combination of keywords the user has supplied in his query. For most of the authors, a document means a file stored within a computer that is accessible in a networked environment. In reality, the document is an abstract entity, which is the logical organization of the information that is intended to be conveyed. Hence it is always possible (theoretically) to construct a single document from two or more related documents by merging them logically (practically in case of documents being accessible in digital form). But this exercise of merging must be done

in such a way that the information intended to be conveyed stands out clearly. Effective merging of documents is possible if we can work with them, logically (and even physically) at granularity lesser than that of a document itself. Recently many authors have attempted to describe and define information, its characteristics and the like. Nobody seems to agree with the conventional definition of “information as processed data”. The lack of completeness in this definition augmented with the frequent usage of the terminology “Information Technology” has made many definitions to appear in the literature [164-168].

Defining information is essential for anybody dealing with information discovery. An attempt is made in filling the crevice in section 4.3. Even before dealing with the definition of information, it would be appropriate to ponder over the situation that existed when information theory originated and the situation that prevails now.

4.2 INFORMATION THEORY

Influenced by the thinking of Nyquist H [169,170], (who was more concerned with telegraph speed), Claude E Shannon [171] developed his mathematical theory of communication built around the schematic diagram of a general communication system (refer [171] for a schematic diagram of general communication system).

Scenario 4-1 describes the problems encountered by Shannon and others. This scenario also helps in justifying the reasoning for the way in which he modelled his general theory of communication systems. However, with the growth in technology and

the Internet, augmented with penetration of WWW, not just the rule of the game but the game itself has changed. The scenario 4-2 better holds for the Internet *user* browsing the WWW. One can also consider intelligent software agents or spiders crawling the web for information in lieu of Internet user.

Scenario	SENDING A TELEGRAM.
Objective :	A person wants to send a Telegram from Cochin ⁵ to his relative in Shimoga.
Method	He ⁶ goes to a nearby telegraph office and fills a form with his message and the address of his relative. After paying necessary amount the message will be passed on to his relative using certain technology that encodes his message (Morse Code) as a telegram and is delivered to the address given.
Action :	Initiated by the information source. <i>Information source is proactive.</i>
Result	Information moves from the source to the destination as is depicted by the schematic diagram of a general theory of communication system of Shannon.

Scenario 4-1 Sending a Telegram

The scenario 4-1 and 4-2 would help to highlight the major difference between Shannon's modelling of source and sink of information to the one that is prevailing on the Internet. The following paragraphs summarize the observation.

Scenario 4-1 clearly demonstrates that in Shannon's model, *information source* was *proactive* whereas going by Scenario 4-2 *information destination* is *proactive*.

⁵ Cochin and Shimoga are two cities in Kerala and Karnataka State of India respectively.

⁶ Read He/She as is convenient.

In scenario 4-2, the user⁷ (information sink) is proactive; in the sense, he guides the browser (a tool to navigate Internet). This change in scenario leads to a remodelling of the schematic diagram of a general communication system, resulting in the schematic diagram of a general information discovery system. The schematic diagram of a general information discovery system is shown in figure 4-1. In figure 4-1, the information discoverer is shown on the leftmost box and set of information repository is represented on the rightmost box. Note the *direction* of information flow.

Scenario	: DRINKING TENDER COCONUT ⁸
Objective	: A person wants to drink a tender coconut to quench his thirst.
Method	: He goes to a nearby shop where a tender coconut is available and gets it cut so that he can use a straw and drink it. He sips the contents of the tender coconut slowly using the straw. The tender coconut water moves from the shell to him. In fact, the person draws the tender coconut water.
Action	: Initiated by the destination. Information moves or is drawn by the destination from the source. <i>Information destination is proactive.</i>
Result	: Information is drawn by destination from source.

Scenario 4-2 Drinking tender coconut

⁷A discoverer can be positioned in place of Internet user broadening the scope of the information discovery problem into a realistic (non-virtual) world where discoverer is proactive. For instance, any researcher can be thought as a discoverer.

⁸One can think of any of the favourite fruit juice or softdrink.

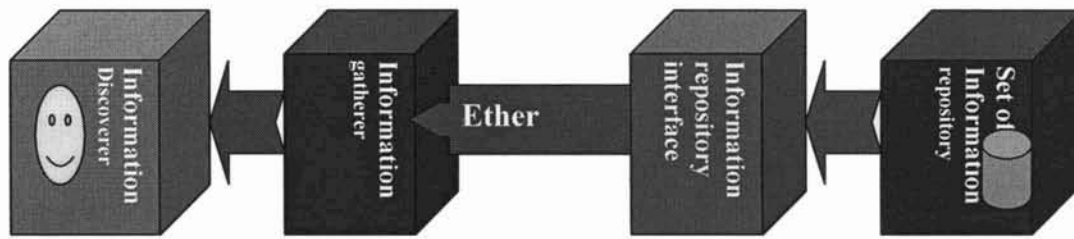


Figure 4-1 Schematic diagram of general information discovery system

The schematic diagram of general information discovery system is strikingly similar to that of information model continuum. The earlier discussion on information model continuum (refer figure 2-1 under section 2.3.2) had the system on one side and the user on the other side (as in figure 4-1) and various models of information being available in the range classified as cognitive, conceptual, and semantic models. *A single model that can cater to the needs of both the user and the system and cruise along the whole spectrum of continuum would be the most appropriate one.* It is therefore held that there is an urgent need to specifically visualize a model along with all the corollary data and be motivated to design an objective machinery for this purpose. This is achieved by dwelling on and developing an adequate concept of infotron. The development of infotron could thus be treated as an active search for the smallest possible, fundamental information element.

4.3 INFOTRONS

Infotron is the smallest possible, fundamental information particle. Infotron is to information just as an electron is for an atom. The notion of infotron further helps to define information. Unlike electrons, however, infotrons do not have mass, charge, or

energy. They do not have to revolve around any other particles neither do they exhibit spin. The comparison made to electrons is mainly to signify the behavior of free electrons that have the capability to carry charges, resulting in the current flow. In a similar way, infotrons will carry information from source to sink resulting in the information flow. It is worth noting at this point that information by its nature *flows*.

When the infotrons are gathered by the Information sink made available by the source by means of a language L governed by grammar G , it results in the *flow of information*. A language and its characterization have been covered in much detail in [172,173].

4.4 DEFINITION OF INFORMATION

Consider a set of infotrons represented by the set ι given by

$$(1) \quad \iota = \{a, b, c, \dots\}$$

Where, a, b, c, \dots are infotrons. This set is called the *set of infotrons* and is defined for a language L with grammar G

Then *Information* $z \in I$ can be defined as the result of an operator $\theta \in \Phi$ on elements $x_1, x_2, x_3, \dots \in I \cup \iota$ as governed by the language L and its grammar G .

In mathematical terms, $\forall z \in I$,

$$(2) \quad z = \theta \{x_1, x_2, x_3, \dots\}$$

and $\theta \in \Phi$ where Φ is the set of all operators and is an attribute of G for a given language L .

$$(3) \quad \Phi = \{\theta_1, \theta_2, \theta_3, \dots\}$$

All operators in set Φ are operable on the elements of the set $I \cup \iota$.

As a byproduct -also called side effect in programming languages [174]- of the operation the operator θ might generate additional infotrons/information.

Hence, information is dependent on a given language and its grammar. It means that if someone wants to measure how much information has been gathered then that should be done explicitly by mentioning the language L and its grammar G .

For instance, consider the binary language L_B . In binary language, 0 and 1 are the infotrons available.

$$(4) \quad \therefore \iota = \{0, 1\}$$

$$(5) \quad \Phi = \{\wedge, \vee, ', \dots\}$$

A proper combination of these as defined by the grammar of the binary language, can lead to information in binary language. Note the information can also be *allowed* by the rules of the grammar to take part in forming information. Nevertheless the fundamental constituents are still infotrons.

To elaborate more, the grammar G for binary language consists of three operations (viz., NOT ($'$), AND (\wedge) and OR (\vee)) defined by the truth tables (refer figure A-1 to A-3 in appendix A). All other abstract operations -binary as well as unary operations like addition, subtraction, multiplication, and division- can be realized by fundamental operations on the infotrons. There are in fact sixteen possible operations on any two variables of type infotrons in binary language, which are detailed in appendix A.

From figure A-4 (refer appendix A), it is evident that there are six non-similar operators of which three (i.e., $f = 0$, $f = 1$, $f = x$) are trivial ones while $f = x \wedge y$, $f = x \vee y$, and $f = x \wedge y \vee x' \wedge y'$ are non-trivial ones. All other functions can be obtained from these six by complementation or by interchange of variables. Two more operations called NAND and NOR operations are sufficient to realize even the three fundamental operations and hence they are called *universal operations*.

Shannon's general communication system works with the basic underlying assumption that the source and sink transmit and receive messages (through intermediary

channel) that is composed of symbols for successful communication. In this general model there is *an implied agreement between the source and sink* about the nature and characteristics of the symbol. Without such an agreement, *there could perhaps never be a successful communication as it would be unfeasible for the sink to separate the noise from the signal*. Most of the times, in telecommunication equipments, these semantics of the symbols (signals) carried within the channel and interpreted by the source and sink (which are designed by using special algorithms and protocols that could either be implemented in hardware, firmware or software) are in place. To achieve the same implied agreement in the ID context that uses infotrons as its base, a framework is needed and such a framework is termed as infotron dictionary.

4.5 INFOTRON DICTIONARY

An infotron dictionary is a repository for infotrons and captures their association, relationship with other infotrons and/or information. It also comprises of constraints and rules imposed by operations that are within the framework of a language and a grammar.

Actually visualization of an infotron dictionary is the corollary to the concept of infotron itself. Thus they act as a linking device between information discoverer and the information contained within documents. The association, relationship, operations allowed with their preconditions, etc., between infotrons, that is to be captured, depend mainly on the application domain as well as the agreed upon language and grammar for ID. For instance, in the library information system application domain, considering

“book” as an infotron, its association with other infotrons like “person”, “member”, and the like is controlled by the operation of “borrow”ing book from the library by a “person” (who should of course be a “member”- a precondition - for the operation) for “read”ing (another operation). Instead, on considering the “audio tape” as another infotron (“audio tape” is not a single word but is still considered as infotron in an ID context as opposed to the notion of terms employed in IR), the “member” can “listen” and “play” (operations) but cannot “read”. If we can capture this association and relationship between the infotrons along with the facts like a “movie tape” can produce “sound” and hence can be “play”ed and “listen”ed like an audio tape apart from “viewing” which can also be done for a “book”. This would potentially help a discoverer in articulating the query and gleaning for infotrons that are capable of being “played”.

There is a subtle difference between the concept of infotron dictionary and the object oriented concepts. One could consider a “book” in the preceding example as a class of objects with “borrowing” and “reading” as its member functions and the interaction between various other objects belonging to different class like “member” “person” etc., being considered as message passing events that trigger possibly other operations not belonging to any of these objects. The object oriented analysis and design is intended and indeed it helps in writing programs in a much better way. Unfortunately this is achieved by compromising at various levels like making a member function belonging to any one of the several available classes in the design and further making access to that member function possible by either deriving it through inheritance, or other

mechanisms like “friend” classes and functions. *This compromise makes it inadequate to model the reality with which ID has to survive in dealing with information.* Moreover the focus in object oriented analysis and design is mainly to classify objects identified in the systems and make them amicable in the deliverable system using an appropriate data structure to represent objects. They serve this purpose extremely well. They do provide a better abstraction for data with the emphasis having been shifted more towards data than procedure as opposed to the case of procedure oriented approaches adopted by others. *This abstraction is still insufficient to provide data the freedom to function as an infotron, which is what an infotron dictionary would strive to provide.*

4.6 IDENTIFYING INFOTRONS IN A DOMAIN AND DEVISING AN INFOTRON DICTIONARY

Let us consider the library domain to identify infotrons and create an infotron set. If one considers the “book” as an infotron, in the library context, identifying other infotrons in the domain becomes easier. The book has an “author” which forms the next possible infotron in the set. Once a few key infotrons in a given domain are identified, an infotron set can be formed. A partial set is given below for reference. It is not just sufficient if the infotrons are identified, the operators within the domain must also be identified. The formation of the operator set along with infotron set is governed by the language and grammar of the domain. Such an operator set is also given below.

$$\iota = \{\text{"book"}, \text{"author"}, \text{"person"}, \text{"member"}, \text{"audio tape"}, \text{"video tape"}, \\ \text{"cd"}, \text{"journal"}, \text{"place"}, \text{"price"}, \text{"book shelf"}, \text{"publisher"}, \\ \text{"sound"}, \text{"supplier"}, \dots \}$$
$$\Phi = \{\text{"borrow"}, \text{"read"}, \text{"listen"}, \text{"play"}, \text{"view"}, \text{"pay"}, \text{"return"}, \\ \text{"arrange"}, \text{"refer"}, \text{"pick"}, \text{"write"} \dots \}$$

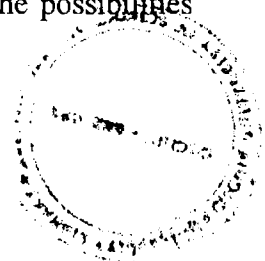
There are instances when infotrons refer to a group of words or terms. For example “audio tape” can become an infotron, if it needs to be maintained (just as books are maintained) in a library. The concept that comes to one’s mind when the term “video tape” is encountered is distinct from “audio tape” even though the word tape is common. As a clue the designer can look for nouns and objects having physical existence to identify infotrons in any domain. The identification of operators follows the identification of infotrons. Again as a clue, designer can look forward for verbs to help identify operators.

Once the infotrons and the operators are identified, capturing their association and interactions in the domain in an appropriate manner for discovery purposes has to be done. This forms the infotron dictionary. A typical infotron dictionary is depicted in table 4-1.

Sl. No	Infotron	Operator applicable	Assigned Weight	Related to other Infotrons	Type of representation within computer	Range of values infotron can take	...
01	Book	01,02, 06, 08, 13,...	08	12,16, 04, 15, 43,...	Text	Any alphabet allowed by the language	...
02	Author	11,02,09,07, ...	25	01,12,07,06 ,09,...	Text	Any alphabet allowed by the language	...
03	Person	02,06,09, 08, ...	30	09,18,23,45, 26,35,22,...	Text	Any alphabet allowed by the language	...
04	Member	02,06, 09, 08, ...	10	01,02,03,09, 18,23,45,...	Text	Any alphabet allowed by the language	...
05	Audio tape	03,04,01,06, 07, ...	18	04,14,13,12, ...	Picture	Not more than 200 pixels	...
06

Table 4-1 A sample infotron dictionary for Library

Infotrons being conceptual entities need some physical representation to be of any use to design and develop an operational system. One of the possible representations could be, for instance, as elements within markup tags used in markup languages like HTML and XML. Another alternative available is to use attribute-value pairs to represent the infotrons. The designer can develop any innovative technique to identify and represent infotrons to suit the domain of the problem he is interested just as the concept of objects in Object Oriented Analysis and Design (OOAD). The possibilities



are limited only by the intuition of the designer himself. In this work however, an approach to the problem by utilizing the elements available within markup language has been followed.

First an attempt to incorporate infotron –the smallest granularity of information– into the documents is made. The study conducted used two hundred documents each in HTML and XML. The focus of study was primarily on identifying which one between HTML and XML would be more suitable for incorporating the infotrons. Next, we provide specification, design and developmental aspects of IDS having the characteristics similar to the one discussed in section 4.2 (refer figure 4-1). Finally, the IDS so developed should be capable of utilization in an application. In order to demonstrate the utilization of the IDS, a legacy system is used. Each of these aspects, are elaborated in following sections.

4.7 AN EMPIRICAL STUDY FOR INCORPORATING INFOTRONS

The empirical study conducted had two hundred documents hand-coded and fine-tuned in accordance with appropriate specification [175-176]. The specification for the latest version of HTML and XML are available from the World Wide Web Consortium (W3C). Two partial sample files one each from HTML and XML are shown in listings 4-1 and 4-2 respectively. The listings illustrate how to incorporate infotrons within the documents.

```

<HTML>
<HEAD>
  <TITLE>A library information file</TITLE>
</HEAD>
<BODY>
  <DIV CLASS="DOCUMENT">
    <DIV CLASS="DOCUMENT.NUMBER"><P>1</P></DIV>
    <DIV CLASS="DOCUMENT.TITLE"><P> A Computer Book File </P></DIV>
    <DIV CLASS="CONTENTS">
      <DIV CLASS="BOOKS">
        <DIV CLASS="COMPUTER.BOOKS">
          <DIV CLASS="BOOK">
            <DIV CLASS="BOOK.TITLE">
              <H2>Developing XML Solutions</H2>
            </DIV>
            <DIV CLASS="AUTHORS">
              <DIV CLASS="AUTHOR">
                <DIV CLASS="FIRST.NAME"><P>STURM</P></DIV>
                <DIV CLASS="MIDDLE.NAME"><P></P></DIV>
                <DIV CLASS="LAST.NAME"><P>JAKE</P></DIV>
              </DIV>
            </DIV>
            <DIV CLASS="PUBLISHER">
              <P>WP Publishers and Distributers Pvt Ltd</P>
            </DIV>
            <DIV CLASS="YEAR"><P>2001</P></DIV>
            <DIV CLASS="ACCESSION.NUMBER"><P>3473</P></DIV>
            <DIV CLASS="ISBN"><P>81-7853-014-7</P></DIV>
            <DIV CLASS="CURRENCY"><P>US $</P></DIV>
            <DIV CLASS="PRICE"><P>49.99</P></DIV>
            <DIV CLASS="SUPPLIER"> <P>Mind storm books center</P></DIV>
            <DIV CLASS="SHELF.MARK"><P>101</P></DIV>
            <DIV CLASS="EDITION"><P> First</P></DIV>
            <DIV CLASS="PAGES"><P>450</P></DIV>
          </DIV>
          ...
        </DIV>
      </DIV>
    </DIV>
  </BODY>

```

Listing 4-1 A sample partial listing of HTML document containing infotrons

```

<?xml version="1.0" ?>
<document>
  <document.number>1</document.number>
  <document.title>A Computer Book File</document.title>
  <contents>
    <books>
      <computer.books>
        <book>
          <book.title>Developing XML Solutions</book.title>
          <author>
            <first.name>STURM</first.name>
            <middle.name />
            <last.name>JAKE</last.name>
          </author>
          <publisher>WP Publishers and Distributers Pvt Ltd</publisher>
          <year>2001</year>
          <accession.number>3473</accession.number>
          <isbn>81-7853-014-7</isbn>
          <currency>US $</currency>
          <price>49.99</price>
          <supplier>Mind storm books center</supplier>
          <shelf.mark>101</shelf.mark>
          <edition>First</edition>
          <pages>450</pages>
        </book>
        <book>
          <book.title>Teach yourself XML in 21 days</book.title>
          <authors>
            <author>
              <first.name>NORTH</first.name>
              <middle.name />
              <last.name>SIMON</last.name>
            </author>
            <author>
              <first.name>HERMANS</first.name>
              <last.name>PAUL</last.name>
            </author>
          </authors>
          <publisher>Sams Techmedia Books</publisher>
          <year>1999</year>
          <accession.number>12345</accession.number>
          <isbn>81-7635-268-3</isbn>
          <currency>Rs.</currency>
          <price>150</price>
          <supplier>Sapna Book House</supplier>
          <shelf.mark>100</shelf.mark>
          <edition>First</edition>
          <pages>600</pages>
        </book>
        ...
      </computer.books>
    </books>
  </contents>
</document>

```

Listing 4-2 A sample partial listing of XML document containing infotrons

Description	XML 1.0	HTML 4	Correct in XML 1.0	Correct in HTML 4
Documents must be well-formed	All elements must either have closing tags or be written in a special form and that all the elements must nest.	Not Necessary	nested elements: <p> here is a bold para .</p>	overlapping elements: <p> here is a bold para </p>.
Element and attribute names must be in lower case	Yes. Since XML is case-sensitive	-----"-----	<student>	<STUDENT>
For non-empty elements, end tags are required	Yes. All elements other than those declared in the DTD as EMPTY must have an end tag.	-----"-----	Terminated elements: <p>here is a para.</p> <p> here is another one.</p>	unterminated elements: <p>here is a para. <p> here is another one.
Attribute values must always be quoted	Yes. Even those which appear to be numeric.	-----"-----	<table rows="2" >	<table rows=2>
Attribute Minimization	Not supported. Attribute-value pairs must be written in full.	Supported	unminimized attributes <dl compact="compact">	minimized attributes: <dl compact>
Empty Elements	Not supported unless an end tag or the start tag ends with />.	-----"-----	Terminated empty tags: <hr/>	unterminated empty tags: <hr>

Table 4-2 A table comparing HTML and XML

Table 4-2 compares certain distinctive features of both HTML and XML. Descriptions of each feature (along with example) and its correctness according to the specifications of XML and HTML are given in the table. The reference [176-178] provides essential background required for creating documents using HTML and XML.

Consequently, the decision making of which one between the two alternatives available on the WWW (HTML or XML), is to be selected and deployed for efficient and effective information discovery becomes essential. An attempt to answer this is made with the experiment as outlined below.

4.7.1 Experimental Setup

The experiment had mainly the following components:

- Scanner
- Analyser

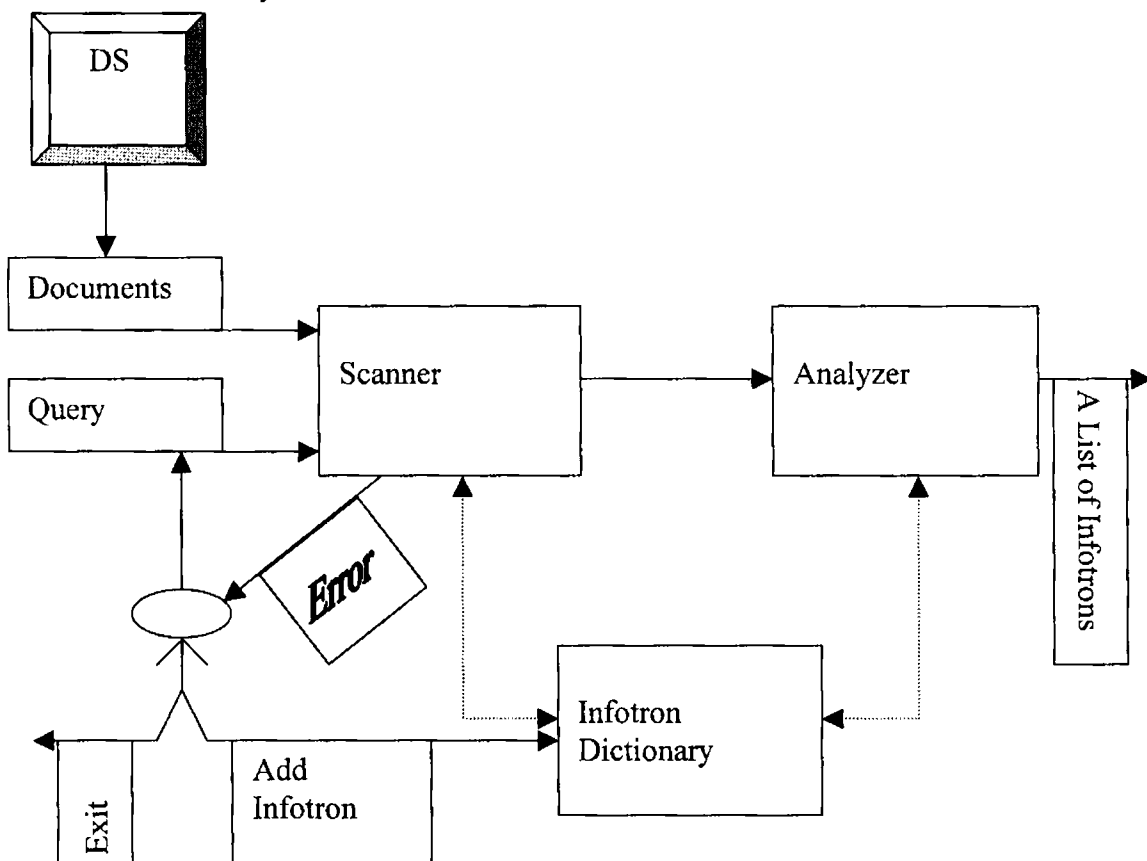


Figure 4-2 Schematic component interaction diagram for the experiment

The interaction among the components is schematically shown in figure 4-2. The functional aspects are explained briefly below.

Scanner: The scanner scans the query and the documents with a view to identifying and extracting infotrons. In this experiment, markup tags are scrutinized to extract infotrons. Metadata if any are ignored while scanning. If anything other than infotron is encountered -that is not understood- by the scanner, it will either seek the user intervention to furnish more details or aborts the operation, in case the user is not having anything or unwilling to provide details. If no such error is encountered then, query and documents along with the identified infotrons are fed to the analyzer for further processing.

Analyzer: The Analyzer will accept the inputs from the scanner and can be reached only if the scanner is successful. Its task is mainly to extract infotrons of interest to the information discoverer, which conforms to the grammar, and language of the domain. For this purpose, Analyzer shall refer to the infotron dictionary to determine associations between the operators and infotrons.

The information discoverer (user) is expected to supply the following details:

- 1.The query (as infotrons).
- 2.The time allowed for the IDS to come out with results (default is 30 minutes).
- 3.Identify the document set (out of the available 200 or all of them) for analysis.

4. Document format is to be chosen from either XML or HTML (default HTML).

The input screen for providing the above information in the form of query is shown in figure 4-3.

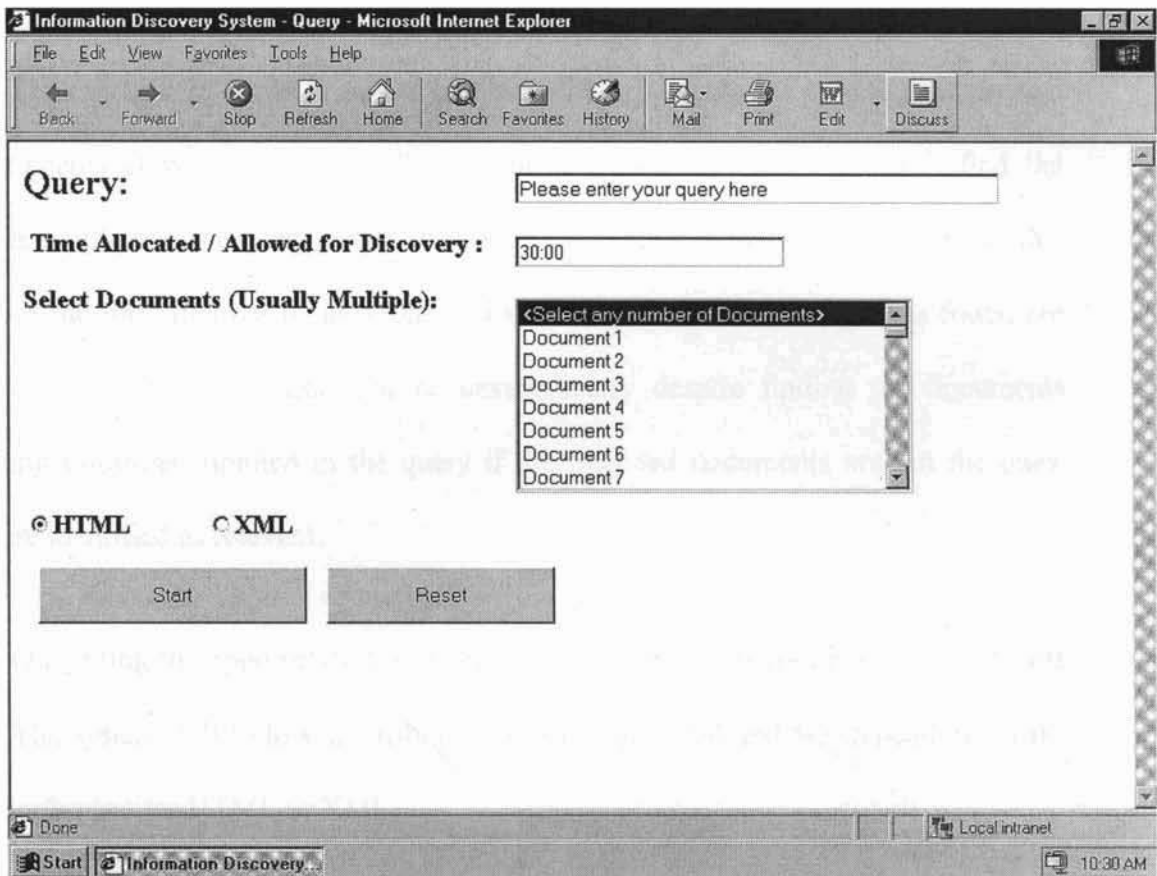


Figure 4-3 A snapshot of the input screen

Queries in this experiment were fed in the form of infotrons. There can be two outcomes for each query and the result could be classified into

1. Satisfactory
- or
2. Unsatisfactory.

The outcome of a trial for the query may be satisfactory if the system could identify the document(s) (among documents chosen) in which the infotrons furnished in the query are found. To determine if a trial is satisfactory the total number of infotrons and the number of distinct infotrons found are recorded.

The outcome of a trial is said to be unsatisfactory in case the system processes all the documents chosen or utilizes all the time available and still could not find the infotrons supplied in the query within the document(s) among documents chosen. In this case both the total number of infotrons and the number of distinct infotrons found are recorded as 0 (Zero). A trial can be unsatisfactory despite finding the documents containing infotrons supplied in the query if the intended documents are not the ones, which are identified as relevant.

On getting the appropriate command from the user (refer figure 4-3), the process starts. The system will follow algorithms shown in figure 4-4 and 4-5 depending on the user's preference for HTML or XML.

The algorithm followed is slightly different for HTML and XML. The differences in the algorithms are mainly due to the nature of HTML and XML specifications themselves.

- Step 1. Read query and extract the infotrons out of it using the scanner and analyzer. Maintain a count of the infotrons as well as the time at which the analysis started.
- Step 2. Read documents in HTML format, marked for analysis one by one and maintain another counter to count the number of documents analyzed. Analyze the document for infotrons extracted in step 1. Count the number of infotrons encountered while analysis of the documents proceeds. Update the time appropriately and if the time exceeds the allocated/allowed time limit, record the time taken for analysis as the time allocated/allowed. Display a suitable message informing the consumption of the time allocated/allowed and then follow step 5.
- Step 3. Analyze the result set (would be in HTML format) got as the output of the analysis phase; record the count of infotrons in the result set and the time taken for analyzing result set. If the result set is nil, initialize both these values to Zero.
- Step 4. Display the total number of infotrons in the result set to the user. Switch to step 6.
- Step 5. Return the result set got after analysis of the documents until this point. Analyze the result set got as the output of analysis phase; record the count of infotrons in the result set and the time taken for analyzing the result set.
- Step 6. Record various parameters measured by the algorithm. If ranking is enabled then prepare the ranking for the result set and measure the time required for the ranking process. Repeat steps 1-6 for all the queries.

Figure 4-4 **Algorithm for analysis of infotrons in HTML documents**

- Step 1. Read query and extract the infotrons out of it using the scanner and analyzer. Maintain a count of the infotrons as well as the time at which the analysis started.
- Step 2. Read documents in XML format, marked for analysis one by one and maintain another counter to count the number of documents analyzed. Validate, parse and analyze the document for infotrons extracted in step 1. Count the number of infotrons encountered while analysis of the documents proceeds. Update the time taken appropriately and if the time exceeds the limit of the allowed time, record the time taken for analysis as the time allowed and then follow step 5.
- Step 3. Analyze the result set got as the output of the analysis phase (would be in XML format); record the count of infotrons in the result set and the time taken for validating, parsing and analyzing the result set. If the result set is nil, initialize both these values to Zero.
- Step 4. Display the total number of infotrons in the result set to the user. Switch to step 6.
- Step 5. Return the result set got after analysis of the documents until this point. Analyze the result set got as the output of analysis phase; record the count of infotrons in the result set and the time taken for analyzing the result set.
- Step 6. Record the all the parameters measured by the algorithm. If ranking is enabled then prepare the ranking for the result set and measure the time required for the ranking process. Repeat step 1-6 for all the queries.

Figure 4-5 **Algorithm for Analysis of infotrons in XML documents**

1. Mention authors for book with title X.
2. Which shelf contains more books?
3. What is the total amount spent on computer books?
4. Mention the title of the book with accession no "12345".
5. Who supplied book with title X?
6. List books having 2 authors and published by X and supplied by Y in the shelf Z?
7. List all 3rd time edited books paid in Rs. and published in the year 1999.
8. List all books having more than 2 authors, published by X and supplied by Y.
9. List all 3rd time edited books paid in Rs. and published after the year 1999.
10. List books that are published by X after year 2000 starting with ISBN "81-78"?
11. List books having 2 authors and published by X in the year 2000 and supplied by Y available in the shelf Z?
12. List all 3rd time edited books paid in Rs. supplied by Y and published in the year 1999 having 2 authors?
13. List books having 2 authors and published by X in the year 1999 and supplied by Y in the shelf Z with exactly 100 pages.
14. List books having less than 2 authors and that cost more than Rs. 500 from publisher X and with less than 100 pages.
15. List books with more than 2 authors and that cost less than Rs. 500 from supplier Y and with more than 1000 pages.
16. What is the total amount spent on the books supplied by Y and available in shelf Z with more than 300 pages published in the year 2000?
17. What is the total amount in Rs. spent on the books published by X and supplied by Y available in shelf greater than Z?
18. List the title of the computer books whose price is greater than Rs. 700 supplied by Y available in shelf greater than Z that has less than 300 pages and contains "81-71" in isbn.
19. What is the total amount spent in \$ (dollars) for computer books having 3 authors and supplied by Y with 300 pages?
20. List books containing the text "XML" anywhere in its title that is priced more than Rs. 300 and published by X available in shelf Z.
21. List the books that are supplied by Y, published by X priced more than Rs. 50 with not less than 50 pages and last name of the author ends with "ni"
22. List books having less than 2 authors in shelf Z and that are priced more than Rs. 500 and starting with ISBN "81-78".
23. List books with more than 2 authors published by X and that are priced less than Rs. 500 and starting with ISBN "81-78"
24. List first and last names of the authors of computer books with the title containing "Arch" and are priced below Rs.1000.
25. What is the total amount spent on the books in Rs that are published by X, supplied by Y and available in shelf Z, with not more than 20 pages and having author whose first name contains "Srinivas"

Listing 4-3 A listing of 25 queries used in the experiment.

25 queries were used (refer Listing 4-3). Further, they were classified as simple, moderate and complex. These three categories of queries were identified on the basis of the number of infotrons in the query. A query is said to be simple if the number of distinct infotrons supplied is less than or equal to three. It is moderate, if the number of distinct infotrons is more than three and less than or equal to five. If the number of infotrons supplied is greater than five it is assumed to be complex.

Query No	Infotrons and operators deduced from query.
1.	authors, book.title
2	books, shelf.mark, >
3	computer.books, price, Σ
4	book.title, accession.number
5	book.title, supplier
6	books, authors, publisher, supplier, shelf.mark
7	books, edition, currency, year
8	books, authors, >, publisher, supplier
9	books, edition, currency, year, >
10	books, publisher, year, >, isbn, ^
11	books, authors, publisher, year, supplier, shelf.mark
12	books, edition, currency, supplier, year, authors
13	books, authors, publisher, year, supplier, shelf.mark, pages
14	books, authors, <, currency, price, >, publisher, pages, <
15	books, authors, >, currency, price, <, supplier, pages, >
16	books, price, Σ , supplier, shelf.mark, pages, >, year
17	books, price, Σ , currency, publisher, supplier, shelf.mark, >
18	book.title, computer.books, price, >, currency, supplier, shelf.mark, >, pages, <, isbn, @
19	currency, price, Σ , computer.books, authors, supplier, pages
20	books, book.title, @, price, >, currency, publisher, shelf.mark
21	books, supplier, publisher, price, >, currency, pages, !<, author, last.name, \$
22	books, authors, <, shelf.mark, currency, price, >, isbn, ^
23	books, authors, >, publisher, currency, price, <, isbn, ^
24	first.name, last.name, authors, author, computer.books, book.title, @, currency, price, <
25	books, currency, price, Σ , publisher, supplier, shelf.mark, pages, !>, author, first.name, @

Table 4-3 Infotrons and operators furnished in the experiment.

The query was furnished, not directly as in Listing 4-3. Instead they were provided to the experiment in the form of infotrons. Table 4-3 depicts the infotrons used on behalf of queries.

Several important measures were recorded as part of the experiment. The description for each is given below:

Total number of documents analyzed (Nda): It is the total number of documents analyzed among the selected ones, for a given query. It could take a maximum value of 200 (as in this experiment, the total number of documents were 200). Ideally it would be the count of documents actually analyzed within the allowed time for analysis.

Total number of infotrons analyzed (Nia): It is the total number of infotrons analyzed before reaching any conclusion on the discovery of information.

Number of infotrons available in the discovered result set (Nir): This is a measure of the number of infotrons available in the discovered result set in case the trial is found to be satisfactory. Otherwise it is recorded as Zero (0).

Total number of distinct infotrons available from, all the documents in the collection (Nid): This measure provides the total number of distinct infotrons available in the domain of analysis.

Total number of distinct infotrons available from, all the analyzed set of documents (Nida): This is the number of distinct infotrons scanned and analyzed from the marked set of documents.

Total number of distinct infotrons in the query (Niq): Total number of distinct infotrons as available in the query supplied by the discoverer. [This can be different from the number of infotrons available in the query if intelligent query reformulation is allowed. However, in this experiment no such query reformulations were allowed.]

Time taken for analysis phase (Ta): It is the total time taken for analyzing all the documents that are marked for analysis.

Time allocated/allowed by discoverer to discover (Tal): This measure is dependent on a given query and represents the time allowed by the discoverer to finish the discovery process. Sometimes if all document collections are exhausted in the analysis process it could be less than the time allocated, in which case this would be the same as time taken for analysis. In this experiment a default value on the higher side of 30 minutes has been used.

Time taken for ranking (searching and sorting) of result set (Trr): Once the results are got then they may have to be ranked based on some criteria. Thus Trr is time taken for ranking the result set. This measure is having relevance only if the outcome is satisfactory. Otherwise it is recorded as Zero (0).

4.7.2 Software

Implementation of this experiment was done using Java programming language. As the Java programming language is object oriented and platform independent, the code that was developed has been used on many different machines and tested on different hardware platforms as mentioned in section 4.7.3. The operating system used was Windows 95 in the developmental system.

4.7.3 Hardware

Developmental hardware platform: The hardware specification of the developmental platform where the actual source code required for the experiment was developed is a machine based on pentium processor running at 200 MHz with 64 MB RAM and 2.1GB hard disk.

Other hardware platforms used: Apart from the system on which the software was developed, parts of system were tested on different hardware platforms. Their details are shown in table 4-4.

HP K Class server K260 3 CPU PA8000 2GB RAM, 4 X 9GB SCSI HDD 19" Color SVGA Monitor.	SUN Workstation ULTRA SPARC –III 300 MHz 256 MB RAM 4.3 GB HDD 19" Color SVGA Monitor.	HP Workstation PA 8200 4 X 128 MB RAM 45 GB HDD 19" Color SVGA Monitor
---	--	---

Table 4-4 Hardware platforms used for testing purpose

4.7.4 Results of the experiment and Conclusion

The experiment is repeated for 25 queries and results are recorded both for HTML and XML. The results are shown in table 4-5 through 4-8. From these tables it is evident that XML way of representing infotrons required less time for analysis.

Query Number	Nda	Nia	Nida	Nid	Nir	Niq	Outcome Satisfactory?
1	200	852	15	586	64	2	Yes
2	200	770	17	586	0	2	No
3	200	412	9	586	0	2	No
4	200	754	14	586	43	2	Yes
5	200	680	13	586	76	2	Yes
6	150	925	12	245	84	5	Yes
7	150	775	11	245	75	4	Yes
8	150	878	12	245	0	4	No
9	150	753	12	245	0	4	No
10	150	867	12	245	0	4	No
11	50	884	15	166	53	6	Yes
12	50	954	16	166	72	6	Yes
13	50	1038	17	166	67	7	Yes
14	50	986	14	166	0	6	No
15	50	972	15	166	0	6	No
16	50	979	15	166	0	6	No
17	50	965	14	166	0	6	No
18	50	482	10	166	0	8	No
19	50	445	9	166	0	6	No
20	50	924	13	166	0	6	No
21	50	889	12	166	0	8	No
22	50	935	13	166	0	6	No
23	50	1011	15	166	0	6	No
24	50	425	9	166	0	8	No
25	50	1089	16	166	0	9	No

Table 4-5 Results of infotron measures for 25 queries in HTML.

Query Number	Ta in (milli seconds)	Tal in (minutes)	Trr (milli seconds)
1	25012	30	0075
2	22789	30	0
3	18420	30	0
4	21675	30	0123
5	19785	30	0135
6	26675	30	2247
7	22295	30	2143
8	25786	30	0
9	22123	30	0
10	25439	30	0
11	25643	30	2167
12	27888	30	2353
13	29876	30	2512
14	29076	30	0
15	28967	30	0
16	28654	30	0
17	28644	30	0
18	15718	30	0
19	14342	30	0
20	26743	30	0
21	25783	30	0
22	26783	30	0
23	29637	30	0
24	13243	30	0
25	30898	30	0

Table 4-6 Results of timing measures for the 25 queries in case of HTML.

Query Number	Nda	Nia	Nida	Nid	Nir	Niq	Outcome Satisfactory?
1	200	815	13	586	55	2	Yes
2	200	721	12	586	34	2	Yes
3	200	378	9	586	24	2	Yes
4	200	705	13	586	42	2	Yes
5	200	656	13	586	54	2	Yes
6	150	892	12	245	61	5	Yes
7	150	758	11	245	56	4	Yes
8	150	831	12	245	78	4	Yes
9	150	753	12	245	52	4	Yes
10	150	834	12	245	0	4	No
11	50	857	13	166	47	6	Yes
12	50	942	14	166	57	6	Yes
13	50	982	13	166	55	7	Yes
14	50	957	14	166	61	6	Yes
15	50	933	12	166	48	6	Yes
16	50	924	12	166	54	6	Yes
17	50	938	13	166	38	6	Yes
18	50	413	9	166	0	8	No
19	50	395	8	166	27	6	Yes
20	50	891	13	166	0	6	No
21	50	868	12	166	0	8	No
22	50	903	12	166	0	6	No
23	50	968	13	166	0	6	No
24	50	364	8	166	0	8	No
25	50	989	14	166	0	9	No

Table 4-7 Results of infotron measures for 25 queries in XML.

Query Number	Ta in (milli seconds)	Tal in (minutes)	Trr (milli seconds)
1	17608	30	67
2	16052	30	363
3	12258	30	225
4	15272	30	50
5	13949	30	83
6	18772	30	998
7	15706	30	1180
8	18150	30	1203
9	15586	30	1032
10	17907	30	0
11	18050	30	1196
12	19621	30	1301
13	21013	30	1394
14	20453	30	1356
15	20376	30	1351
16	20157	30	238
17	20150	30	562
18	12306	30	0
19	11757	30	128
20	18820	30	0
21	18148	30	0
22	18858	30	0
23	20845	30	0
24	10853	30	0
25	21718	30	0

Table 4-8 Results of timing measures for the 25 queries in case of XML.

Further, T_a measures for both XML and HTML are plotted. The results are shown in the form of figure 4-6. From the chart it can be seen that XML representation takes less time for analysis. Thus the figure 4-6 clearly favors XML representation for infotrons. In both cases the curves tread a similar path. This indicates the effect of the analysis routines on both XML and HTML representation is same.

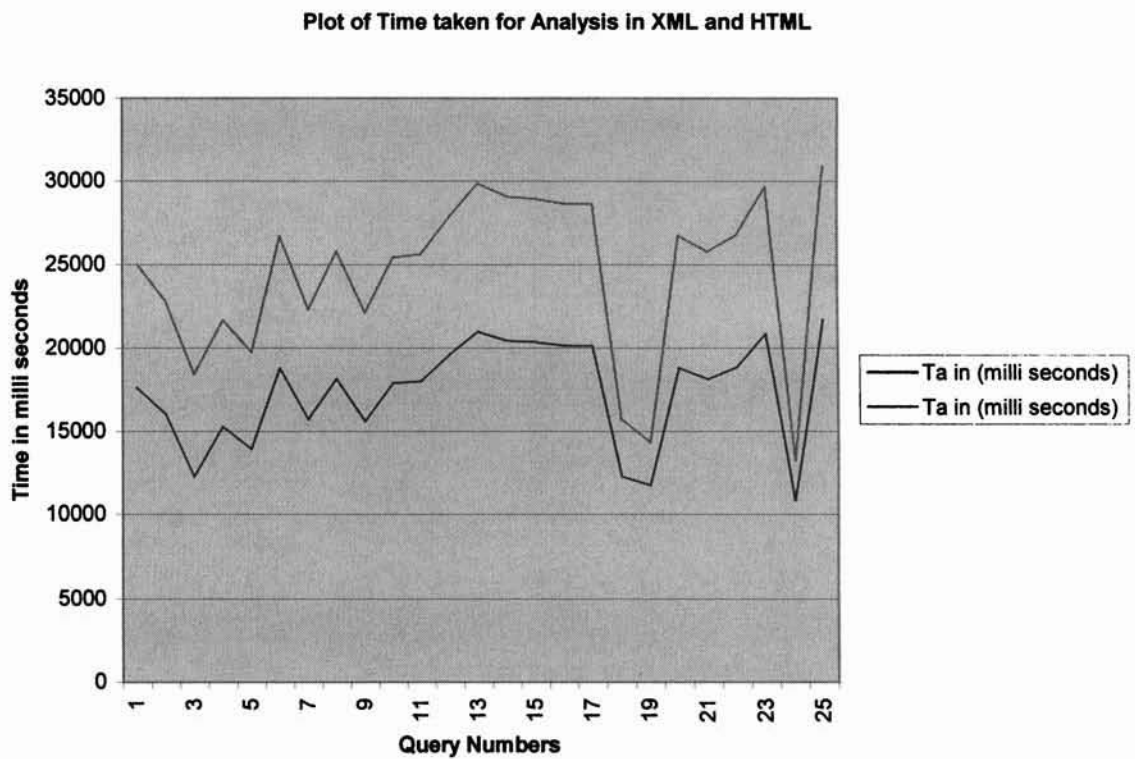


Figure 4-6 Chart of time taken for analysis for 25 queries in HTML & XML.

It is observed that all simple queries yielded satisfactory results in XML. It performed better in other categories of queries as well. From the results as in the table

4-9, it could be concluded that XML would be better by a factor of 2 when compared with HTML for infotron incorporation.

Alternatives for Infotron Representation	No of Queries	Query Type		Percentage Of Queries Answered Satisfactorily	Overall Percentage
HTML	25	Simple	20%	60%	32%
		Moderate	20%	40%	
		Complex	60%	20%	
XML	25	Simple	20%	100%	68%
		Moderate	20%	80%	
		Complex	60%	53.33%	

Table 4-9 Summarized result of the experiment

Apart from these testimonies, there are other reasons for preferring XML to HTML. Major ones are mentioned here.

- XML documents need to be validated and well formed. Such a structuring restriction on the contents of the documents makes it adaptive for information (refer section 4.4), by conforming to a given language and its grammar.

- Processing can be in multiple ways. For instance, depending on the problem in hand the processing can proceed by following event driven, DOM or tree based method. In fact such processing in XML, resulted in computing the total amount spent on computer books. Also it becomes easy to compare the total number of books present in each shelf and find which shelf contains more number of books.
- XML is being augmented by other standards such as Xpointer, Xpath, Xforms Etc.,

4.8 INFORMATION DISCOVERY SYSTEM

A system, which we call Information Discovery System (IDS), is one that starts with the infotron(s) supplied as clue(s) (fed to the system in the form of a query), and results in brewing the information required to satisfy the need of the information discoverer by utilizing the documents at its disposal (as vast information space). IDS uses documents at its disposal in addition to the infotron dictionary for discovering information. In order to discover the information, the system starts with the query containing infotrons. Infotrons that are provided in the query becomes clue to the system. Information discoverer should allow sufficient time for system to explore the vast information space. Given sufficient time, it is expected that an IDS in return reward the information discoverer with information that would satisfy him.

4.8.1 Requirements

Information discovery system is expected to possess certain characteristics. They are enumerated below:

IDS must be

- Extensible.
- Scalable.
- Adaptable.
- Robust.
- Functional in an open environment.
- Having a vast information space at its disposal for discovery.
- Capable of surviving network faults and errors (fault tolerant).

4.8.2 Specifications

In order to design such an IDS system, the following specifications were used.

- The system will be designed to study the feasibility of information elements as the modelling base for successful information discovery using XML documents (for incorporating infotrons) available in the internal network, in lieu of full text database, as is the case in IR.
- The IDS must be configurable to work in two modes. IDS can be configured to return a single document containing the merged result from different resources/documents. Alternatively, it can return multiple documents from the information space.

- The vast information space is a collection of XML documents containing the infotrons. If such a space is not available then it must be created using the transformation of traditional databases or HTML pages into XML documents. Further, these XML documents must be validated for well formedness. This is necessary, as the IDS should be deployed in an open distributed environment. Task of conversion of databases to XML is fairly easy. Major vendors of RDBMS packages are already providing the facilities to export or import data from XML (for instance MS-SQL Server, DB2 and Oracle).
- The system should have a web-based interface for the discoverer to interact with the system for providing vital inputs either by means of a query or a system initiated dialog to control the process of discovery.
- The system should have provisions to scan both the information content in a document for discovery purpose and the query supplied by the information discoverer. Documents are supplied by the Gatherer subsystem (refer to Gatherer in section 4.8.3 below). The information discoverer should supply the query string with infotrons. The Scanner should identify and separate the infotrons. The Analyzer, Classifier and Categorizer and other components of the system would further brew information from the scanner-identified infotrons; thus ultimately resulting in information being discovered.
- The system should be capable of maintaining three different types of dictionaries viz., system wide infotron dictionary, user specific infotron dictionary, and session specific infotron dictionary.

- Maximum reuse should be attempted by utilizing the same algorithms for different components viz., Scanner, Analyzer, Classifier and Categoriser etc., wherever it is possible to do so.
- The discoverer controls the way the infotron is stored in his private infotron dictionary. A wizard like interface to help the discoverer should be provided for manipulating the discoverer's private infotron dictionary. The manipulations allowed mainly are adding, modifying, and deleting infotron's from discoverer's private infotron dictionary.
- The provision for maintaining multiple infotron dictionaries by the same discoverer should be provided.
- During reference of infotron dictionary, the first preference must be given to the discoverer's private infotron dictionary. In case the infotrons expected to be present in discoverer's private infotron dictionary are unavailable, then the system wide infotron dictionary can be consulted with discoverer's consent. Alternatively, a session wide infotron dictionary could be created whose life ends as soon as the time span of the session expires either due to user's request or due to system failure.
- Multiple infotron dictionaries can be active at any given time for the same discoverer, provided there is no conflict among them. Conflict between two infotron dictionaries is said to exist, if both of them contain the same infotron with different specifications for them. Discoverer cannot have infotrons in his private dictionary those that are part of the system dictionary. Thus any potential conflicts between system and discoverer's dictionary are not allowed to creep in.

- The ranking of the resulting set after discovery should be done (in case the IDS is operating in multiple document mode). This may be achieved by using some statistical evaluations on the number of infotrons present in each document of the resulting set. The ranking question does not arise in case the IDS is configured to supply only a single document as result.
- The system should be designed by keeping in mind the penta-tier architecture described in chapter 3. Components of the IDS system are expected to run in one/more of the tiers of the architecture. The exact positioning of independent components in the architecture depends on the nature and functionality of the components. In case a component has to be developed which has to appear in more than one tier then such a component has to be logically divided into subcomponents with proper care and implemented separately by making them amenable to different respective tiers of the architecture.

4.8.3 Architecture

The architecture of IDS that satisfies most of the specifications and the characteristics listed above is shown in figure 4-7.

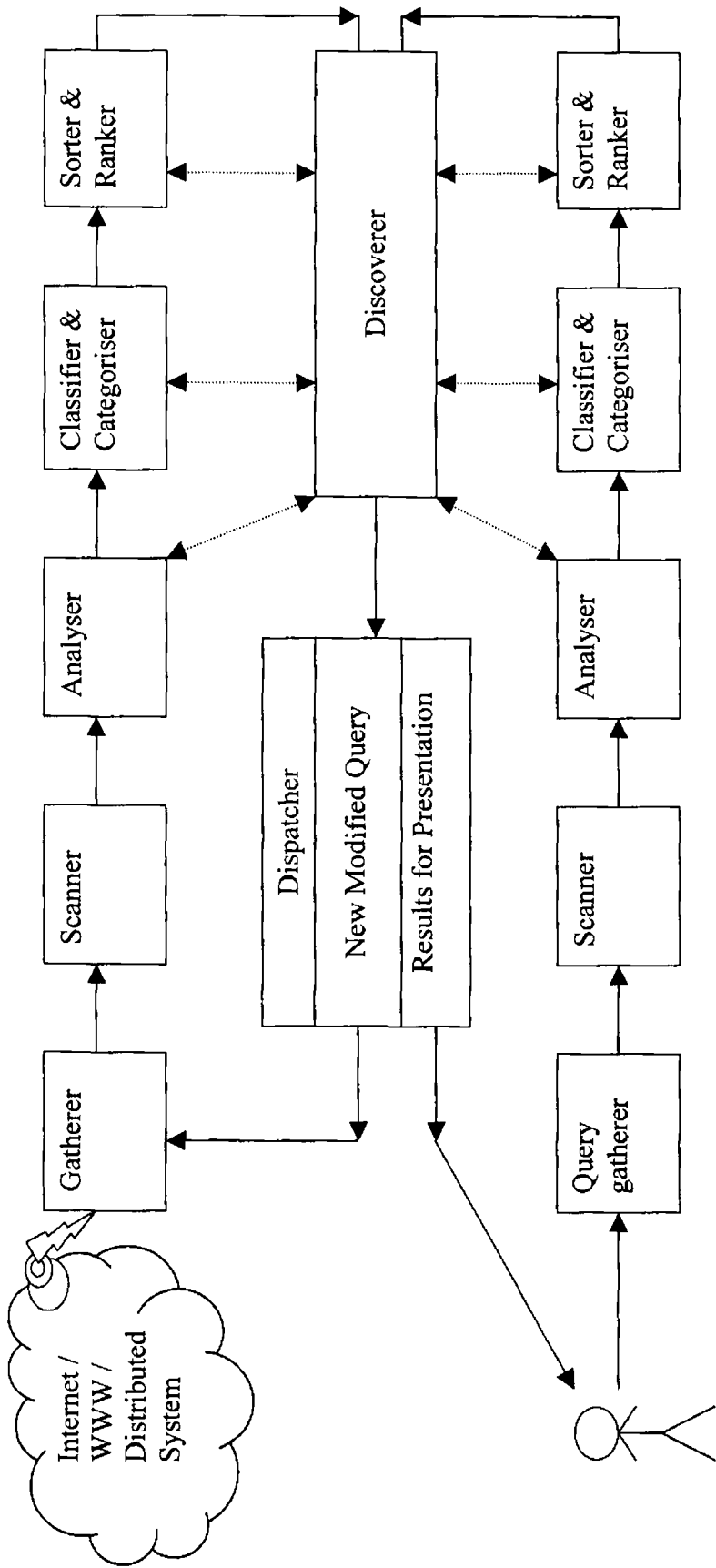


Figure 4-7 Architecture of Information Discovery System

The architecture shown contains the following components:

- Scanner
- Analyzer
- Classifier and Categoriser
- Sorter and Ranker
- Discoverer
- Dispatcher
- Gatherer
- Query gatherer

Scanner and Analyser are described in section 4.7.1. The rest of them are described below:

Classifier and Categoriser: From the list of the infotrons provided by the Analyser, this component of the architecture classifies and categorises them. This would further help in expansion or shrinking of the candidate document set that was initially (or subsequently generated by the system at any given point of iteration) identified for investigation.

Sorter and Ranker: This component of the IDS is responsible for assigning the weights to the infotrons in association with the Discoverer component (described next) and infotron dictionary. Such a weight assigning mechanism would help in sorting and ranking of the documents -identified as relevant- for the purpose of presentation.

Discoverer: This component is the heart of the IDS. Basically it is a collection of algorithms based on the IR techniques (discussed in detail in section 2.4), tuned to work with infotrons and infotron dictionary. Since the process followed is the same for both the query as well as the documents (refer figure 4-7), intelligent information discovery is possible.

Dispatcher: From the document set provided by the Discoverer component, Dispatcher would appropriately place them based on the weights associated with infotron and relationship between other infotrons (derived from infotron dictionary), to provide a single or multiple documents as requested by the mode selected for operation of IDS by the user. In the context of a hint from the discoverer for gathering more documents from the networked environment, this component feeds the Gatherer component (see below) with appropriate infotrons.

Gatherer: Gatherer component is composed of multiple spiders, which can traverse independently the information space to identify and retrieve the documents appropriate for the infotrons that Gatherer has been fed with. The documents so gathered

are further channelised to the Scanner (refer section 4.7.1) thus completing the cycle, which can be repeated till a satisfactory answer is reached by the Discoverer component.

Query gatherer: The user of the IDS starts the above repetitive cycle by invoking this component. This component acts as an interface for the user to supply the infotrons as query, which would be fed to the Scanner for analysis and refinement. Other inputs like the time allowed for the discovery and the selection of initial document set can also be incorporated for more efficient and effective discovery in this component of the system similar to the one in figure 4-3.

All the components of the system described above, have interaction with the infotron dictionary that is not shown in figure 4-7. Hence the design and development of infotron dictionary is as crucial as the design of the data warehouse and populating it. The design of infotron dictionary is highly dependent on the domain in question. It is quite evident that infotron dictionary banks on the language and grammar of the infotrons -that form its basic constituent parts- unlike data warehouse that does not have any such constraint. Nevertheless, the important point to note is the similarity of the nature of infotron dictionary with that of the static data discussed earlier (refer chapter 3) with respect to the penta-tier architecture. The following paragraph envisages the analogy among penta-tier architecture and the IDS architecture and suggests how and where different IDS components can fit within the available tiers of the penta-tier architecture (refer figure 4-8).



Figure 4-8 IDS components in Penta-tier Architecture.

The (system wide) infotron dictionary represents the static data for the IDS. It is ideal to place it into the data warehouse tier in the penta-tier architecture. Occasionally the user keeps creating the session-wide infotron dictionary and he also maintains his own private infotron dictionaries apart from the system wide infotron dictionary. These would act as the dynamic data and in due course of time (if the user prefers), it could be merged with the system-wide infotron dictionary thus making it static. The networked computing machine and the network operating system used by the underlying DS (whether Intranet/Extranet/Internet) forms the lowest tier in the architecture. The datamining tier is composed of the Discoverer and Classifier & Categoriser components

of the IDS (refer figure 4-7). The OLAP tier consists of the Sorter and Ranker component apart from other system-wide cache that may have to be maintained for effective accessing (like Document Object Modelling (DOM) and other accessing components, Simple Object Access Protocol (SOAP) etc., used for XML) of the documents in information space. The main candidate for the Data visualization tier of the penta-tier architecture is Query gatherer and parts of Dispatcher components. The rest of the parts of Dispatcher component would fit more appropriately into the datamining tier.

4.8.4 Implementation Aspects

Software and the tools used: Implementation of IDS was done primarily by using the combination of *Java programming language* and *Java scripts*. For generating the XML documents mainly *Notepad* was used. XML documents were validated using softwares like *XMLSpy*, *XMLwriter* etc. Since the implementation was providing web interface, most parts of system were tested on different hardware platforms as mentioned earlier in section 4.7.3. There was no change in the configuration of the developmental system.

4.8.5 Performance results of the IDS components

It was not possible to verify the relative performance of IDS with other similar system that uses XML, mainly because of the non-availability of the latter. To the authors knowledge there is currently no such system, which uses XML and is operational having the features as IDS. Hence profiling of the components of IDS was done to assess

the resource utilization pattern. This identified components that were in the state of execution for a long time. It was observed that the Scanner, Analyser and Discoverer components were CPU intensive where as Gatherer and Dispatcher were I/O intensive. Analyser along with Discoverer took more than 80% of the CPU time as compared to all other components put together. Analyser and Discoverer components were accessing the Random Access Memory (RAM) substantially. This was mainly to access the infotron dictionary, which was maintained in RAM for efficient operations. The table 4-10 depicts the resource utilization by different components of IDS.

Sl.No	Name of the Component	% CPU Utilisation
1	Scanner	7
2	Analyser	42
3	Classifier & Categoriser	6
4	Sorter & Ranker	3
5	Discoverer	39
6	Dispatcher	2
7	Gatherer & Query Gatherer	1

Table 4-10 Resource utilization of components of IDS.

To summarise, a generic IDS is developed according to the specifications given in section 4.8.3. The penta-tier architecture was followed for the development of IDS. The results observed were encouraging. Further to test and demonstrate the usefulness of the system an application for its usage is developed which we called “IDLIS” (Information Discovery in Library Information System). IDLIS is described in the next section.

4.9 INFORMATION DISCOVERY IN LIBRARY INFORMATION SYSTEM (IDLIS)

In order to demonstrate the working of the IDS developed in the previous section and to discover the information (without modification to LIS), an Information Discovery in Library Information System (IDLIS) application was developed. IDLIS is essentially a wrapper for the LIS (Library Information System), which maintains all the databases of the library and transactions on these databases. The purpose was to demonstrate that the functionality of a legacy system could be enhanced with the augmentation of IDS leading to information discovery service. In the process of demonstrating IDS, the focus was on the issues that would prove to be significant at the time of implementation and deployment. Thus a front end, which is a web-based interface with Graphical User Interface (GUI) capability, was developed for interaction with legacy system. This can conveniently be called an ID front. The GUI interface is as shown in figure 4-9.

The IDS developed requires XML document collection for discovery, therefore library database that was available in the form of a proprietary database was transformed into XML documents. These documents were then used by IDS system for analysis and information discovery. The database used for LIS had multiple tables each with several records. Details of LIS database are shown in table 4-11. Since the LIS had the books table containing the bulk of records only this table has been used.

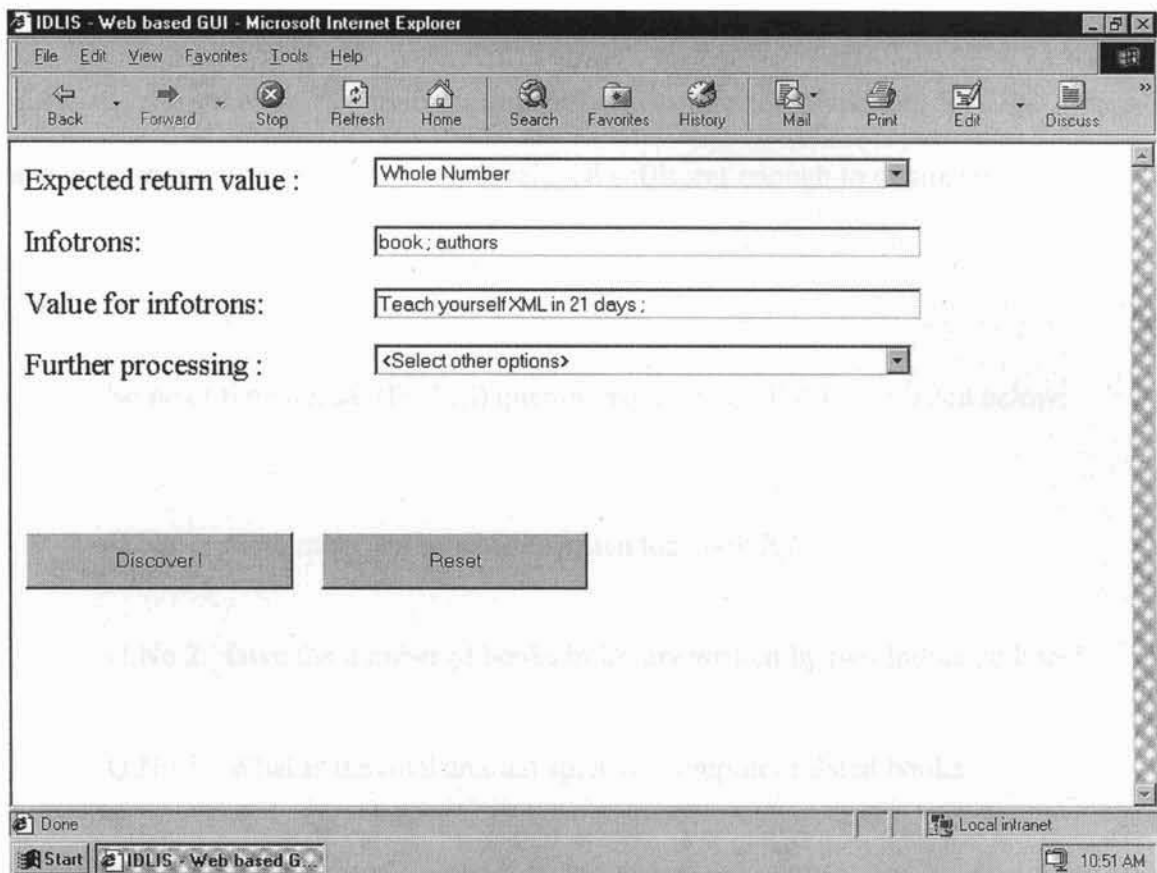


Figure 4-9 Web-based GUI for IDLIS

Tables in the LIS database	Number of records
Books	49,369
Authors	39,699
Publishers	7,873
Borrowers	3,823
Journals	750
Serials	722
Audio-Visuals	32

Table 4-11 Details of LIS Database

The user was free to enter any query related to a book. Answering many of the questions from the user's required a comprehensive infotron dictionary for LIS. A small and concise infotron dictionary was developed sufficient enough to demonstrate IDS.

Some of the unusual (for LIS) queries processed by IDLIS are listed below:

Q.No 1: How many authors have written the book X?

Q.No 2: Give the number of books in library written by two Indian authors?

Q.No 3: What is the total amount spent on computer related books?

Q.No 4: List the books available in library that are published by X after 2000 AD?

Q.No 5: Which shelf contains more computer books?

In order to answer these queries first the actual intention of the user should be made clear and explicit to the system. Therefore, the queries must be transformed to reflect these intentions. From the queries listed before, discrete infotrons as given in the table along with the expected return value by IDS can be derived. The table 4-12 also shows the infotrons extracted from the query and their associated value. Using these infotrons, the IDS employs a strategy to explore the XML documents looking for the elements as given by column 3 and satisfying conditions in column 4 of table 4-12. The

relevant documents are further processed based on column 2 and column 5. Additional miscellaneous data may have to be supplied as in the case of query 3.

Query Number	Expected return value by the IDS	A list of infotrons furnished to IDS	A value for X, which is related to an infotron (given in bracket).	Miscellaneous (Additional processing required)
Q.No 1	Whole Number (integer)	book, authors	Value:X (book)	
Q.No 2	Whole Number (integer)	book, write, author, publisher, nationality	Value:"India" (place); Value:"two"(author); Value:"2" (author)	
Q.No 3	Currency	books, price	Value:"computer"(book)	Total
Q.No 4	List	book, publish, publisher, >, year	Value:X(publisher); Value:>2000(year)	
Q.No 5.	Whole Number (integer)	book, shelf.mark	Value:"computer"(book)	

Table 4-12 Infotrons furnished to IDS system

The system was functioning reasonably well for the unusual queries as mentioned above. The performance was satisfactory. The result of the system for some queries, which were answered correctly by the system, is provided in figures 4-10 through 4-12.

The study resulted in surfacing certain issues which otherwise could have gone unnoticed. These issues are mentioned below:

1. Lack of consistent method for specifying and formulating queries.
2. Lack of user's domain knowledge could make the specification task much difficult. It was observed that the users who had some knowledge about the

application domain had a better vision of what to expect from the system and hence could formulate better queries.

3. Merging of documents is not a trivial task and requires a more comprehensible infotron dictionary.
4. The domain knowledge of the application plays a major role in designing and deploying the infotron dictionary.
5. At the practical level, generating the XML documents, which conforms to specification, requires considerable expertise.

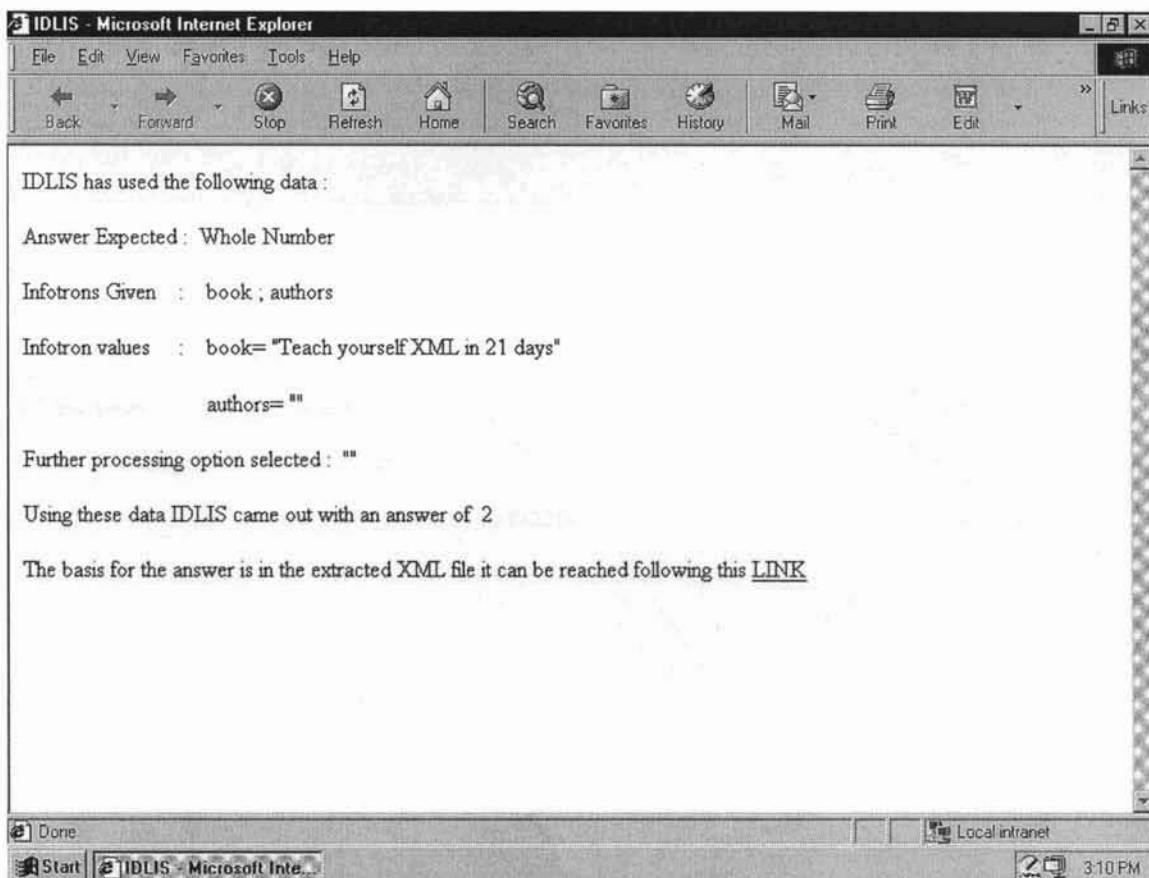


Figure 4-10 Results for Q.No 1

```
<?xml version="1.0" ?>
<document>
<book>
  <book.title>Teach yourself XML in 21 days</book.title>
  <authors>
    <author>
      <first.name>NORTH</first.name>
      <middle.name />
      <last.name>SIMON</last.name>
    </author>
    <author>
      <first.name>HERMANS</first.name>
      <last.name>PAUL</last.name>
    </author>
  </authors>
</book>
</document>
```

Figure 4-11 A corresponding XML file got as result.

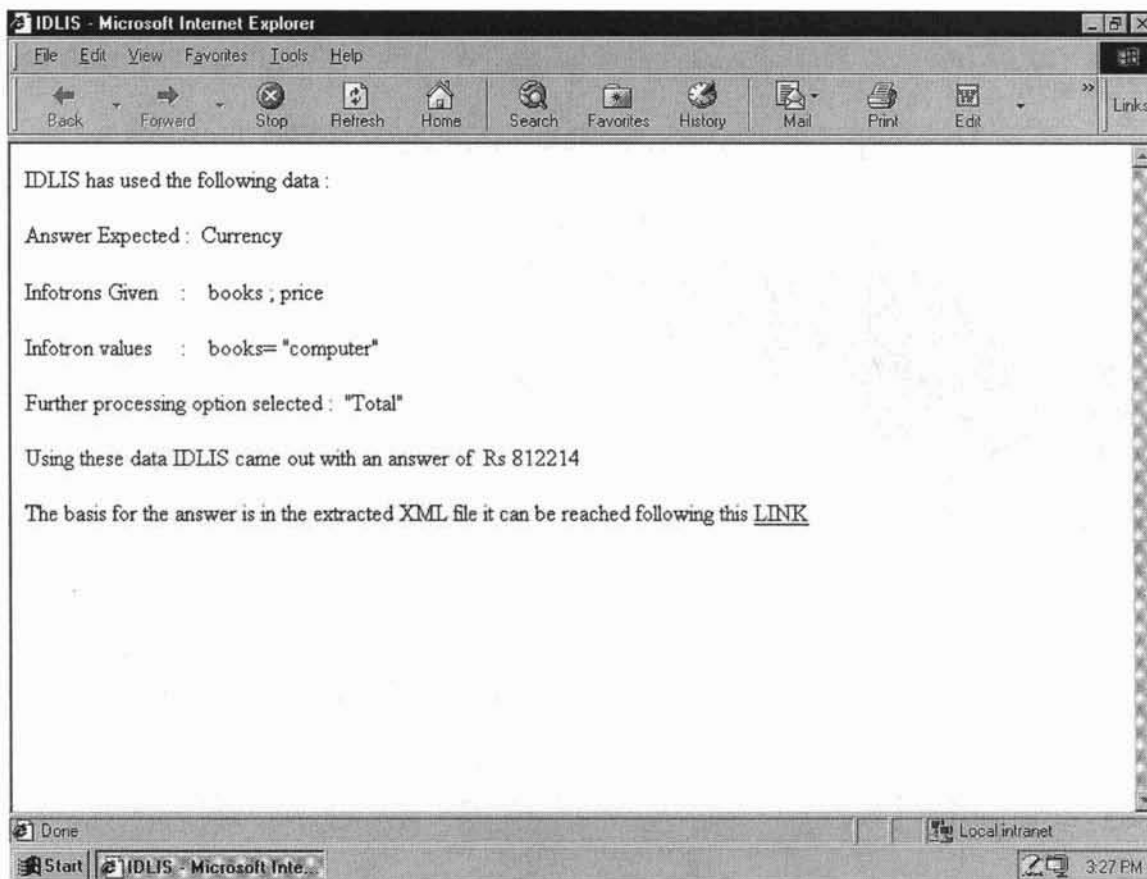


Figure 4-12 Results for Q.No 3 from IDLIS

4.10 SUMMARY

An attempt has been made to arrive at a general information discovery system along with a useful and novel definition for information. To resolve the problems faced by the user and to reflect his understanding to the system, the concept of infotrons is supplemented by infotron dictionary. The study conducted shows that XML is the preferred method for incorporating infotrons. IDS design and architecture is benefited with the penta-tier architecture. IDS can be augmented to the legacy system for providing the additional functionality of Information Discovery Services.

CHAPTER 5

CONCLUSION AND FUTURE DIRECTIONS

The research work was pursued, by following a novel approach towards information as a whole and its discovery, with the prospect of identifying units of information elements. A proper pathway towards the end result has been achieved showing light at the end of the tunnel. The software architecture was presented which can scale along with new techniques and methodologies as they evolve. It is possible to implement the IDS even with the legacy system. Legacy system gets additional capability with the ID front. The following concepts/ideologies were used to realize IDS.

- A. Information Discovery Paradigm (Section 3.2)
- B. Penta-tier Architecture with five complementary technologies. (Section 3.3.2)
- C. Concept of infotrons. (Section 4.3)
- D. A formal definition of Information using infotrons. (Section 4.4)
- E. Comparison of XML and HTML for ID purpose. (Section 4.7)
- F. Infotron Dictionary. (Section 4.5)

This chapter mainly has 3 sections. Section 5.1 lists the major contributions of the work. Section 5.2 observes the areas where IDS can find application. Section 5.3 identifies major issues that one might address to enrich the discipline of information and its discovery.

5.1 MAJOR CONTRIBUTIONS OF THE WORK

The major contributions of the research work reported in this thesis can be described as follows:

1. Identification of major issues and problems that surround the design of IDS.
2. Identification of XML as a new way of creating and disseminating information in the context of DS. Section 4.7.2 has provided the rationale for the suitability of the XML in the design of IDS.
3. It has been demonstrated that information can be conveniently described with the concept of infotrons. The infotrons can be easily implemented using XML like specification which makes the various ways of processing the infotrons possible by allowing it to be processed using either event driven or tree based techniques which come naturally as a byproduct of the usage of XML.
4. The concept of infotrons are implemented in IDLIS application to show and demonstrate that the IDS can indeed be used without much difficulty even with the legacy systems with little or no modification to the system. Most of the times only an appropriate conversion routine is all that is necessary to convert the data in the legacy system in to XML documents. Even this is not the case in future, as

most of the industry leaders are already in the process of providing a way to convert the data stored in proprietary form to XML.

5. The penta-tier architecture based on the five complementary technologies (viz., Networked Computing Elements and Network Operating System, Data Warehousing, Data Mining, OLAP and Data Visualization).

Limitations of the Present Work:

1. The decision of using one among the alternatives available in the context of WWW (viz., HTML and XML) is based on the experiment conducted, which used 200 carefully hand coded documents. The hand coding was preferred as some of the editors were in their infancy for XML at the time experiment was conducted. Though the number of documents used in the experiment is quite small compared to the number of documents stored while the IDS would be operational, there are other good reasons to tilt the scale in favor of XML.
2. The system has not been tested on the Internet. The reasons for this are the following:
 - A. Insufficient number of XML documents for testing on WWW.
 - B. Browsers rendering XML were hardly available and using XSL for compensating browser inabilities would have brought us back to HTML.

3. The system has been tested to work properly on Linux and Windows XX platforms at the user interface side. Other major platforms can be used but have not been tested.

5.2 POSSIBLE APPLICATIONS OF IDS

The following are the major areas where one can apply the IDS with suitable modifications

5.2.1 Internet Information Discovery (IID)

This application can become prominent when the Internet grows with lot of XML web pages or most of the information providers plan to disseminate information in XML. The present search engines can be used as the starting point to collect and gather the URL's of interesting Internet resources, which could be further explored, using the IDS.

5.2.2 Web Services Discovery

With the growth in Internet, there is a clear trend to use WWW as alternative to house the services. All serious service providers are already harnessing this option. This may lead to a plethora of web services and its providers. In the context of many service providers being available, selecting the best service provider is a non-trivial task. IDS helps to discover the right service provider. Further, when the competition among service providers gets intense, they will keep changing their service charges. In such situations, IDS can be used to identify best web service providers from time to time based

on certain constraints provided by the service seeker. The constraints may vary often. These are dependent on the urgency and the need of the service seeker. Constraints could be for instance, economical, geographical, technical or any other.

5.2.3 Forensic Systems

Another important area is the security of resources (both hardware and software) apart from the information itself (most organizations consider information as a resource now), which are shared among the users of a distributed system. This sharing of the resources comes with the issue of safety and security of the same. No wonder then today Internet and its well-wishers (including IETF, W3C, academicians and common man and organizations) are concerned about the security of resources. It may be difficult if not impossible to evade the hackers, crackers and miscreants. Despite threats, the brave hearts still want to utilize and benefit from the irresistible opportunities a DS provides. A supportive effort for such individuals and organizations is the forensic systems. These systems provide the information about the intruder and would help in tracing the miscreants and the cause of the disaster. IDS can help discover the information related to the intruders and their modus operandi, thus helping the forensic systems to benefit immensely by incorporating IDS within their hood.

5.2.4 Machine Translation and Natural language understanding

With the machine having at its disposal an infotron dictionary belonging to each natural language, machine translation of natural languages can be easily achieved. Even

machines of future may understand the natural languages better, with IDP, infotron and infotron dictionary concepts. The introduction of infotron dictionary, along with IDS in to the machine, will radically change its capacity to understand the natural language instantaneously.

5.3 FUTURE DIRECTIONS

The journey in the field of information discovery needs much to be done than what has been achieved. The following issues that are identified might be helpful as a whole for the discipline of information and its discovery.

1. While designing and developing IDLIS it was observed that begetting end users and domain experts to consensus and making them understand why the data is put the way it is put and why some of them cannot be treated as infotrons was not a trivial one. This has mooted a bigger question of formulating standards for determining the infotrons in a consistent way. The chances of determining the infotrons in a non-consistent and designer dependent way can make the whole of IDS behave in an unanticipated-way and make it useless in delivering the services intended. More harm is caused when the system delivers with wrong inferences and misinformation is discovered rather than information. Thus, it is necessary to evolve a global standard in order to determine infotrons in a consistent and not being designer dependent way.

2. As the IDS has to work with multiple infotron dictionaries; the results obtained are having direct consequences on the quality of the information being discovered. This necessitates infotron dictionary to be handled very carefully. The concepts like static, dynamic and meta data as explained in the context of penta-tier architecture must have an unambiguous policy of classification and categorizing without a possible overlap among themselves. Developing infotron dictionary with this approach requires considerable effort on the part of designer and implementer. Since computing machines have to use these dictionaries ultimately, there can be only one interpretation possible for each entry in the infotron dictionary. Other possible interpretations for the infotrons may have to be facilitated in the infotron dictionary again in a consistent way just like that of identifying infotrons in an application domain.
3. Deciding relationship between infotrons and their operators is a non-trivial task. This fact actually highlights the domain expert's knowledge and his/her skill in gluing infotrons with their operators. Here again there is a requirement for a global level agreement among designers and architects of IDS about interactions of operators and infotrons. Indeed rigorous research in this direction is essential to contemplate identification of operator-infotron interactions in an efficient and effective manner. Also the use-cases (widely applied in object oriented systems development) can be tailored to suit IDS context, there by making it possible to even generate the code for IDS like systems automatically and instantaneously.

4. For implementation of the IDS in this study - java and related technology was used. The java being an object-oriented language facilitates platform independence. It also carries with it most of the advantages that any other procedural and/or object oriented programming languages may achieve. However, there are other languages, which support logical programming. These are extensively used in AI field. Some of these programming languages like Lisp and Prolog may change the performance level of the IDS in particular. Thus utilization of programming languages like Lisp, Prolog than the conventional C, C++, C# or Java for the implementation aspects can be considered for further research.

5. Most of the search engines available on the Internet use keyword oriented searching in the Internet information space. Along with the keywords some of the search engines even allow modifiers (special symbols), which change the meaning and association of the keywords present in the query. As a result, the links delivered may change drastically depending on the keyword and modifiers in the query. One excellent idea would be to generate the queries to such search engines including the IDS, automatically / programmatically. It would be interesting to experience the results of automatic query formulation in terms of infotrons than providing keywords, as it exists now. Such automatic query formulation and the consequent results discovered in particular might give humanity the power to discover universal truths in a methodological way than expecting them to be invented by accidents.

6. The dream of any researcher would be to build infotrons and infotron dictionary for all the natural (spoken) languages. Assuming that one-day we are successful in doing so for all the natural languages available, then, knowledge store available in different languages can be subjected to machine translation almost instantaneously creating a universal knowledge store.

APPENDIX A

x	$y=x'$
0	1
1	0

Figure A-1 Truth table for NOT operation in \mathbf{L}_B

X	Y	$z = x \vee y$
0	0	0
0	1	1
1	0	1
1	1	1

Figure A-2 Truth table for OR operation in \mathbf{L}_B

X	Y	$z = x \wedge y$
0	0	0
0	1	0
1	0	0
1	1	1

Figure A-3 Truth table for AND operation in \mathbf{L}_B

Function $f(x,y)$	Name of the function
0	Inconsistency
$X' \wedge Y'$	NOR
$X \wedge Y$	
X'	NOT or Complement
$X \wedge Y'$	
Y'	NOT or Complement
$X' \wedge Y \vee X \wedge Y'$	Exclusive OR (Modulo 2 addition)
$X' \vee Y'$	NAND
$X \wedge Y$	AND
$X \wedge Y + X' \wedge Y'$	Equivalence
Y	
$X' \vee Y$	Implication
X	
$X \vee Y'$	Implication
$X \vee Y$	OR
1	Tautology

Figure A-4 Each row represents an operation $\in \Phi$ of G_B in L_B

REFERENCES

- [1] GRAY, MATTHEW K. 1996. *Web Growth Summary*. Available on WWW for online viewing at: <http://www.mit.edu/people/mkgray/net/web-growth-summary.html> (Last visited on 31-May-2002)
- [2] NUA considered as a recognized authoritative online source. 2001. Available on WWW for online viewing at: http://www.nua.ie/surveys/how_many_online/index.html (Last visited on 31-May-2002).
- [3] DANNY SULLIVAN ed., Available at WWW for online viewing at: <http://www.searchenginewatch.com> (Last visited on 31-May-2002)
- [4] IBM's "*The Clever Project*" Available on WWW for online viewing at: <http://www.almaden.ibm.com/cs/k53/clever.html> (Last visited on 31-May-2002).
- [5] SHENK DAVID; 1998; Revised and updated edition "*Data smog: Surviving the information glut*". Publisher: Harper San Francisco, June, 1998.
- [6] NEIL POSTMAN; 1993; "*Technopoly: The Surrender of Culture to Technology*". Publisher: Vintage Books, New York, March, 1993.
- [7] SADANANDA.R; Oct-Nov 1996, "*Challenges of Knowledge Discovery in Global Information Systems*", CSI-96, INDIA The emerging information technology giant, in Proceedings of the 31st Annual convocation of the Computer Society of India, Bangalore, pp 189-196
- [8] PAUL WADDINGTON. 1996. "*Dying for Information? A report on the effects of information overload in the UK and worldwide*" Reuters, United Kingdom. Available on WWW for online viewing at: <http://www.cniorg/regconfs/1997/ukoln-content/repor~13.html> (Last visited on 31-May-2002).
- [9] DREILINGER DANIEL, HOWE ADELE E Jul 1997, "*Experiences with selecting search engines using Metasearch*", ACM Transactions on Information Systems, Volume 15, Number 3, pp 195- 222.
- [10] W3C <http://www.w3c.org> .The official website of World Wide Web Consortium.

- [11] GEORGE COULOURIS, JEAN DOLLIMORE, TIM KINDBERG; 2001. "*Distributed Systems - Concepts and Design*", 3rd edition, Publisher: Addison-Wesley, 2001.
- [12] A.S. TENENBAUM .1995. "Distributed Operating System". Publisher: Prentice Hall, 1995.
- [13] VINTON G CERF; 1996. "*How the Internet Really Works - A Modest Analogy*" Available on WWW for online viewing at: http://www.worldcom.com/about_the_company/cerfs_up/technical_writings/hownetworks.phtml (Last visited on 31-May-2002)
- [14] JOHN KERMAN; 1997 "*Historical Distributed Systems: The ARPANET*" ICSA750. Available on WWW for online viewing at: <http://www.isc.rit.edu/~ijk6933/paper1.htm> (Last visited on 31-May 2002)
- [15] DANIELS, P.J. 1986. Cognitive Models in Information Retrieval: An Evaluative Review. *Journal of Documentation*. 1986 December; 42(4): 272-304. ISSN: 0022-0418.
- [16] BRAJNIK, GIORGIO;GUIDA, GIOVANNI; TASSO, CARLO. 1987. User Modeling in Intelligent Information Retrieval. *Information Processing & Management*. 1987; 23(4): 305-320. ISSN: 0306-4573.
- [17] OBERQUELLE, HORST. 1984. On Models and Modelling in Human Computer Co-operation. In: Veer, Gerrit C. van der; Tauber, M.J.; Green, T.R.G.; Gorny, P., eds. *Readings on Cognitive Ergonomics: Mind and Computers*. Berlin, West Germany: Springer-Verlag; 1984. 26-43. ISBN: 0-387-13394-1.
- [18] MURRAY, DIANNE M. 1987. Embedded User Models. BULLINGER, H. J.; SHACKEL, B., eds 1987. *Human-Computer Interaction-INTERACT'87*. Amsterdam, The Netherlands: Elsevier Science Publishers B.V. (North-Holland); 1987. 229-235. ISBN: 0-444-70304-7.
- [19] WHITEFIELD, ANDY. 1987. Models in Human Computer Interaction: A Classification With Special Reference to Their Use in Design BULLINGER, H. J.; SHACKEL, B., eds 1987. *Human-Computer Interaction-INTERACT'87*. Amsterdam, The Netherlands: Elsevier Science Publishers B.V. (North-Holland); 1987. ISBN: 0-444-70304-7.57-63.
- [20] LYYTINEN, KALLE. 1987. Two Views of Information Modelling. *Information & Management*. 1987 January; 12(1): 9-19. ISSN: 0378-7206.
- [21] BRUNER, JEROME S.; GOODNOW, JACQUELINE J.; AUSTIN, GEORGE A. 1956. *A Study of Thinking*. New York, NY: John Wiley and Sons; 1956. 330p

- [22] ROSCH, ELEANOR; LLOYD, BARBARA B., eds. 1978. *Cognition and Categorization*. Hillsdale, NJ: Lawrence Erlbaum, Associates; 1978. 328p. ISBN: 0-89859-433-2.
- [23] SMITH, EDWARD E.; MEDIN, DOUGLAS L. 1981. *Categories and Concepts*. Cambridge, MA: Harvard University Press; 1981. 203p. ISBN: 0-674-15725-7.
- [24] BARSALOU, LAWRENCE W. 1989. Intraconcept Similarity and Its Implications for Interconcept Similarity. VOSNIADOU, STELLA, ORTONY, ANDREW, eds. *Similarity and Analogical Reasoning*. Cambridge, England: Cambridge University Press; ISBN: 0-521-36295-4. pp76-121.
- [25] VOSNIADOU, STELLA, ORTONY, ANDREW, eds. 1989. *Similarity and Analogical Reasoning*. Cambridge, England: Cambridge University Press; 1989. 592p. ISBN: 0-521-36295-4.
- [26] ROSCH, ELEANOR. 1978. Principles of Categorization. ROSCH, ELEANOR; LLOYD, BARBARA B., eds. 1978. *Cognition and Categorization*. Hillsdale, NJ: Lawrence Erlbaum, Associates; 1978. ISBN: 0-89859-433-2. pp27-48.
- [27] KEIL, FRANK C. 1987. Conceptual Development and Category Structure. In: Neisser, Ulric, ed. *Concepts and Conceptual Development: Ecological and Intellectual Factors in Categorization*. Cambridge, England: Cambridge University Press; 1987. 175-200. ISBN: 0-521-32219-7.
- [28] MEDIN, DOUGLAS L. 1989. Concepts and Conceptual Structure. *American Psychologist*. 1989 December; 44(12): 1469-1481. ISSN: 0003-066X.
- [29] MEDIN, DOUGLAS L.; SHOBEN, EDWARD J. 1988. Context and Structure in Conceptual Combination. *Cognitive Psychology*. 1988; 20:158-190. ISSN: 0010-0285.
- [30] HUMPHREY, R. 1985. How Work Roles Influence Perception: Structural Cognitive Processes and Organizational Behavior. *American Sociological Review*. 1985; 50(2): 242-252. ISSN: 0003-1224.
- [31] VICKERY, B.C. 1986. Knowledge Representation: A Brief Review. *Journal of Documentation*. 1986 September; 42(3): 145-159. ISSN: 0022-0418.
- [32] CHAFFIN, ROGER; HERRMANN, DOUGLAS J. 1984. The Similarity and Diversity of Semantic Relations. *Memory and Cognition*. 1984; 12(2):134-141. ISSN: 0090-502X.

- [33] CHAFFIN, ROGER; HERRMANN, DOUGLAS J. 1987. Relation Element Theory: A New Account of the Representation and Processing of Semantic Relations. In: Gorfein, David S.; Hoffman, Robert R., eds. *Memory and Learning: The Ebbinghaus Centennial Conference*; 1985 April 19-20; New York, NY. Hillsdale, NJ: Lawrence Erlbaum Associates; 1987. 221-245. ISBN: 0-89859-653-X
- [34] WINSTON, MORTONE.; CHAFFIN, ROGER; HERRMANN, DOUGLAS, 1987. A Taxonomy of Part-Whole Relations *Cognitive Science*. 1987; 11: 417-444. ISSN: 0364-0213.
- [35] SCHUMACHER, ROBERT M.; GENTNER, DEDRE. 1988. Transfer of Training as Analogical Mapping. *IEEE Transactions on Systems, Man, and Cybernetics*. 1988 July/August; 18(4): 592-599. ISSN: 0018-9472.
- [36] MIYAKE, N. 1986. Constructive Interaction and the Iterative Process of Understanding. *Cognitive Science*. 1986 April-June; 10(2): 151-177. ISSN: 0364-0213.
- [37] LYYTINEN, KALLE. 1987. Different Perspectives on Information Systems: Problems and Solutions. *ACM Computing Surveys*. 1987 March; 19(1): 5-46. ISSN: 0360-0300.
- [38] BELKIN, N.J.; SEECER, T.; WERSIG, G. 1983. Distributed Expert Problem Treatment as a Model for Information System Analysis and Design. *Journal of Information Science*. 1983 February; 5(5): 153-167. ISSN: 0165-5515.
- [39] BELKIN, N.J. 1984. Cognitive Models and Information Transfer. *Social Science Information Studies*. 1984 April/July; 4(2 & 3): 111-129. ISSN: 0143-6236.
- [40] FAROOQ, MOHAMMAD U.; DOMINICK, WAYNE D. 1988. A Survey of Formal Tools and Models for Developing User Interfaces. *International Journal of Man-Machine Studies*. 1988; 29: 479-496. ISSN: 0020-7373.
- [41] REISNER, PHYLLIS. 1981. Formal Grammar and Human Factors Design of an Interactive Graphics System. *IEEE Transactions on Software Engineering*. 1981; 7(2): 229-240. ISSN: 0098-5589.
- [42] REISNER, PHYLLIS. 1984. Formal Grammar as a Tool for Analyzing Ease of Use: Some Fundamental Concepts. In: Thomas, John C.; Schneider, Michael L., eds. *Human Factors in Computer Systems*. Norwood, NJ: Ablex Publishing Corporation; 1984. 53-78. ISBN: 0-89391-146-1.

- [43] PAYNE, S.J.; GREEN, T.R.G. 1986. Task Action Grammars: A Model of the Mental Representation of Task Languages. *HCI*. 1986; 2: 93-133. ISSN: 0737-0024.
- [44] CARD, STUART K.; MORAN, THOMAS P.; NEWELL, ALLEN. 1983. *The Psychology of HCI*. Hillsdale, NJ: Lawrence Erlbaum Associates; 1983. 469p. ISBN: 0-89859-243-7.
- [45] LEVESON, NANCY G.; WASSERMAN, ANTHONY I.; BERRY, DANIEL M. 1983. *BASIS: A Behavioral Approach to the Specification of Information Systems*. *Information Systems*. 1983; 8(1): 15-23. ISSN: 0306-4379.
- [46] GREEN, THOMAS R.G.; SCHIELE, FRANZ; PAYNE, STEPHEN J. 1988. Formalisable Models of User Knowledge in Human-Computer Interaction. In: Veer, Gerrit C. van der; Green, et.al., eds. *Working With Computers: Theory Versus Outcome*. London, England: Academic Press; 1988. 346. ISBN: 0-12-711705-9.
- [47] KIERAS, D.E.; POLSON, P.G. 1985. An Approach to the Formal Analysis of User complexity. *Intl. Journal of Man-Machine Studies*. 1985 April; 22(4): 365-394. ISSN: 0020-7373.
- [48] PAYNE, S.J.; GREEN, T.R.G. 1989. The Structure of Command Languages: An Experiment on Task-Action Grammar. *International Journal of Man-Machine Studies*. 1989; 30: 213-234. ISSN: 0020-7373.
- [49] FROHLICH, DAVID M.; LUFF, PAUL. 1989. Some Lessons from an Exercise in Specification. *Human-Computer Interaction*. 1989; 4:121-147. ISSN: 0737-0024.
- [50] FOLEY, JAMES D.; VAN DAM, ANDRIES. 1983. *Fundamentals of Interactive Computer Graphics*. Reading, MA: Addison-Wesley Publishing Company; 1983. 664p. ISBN: 0-201-14468-9.
- [51] BOBROW, DANIEL G. 1975. Dimensions of Representation. In: Bobrow, Daniel G.; Collins, Andrew, eds. *Representation and Understanding*. New York, NY: Academic Press; 1975. 1-34. ISBN: 0-12-108550-3.
- [52] CODD, E.F. 1970. A Relational Model of Data for Large Shared Data Banks. *Communications of the ACM*. 1970 June; 13(6): 377-387. ISSN: 0001-0782.
- [53] CHEN, P. 1976. The Entity-Relationship Model: Toward a Unified View of Data. *ACM Transactions on Database Systems*. 1976 March; 1(1):9-36. ISSN: 0362-5915.

- [54] DAVIS, J. STEVE. 1990. Experimental Investigation of the Utility of Data Structure and E-R Diagrams in Database Query. *International Journal of Man-Machine Studies*. 1990; 32: 449-459. ISSN:0020-7373.
- [55] JARVENPAA, SIRKKA L.; MACHESKY, JEFREY J. 1989. Data Analysis and Learning: An Experimental Study of Data Modeling Tools. *International Journal of Man-Machine Studies*. 1989; 31: 367-391. ISSN: 0020-7373.
- [56] TSICHRITZIS, D.C.; LOCHOVSKY, F.H. 1982. *Data Models*. Englewood Cliffs, NJ: Prentice-Hall;1982. 381p. ISBN:0-13196-428-3.
- [57] HULL, RICHARD; KING, ROGER. 1987. *Semantic Database Modeling: Survey, Applications, and Research Issues*. *ACM Computing Surveys*. 1987 September; 19(3): 201-260. ISSN: 0360-0300.
- [58] PECKHAM, JOAN; MARYANSKI, FRED. 1988. *Semantic Data Models*. *ACM Computing Surveys*. 1988 September; 20(3): 153-189. ISSN: 0360-0300.
- [59] BROWN, A.W. 1989. From Semantic Data Models to object orientation in design databases. *Information and Software Technology*. 1989 Jan-Feb; 31(1): 39-46. ISSN: 0950-5849.
- [60] POTTER, W.D.; TRUEBLOOD, R.P. 1988. Traditional, Semantic, and Hyper-semantic Approaches to Data Modeling. *Computer*. 1988 June; 21(6): 53-63. ISSN: 0018-9162.
- [61] POTTER, W.D.; TRUEBLOOD, R.P.; EASTMAN, C.M. 1989. Hyper-semantic Data Modeling. *Data & Knowledge Engineering*. 1989; 4: 69-90. ISSN: 0169-023X.
- [62] HAMMER, M.; MACLEOD, D. 1981. Database Description With SDM: A Semantic Database Model. *ACM Transactions on Database Systems*. 1981 September; 6(3): 351-386. ISSN: 0362-5915.
- [63] CODD, E.F.1979. Extending the Relational Model to Capture More Meaning. *ACM Transactions on Database Systems*. 1979 December; 4(4): 397-434. ISSN: 0362-5915.
- [64] BOWERS, D.S. 1989. From Database to Information Base: Some Questions of Semantics and Constraints. *Information and Software Technology*. 1989 October; 31(8): 402-410. ISSN:0950-5849.
- [65] BRODIE, M.L. 1984. On the Development of Data Models. In: Brodie, M.L.; Mylopoulos, J.; Schmidt, J.W., eds. *On Conceptual Modelling: Perspectives From Artificial Intelligence*,

- Databases, and Programming Languages. New York, NY: Springer-Verlag; 1984. 19-48. ISBN: 0-387-90842-0.
- [66] SU, STANLEY Y.W. 1983. SAM*: A Semantic Association Model for Corporate and Scientific-Statistical Databases. *Information Sciences*.1983 May-June; 29(2&3):151-199. ISSN:0020-0255.
- [67] SU, STANLEY Y.W. 1986. Modeling Integrated Manufacturing Data with SAM*. *Computer*. 1986 January; 19(1): 34-49. ISSN:0018-9162.
- [68] Booch Grady. *Object-Oriented Design with Applications*. Melno Park CA: Publisher:Benjamin-Cummings,1991.
- [69] RICHARD P GABRIEL *Patterns of Software:tales from the software community*. Publisher: Oxford University Press. 1996.
- [70] ALI BAHRAMI 1999. *Object Oriented Systems Development*. Publisher: McGraw Hill.
- [71] CARROLL JOHN M . 1989. *Evaluation Description and Invention: Paradigm of HCI*. *Advances in Computers*. 1989; 47-77 ISSN 0020-7373.
- [72] SALTON, GERARD. 1986. Another Look at Automatic Text-Retrieval Systems. *Communications of the ACM*. 1986 July; 29(7): 648~656. ISSN: 0001-0782.
- [73] SALTON, GERARD; MCGILL, MICHAEL J. 1983. *Introduction to Modern Information Retrieval*, New York, NY: McGraw-Hill, 1983. 400p, ISBN: 0-07-054484-0.
- [74] SPARCK JONES, KAREN. 1974. Automatic Indexing. *Journal of Documentation* 1974;30(4):393-432 ISSN : 0022-0418.
- [75] BOOKSTEIN, ABRAHAM. 1985. Probability and Fuzzy-Set Applications to Information Retrieval. In: Williams, Martha E., ed. *Annual Review of Information Science and Technology(ARIST): Volume 20*. White Plains, NY: Knowledge Industry Publications, Inc. for the American Society for Information Science; 1985. 117-151. ISSN: 0066-4200; ISBN: 0-86729-175-3.
- [76] ROBERTSON, S. E. 1977. Theories and Models in Information Retrieval. *Journal or Documentation*. 1977 June; 33(2): 126-148. ISSN: 0022-0418.
- [77] SALTON,GERARD, 1979. Mathematics and Information Retrieval. *Journal of Documentation*, 1979; 35(1): 1-29. ISSN: 0022-0418.

- [78] VAN RIJSBERGEN, C. J. 1979. *Information Retrieval*. 2nd ed. London, England: Butterworths; 1979. 208p. ISBN: 0-408-70929-4.
- [79] SALTON, GERARD. 1968. *Automatic Information Organization and Retrieval*. New York, NY: McGraw-Hill; 1968. 514p.
- [80] WONG, S. K. M.; RAGHAVAN, VIJAY V. 1984. *Vector Space Model of Information Retrieval - A Re-evaluation*, VAN RIJSBERGEN, C. J., ed. 1984. *Research and Development in Information Retrieval: Proceedings of the 3rd Joint British Computer Society (BCS) and ACM Symposium*; July 2-6; Cambridge, England. Cambridge, England: Cambridge University Press; 167-185.
- [81] WONG, S. K. M.; ZIARKO, WOJCIECH. 1985. *On Generalized Vector Space Model in Information Retrieval*. *Annales Societatis Mathematicae Polonae: Series IV: Fundamenta Informaticae*. 1985; 8(2): 253-267.
- [82] SALTON, GERARD. 1981. *A Blueprint for Automatic Indexing*. *SIGIR Forum*. 1981; 16: 22-38. (A publication of ACM SIGIR).
- [83] BUCKLEY, CHRIS; LEWIT, ALAN F. 1985. *Optimization of Inverted Vector Searches*. 97-110. *Research and Development in Information Retrieval: [Proceedings of the] 8th Annual International ACM SIGIR Conference*; 1985 June 5-7; Montreal, Canada. New York, NY: ACM, Inc.; 1985. ISBN: 0-89791-159-8.
- [84] CROFT, W. BRUCE; PARENTY, THOMAS J. 1985. *A Comparison of a Network Structure and a Database System Used for Document Retrieval*. *Information Systems*. 1985; 10(4): 377-390. ISSN: 0306-4379.
- [85] SMEATON, A. F.; VAN RIJSBERGEN, C. J. 1981. *The Nearest Neighbor Problem in Information Retrieval: An Algorithm Using Upper Bounds*. *Proceedings of the ACM SIGIR, 4th International Conference on Information Storage and Retrieval*. 1981 May 31-June 2; Oakland, CA, New York, NY: ACM, Inc., 1981. 83-87. ISBN: 0-89791052-4.
- [86] CROFT, W. BRUCE; THOMPSON, ROGER H. 1987. *I³R: A New Approach to tile Design of Document Retrieval Systems*. *JASIS*. 1987. ISSN: 0002-8231.

- [87] FOX, EDWARD A. 1983. Extending the Boolean and Vector Space Models of Information Retrieval with P-Norm Queries and Multiple Concept Types. Ithaca, NY: Cornell University; 1983. (Ph.D. dissertation). 386p.
- [88] SALTON, GERARD. 1983. On the Representation of Query Term Relations by Soft Boolean Operators. In: Proceedings of the Association for Computational Linguistics (ACL) European Chapter 2nd Conference; 1985 March 27-29; Geneva, Switzerland. 116-122.
- [89] SALTON, GERARD. 1985. Some Characteristics of Future Information Systems. SIGIR Forum. 1985 Fall; 18(2-4): 28-39. (A publication ACM SIGIR).
- [90] SALTON, GERARD; VOORHEES, ELLEN M. 1985. Automatic Assignment of Soft Boolean Operators. Research and Development in Information Retrieval: [Proceedings of the] 8th Annual International ACM SIGIR Conference; 1985 June 5-7; Montreal, Canada. New York, NY: ACM, Inc.; 1985. 54-69.
- [91] SALTON, GERARD; FOX, EDWARD A.; WU, HARRY. 1983, Extended Boolean information retrieval. Communications of the ACM. 1983 November 26(11): 1022-1036. ISSN: 0001 -0782.
- [92] ROBERTSON, S. E.; SPARCK JONES, KAREN. 1976. Relevance Weighting of Search Terms. JASIS. 1976 May/June; 27(3): 129-146. ISSN: 0002-8231.
- [93] ROBERTSON, S. E. 1977. The Probability Ranking Principle in IR. Journal of Documentation. 1977 December; 33(4): 294-304. ISSN: 0022-0418.
- [94] HARPER, D. J.; VAN RIJSBERGEN, C. J. 1978. An Evaluation of Feedback in Document Retrieval Using Co-Occurrence Data. Journal of Documentation. 1978; 34(3): 189-216. ISSN: 0022-0418.
- [95] SPACK JONES, KAREN; WEBSTER , C.A. 1980. Research on relevance weighting. Cambridge, England : University of Cambridge, Computer, Laboratory; 1980. (British Library R&D Report 5553).
- [96] CROFT, W. BRUCE; HARPER, D. J. 1979. Using Probabilistic Models of Document Retrieval Without Relevance Information. Journal of Documentation. 1979 December; 35(4): 285-295. ISSN: 0022-0418.

- [97] CROFT, W. BRUCE. 1983. Experiments with Representation in a Document Retrieval System. *Information Technology: Research and Development*. 1983 January; 2(1): 1-21. ISSN: 0144-817X.
- [98] CROFT, W. BRUCE. 1984. A Comparison of Cosine Correlation. *Information Technology: Research and Development*. 1984; 3: 113-114. ISSN: 0144-817X.
- [99] FUHR, NORBERT. 1986. Two Models of Retrieval with Probabilistic Indexing. RABITTI, FAUSTO, ed.[Proceedings of the] (ACM) Conference on Research and Development in Information Retrieval; 1986 September 8-10; Pisa, Italy. 249-257.
- [100] VAN RIJSBERGEN, C, J. 1977. A Theoretical Basis for the Use of Co-occurrence Data in Information Retrieval. *Journal of Documentation*. 1977 June; 33(2): 106-119. ISSN: 0022-0418.
- [101] YU, C. T.; LUK, W. S.; SIU, M. K. 1979. On models of information retrieval Processes. *Information Systems*. 1979; 4(3): 205-218. ISSN: 0306-4379.
- [102] YU, C.T.; BUCKLEY, CHRIS; LAM, K.;SALTON,GERARD. 1983. A Generalized Term Dependence Model in Information Retrieval, *Information Technology: Research and development*. 1983 October; 2(4):129-154. ISSN: 0144-817X.
- [103] VAN RIJSBERGEN, C. J.; ROBERTSON, S. E.; PORTER, M. F. 1980. *New Models in Probabilistic Information Retrieval*. Cambridge, England: University of Cambridge, Computer Laboratory; 1980. 123p. (British Library R&D Report 5587).
- [104] MCGILL, MICHAEL J.; HUITFELDT, JENNIFER. 1979. Experimental Techniques of Information Retrieval, In: Williams, Martha E., ed. ARIST: Volume 14. White Plains, NY: Knowledge Industry Publications, Inc. for the American Society for Information Science; 1979. 93-127. ISSN: 0066-4200; ISBN:0-914236-44-X.
- [105] MCGILL, MICHAEL J.; KOLL, MATTHEW B.; NOREAULT, TERRY. 1979. An Evaluation of Factors Affecting Document Ranking by Information Retrieval Systems. Syracuse, NY: Syracuse University, School of Information Studies; 1979. (Technical Report).

- [106] SNEATH, PETER H. A.; SOKAL, ROBERT R. 1973. Numerical Taxonomy: The Principles and Practice of Numerical Classification. San Francisco, CA: W.H.Freeman; 1973. 573p. ISBN: 0-7167-0697-0.
- [107] CHARNIAK, EUGENE; MCDERMOTT, DREW. 1985. Introduction to Artificial Intelligence. Reading, MA: Addison-Wesley; 1985. 701p. ISBN: 0-201-11945-5.
- [108] WALKER, DONALD E.; HOBBS, JERRY R, 1981. Natural Language Access to Medical Text., Menlo Park, CA: SRI International, Artificial Intelligence Centre, Computer Science and Technology Division; 1981 March, 20p, (Technical Note 240; Project 1944).
- [109] SIMMONS, ROBERT F. 1987. A Text Knowledge Base from the AI handbook, Information Processing & Management. 1987. ISSN: 0306-4573.
- [110] SCHANK, ROGER C. 1975. Conceptual Information Processing. Amsterdam, The Netherlands: North-Holland; New York, NY: American- Elsevier; 1975. 374p. ISBN: 0-444-10773-8
- [111] SCHANK, ROGER C.; ABELSON, ROBERT P. 1977. Scripts, Plans, Goals, and Understanding: An Inquiry into Human Knowledge Structures. Hillsdale, NJ: L. Erlbaum Associates; 1977. ISBN: 0-470-99033-3.
- [112] VAN RIJSBERGEN, C. J. 1986. A New Theoretical Framework for Information Retrieval. RABITTI, FAUS'FO, ed. [Proceedings of the] ACM Conference on Research and Development in Information Retrieval; 1986 September 8-10; Pisa, Italy.194-200.
- [113] TONG, RICHARD M.; SHAPIRO, DANIEL G. 1985. Experimental Investigations of Uncertainty in a Rule-Based System for Information Retrieval. International Journal of Man-Machine Studies. 1985; 22(3): 265~282. ISSN: 0020-7373.
- [114] TONG, RICHARD M.; ASKMAN, VICTOR N.; CUNNINGHAM, JAMES F.; TOLLANDER, CARL J. 1985. Research and Development in Information Retrieval: [Proceedings of the] 8th Annual International ACM SIGIR Conference; 1985 June 5-7; Montreal, Canada. New York, NY: ACM, Inc.; 1985. 243-251.
- [115] SHOVAL, PERETZ. 1985. Principles, Procedures and Rules in an Expert System for Information Retrieval. Information Processing & Management. 1985; 21(6): 475-487. ISSN: 0306-4573.

- [116] BELKIN, N. J.; ODDY, R. N.; BROOKS, H. M. 1982. ASK for Information Retrieval: Part I. Background and Theory; Part II: Results of a Design Study. *Journal of Documentation*. 1982 June; September; 38(2-3): 61-71; 145-164. ISSN: 0022-0418.
- [117] BELKIN, N.J.; KWASNIK, B.H. 1986. Using Structural Representations of Anomalous States of Knowledge (ASK) for Choosing Document Retrieval Strategies. *Research and Development in Information Retrieval: [Proceedings of the] ACM Conference*; 1986 September 8-10; Pisa, Italy. ISBN: 0-89791-187-3. RABITTI, FAUSTO, ed. 11-22.
- [118] ARNOW, DAVID M.; TENENBAUM, AARON M.; WU, CONNIE, 1985, P-Trees: Storage Efficient Multiway Trees. 111-121; *Research and Development in Information Retrieval: [Proceedings of the] 8th Annual International ACM SIGIR Conference*; 1985 June 5-7; Montreal, Canada. New York, NY: ACM, Inc.; 1985. ISBN: 0-89791-159-8.
- [119] CARROLL, DAVID M.; POGUE, CHRISTINE A.; WILLETT, PETER. 1987. Bibliographic Pattern Matching Using the ICL Distributed Array Processor. *Journal of the American Society for Information Science*. 1987. ISSN: 0002-8231.
- [120] HOLLAAR, LEE A. 1984. The Utah Text Retrieval Project. A Status Report VAN RIJSBERGEN, C. J., ed. 1984. *Research and Development in Information Retrieval: Proceedings of the 3rd Joint British Computer Society (BCS) and Association for Computing Machinery (ACM) Symposium*; 1984 July 2-6; Cambridge, England. Cambridge, England: Cambridge University Press; (The British Computer Society Workshop Series). ISBN: 0-521-26865-6. 123-132.
- [121] MOHAN, KONDRAHALLI C.; WILLETT, PETER. 1985. Nearest Neighbor Searching in Serial Files Using Text Signatures. *Journal of Information Science*. 1985; 11(1): 31-39. ISSN: 0165-5515.
- [122] FOX, EDWARD A. 1987. Improved Retrieval Using a Relational Thesaurus Expansion of Boolean Logic Queries. In: Evens, Martha, ed. *Proceedings of a Workshop on Relational Models of the Lexicon*; 1984 July; Stanford, CA.
- [123] BATES, MARCIA J. 1986. Subject Access in Online Catalogs: A Design Model. *Journal of the American Society for Information Science (JASIS)*. 1986 November; 37(6): 357-376. ISSN:0002-8231.

- [124] GRIFFITHS, ALAN; LUCKHURST, H. CLAIRE; WILLETT, PETER. 1986. Using Inter document Similarity Information in Document Retrieval Systems. *JASIS*. 1986 January; 37(1): 3-11. ISSN: 0002-8231.
- [125] WILLETT, PETER. 1984. A Note on the Use of Nearest Neighbors for Implementing Single Linkage Document Classifications. *JASIS*. 1984 May; 35(3): 149-152. ISSN: 0002-8231.
- [126] VOORHEES, ELLEN M. 1986. The Efficiency of Inverted Index and Cluster Searches. RABITTI, FAUSTO, ed. [Proceedings of the] ACM Conference on Research and Development in Information Retrieval; 1986 September 8-10; Pisa, Italy. 164-174.
- [127] ODDY, R. N. 1977. Information Retrieval through Man-Machine Dialogue. *Journal of Documentation*. 1977; 33: 1-14. ISSN: 0022-0418.
- [128] FREI, H. P.; JAUSLIN, J. F. 1983. Graphical Presentation of information and Services: A User-Oriented Interface, *Information Technology: Research and Development*. 1983 January; 2(1): 23-42. ISSN: 0144-817X.
- [129] COHEN, PAUL R.; KJELDSSEN, RICK. 1987. Information Retrieval by Constrained Spreading Activation in Semantic Networks. *Information Processing & Management*. 1987. ISSN: 0306-4573.
- [130] RAU, LISA F. 1987. Knowledge Organization and Access in a Conceptual Information System. *Information Processing & Management*. 1987. ISSN: 0306-4573.
- [131] CROFT, W. BRUCE; THOMPSON, ROGER H. 1984. The Use of Adaptive Mechanisms for Selection of Search Strategies in Document Retrieval Systems. See reference: *Research and Development in Information Retrieval: Proceedings of the 3rd Joint British Computer Society (BCS) and Association for Computing Machinery (ACM) Symposium*; 1984 July 2-6; Cambridge, England. Cambridge University Press; (The British Computer Society Workshop Series). ISBN: 0-521-26865-6; VAN RIJSBERGEN, C. J., ed. 1984, 95-110.
- [132] MCCALL, FIONA M.; WILLETT, PETER, 1986. Criteria for the Selection of Search Strategies in Best Match Document Retrieval Systems. *International Journal of Man-Machine Studies*. 1986 September; 25(3): 317-326. ISSN: 0020-7373.

- [133] KOLL, MATTHEW B.; NOREAU, TERRY; MCGILL, MICHAEL J. 1984. Enhanced Retrieval Techniques on a Microcomputer. In: Williams, Martha E.; Hogan, Thomas H., comps. Proceedings of the National Online Meeting; 1984 April 10-12; New York, NY. Medford, NJ: Learned Information, Inc.; 1984. 165-170. ISBN: 0-938734-07-5.
- [134] SRINIVAS NARASIMHA KINI, BABY M.D., POULOSE JACOB K.; 1999; "Information Discovery: A paradigm for information exploration in a networked environment"; CALIBER 2000; 7th National Convention with the theme "Information Services in a networked environment in India"; February 15-18; Chennai, India.
- [135] H.A. PROPER ; P.D. BRUZA; 1999; "What is Information Discovery About?"; JASIS; 50(9); 737-750, 1999.
- [136] CLIFFORD A LYNCH. 1995 "Networked Information Discovery: A overview of current issues" (Invited Paper). IEEE Journal on selected areas of communications, 13(8), pp1502-1522.
- [137] MOOERS, C.N.1952 "Information retrieval viewed as temporal signalling". Proceedings of International Conference of Mathematicians, Cambridge, Massachusetts Aug 30 - Sep 6 1950. pp. 572-573. Providence, R.I.: American mathematical Society.
- [138] FAITHHORNE,R.A.1961. "Towards information retrieval". London:Butterworths
- [139] ELLIS DAVID.; ANA VASCONCELOS. 1999. "Ranganathan and the Net: using facet analysis to search and organise the World Wide Web". In: Aslib Proceedings volume 51 Number 1 January 1999. pp 3-10
- [140] A. EMTAGE, P. DEUTSCH; "Archie-An electronic directory service for the Internet," in Proceedings Winter 1992 USENIX Conf. Jan 20-24, 1992. San Francisco, CA. Berkeley, CA: USENIX, 1991. pp. 93-110.
- [141] P. DEUTSCH, "Resource discovery in an Internet environment-The Archie approach," Electronic Networking: Research, Applications, and Policy ". Vol. 2, No. 1, pp. 45-51, Spring 1992.
- [142] L. MCGILLIS, "Gopher searching using VERONICA". Reference Librarian, no. 41-42, pp. 25-35, 1994.

- [143] WEB-WALKERS and WEB_CRAWLERS: See <http://home.netscape.com/home/internet-search.html>. (Last Visited on 31-May-2002).
- [144] M. DILLON ET AL., "Assessing information on the Internet: Toward providing, library services for computer-mediated communication. Results of an OCLC research project". *Internet Research*, vol.3, no. 1, pp. 54-69, Spring 1993.
- [145] TOPNODE project: See <ftp.cni.org> for information on the TOPNODE project. (Last Visited on 31-May-2002).
- [146] E. J. CHRISTIAN, "Helping the public find information: The U.S. government information locator service (GILS)", *Journal Government Information*, Vol 21, n0 4, pp 305-314, July/Aug. 1994.
- [147] B. KAHLE ET AL., 1992 "Wide area information servers: An executive information system for unstructured files", *Electronic Networking: Research, Applications, and Policy*. Vol. 2, No.1, pp. 59-68, Spring 1992.
- [148] B. KAHLE, H. MORRIS, J. GOLDMAN, T. ERICKSON, A. HERTZ, J. CURRAN. 1992 "Interfaces for distributed systems of information servers", in *Proceedings ASIS 1992 Mid-Year Meeting, Networks, Telecommunications and the Networked Information Resource Revolution*, May 27-30, 1992. Albuquerque, NM. Silver Spring, MD: American Society for Information Science, 1992, pp. 124-148.
- [149] NATIONAL INFORMATION STANDARDS ORGANIZATION (NISO), ANSI/NISO Z39.50-1994, *Application Service Definition and Protocol Specification*. Bethesda. MD: NISO Press, 1995. Available online at <ftp://ftp.loc.gov>.
- [150] C. M. BOWMAN, P. B. DANZIG, D. R. HARDY, U. MANBER, M. F. SCHWARTZ; 1994 "The harvest information discovery and access system", in *Proceedings of Second International World Wide Web Conference*, Chicago, IL. Oct. 1994, pp. 763-771. See also: C. M. Bowman. P. B. Danzi., D. R. Hardy, U. Manber, M. F. Schwartz, and D. P. Wessels, "Harvest: A scalable, customizable discovery and access system", Technical Report CU-CS-732-94, Department of Computer Science, University of Colorado, Boulder, CO, 1994.

- [151] C. M. BOWMAN, P. B. DANZIG, U. MANBER, M. F. SCHWARTZ; “Scalable internet resource discovery: Research problems and approaches”, *Communications of ACM*, vol. 37, no. 8, pp. 98-107, Aug. 1994.
- [152] W. Y. ARMS ET AL., “The design of the Mercury electronic library”, *EDUCOM Review*, vol. 27, no. 6, pp. 38-41, Nov./Dec. 1992. See also M. Kibby and N. H. Evans, “The network is the library”, *EDUCOM Review*, vol. 24, no. 3, pp. 15-20, Fall 1989.
- [153] P. B. DANZIG, L. SHIH-HAO, AND K. OBRACZKA, “Distributed indexing of autonomous internet services”, *Computing Systems*, vol. 5, no. 4, pp. 433-459, Fall 1992.
- [154] P. B. DANZIG, J. AHN, J. NOLL, AND K. OBRACZKA, “Distributed indexing: A scalable mechanism for distributed information retrieval”, in *Proceedings of 14th Annual International ACM/SIGIR Conference Research and Development in Information Retrieval*, Chicago, IL, Oct. 13-16, 1991. Special issue of *SIGIR Forum*, 1991, pp. 220-229.
- [155] D. R. HARDY AND M. F. SCHWARTZ, “Essence: A resource discovery system based on semantic file indexing”, in *Proceedings of Winter 1993 USENIX Conference*, Jan. 25-29, 1993, San Diego, CA. Berkeley, CA: USENIX Association, 1993, pp. 361-373.
- [156] CAMEGIE MELLON UNIVERSITY. Lycos, The Catalog of the Internet. Available on WWW for online viewing at: <http://lycos.cs.cmu.edu> (Last visited on 31-May-2002)
- [157] COMMUNICATIONS OF THE ACM 35:12, Special Issue on Information filtering, December. 1992.
- [158] STANFORD UNIVERSITY, Stanford Information Filing Tool Available on WWW for online viewing at : <http://sift.stanford.edu>. See: The Paper on SIFT.
- [159] Google’s Architecture. Available on WWW for online viewing at <http://www-db.stanford.edu/~backrub/google.html> (Last visited on 31-May-2002).
- [160] TIM BERNERS-LEE; JAMES HENDLER; ORA LASSILA; *The Semantic Web* in *Scientific American* May 2001 Issue.
- [161] SRIKANTHA 1998. “*Data Mining and data Visualisation system for the Shimoga constituency loksabha election 98*”, An unpublished dissertation work for Master of Computer Applications.

- [162] KINI, SRINIVAS NARASIMHA; SRIKANTHA; JACOB,POULOSE K. 1999. "*A system for analysis and visualization of Domain specific Data*". In: Proceedings of the First International Conference on Enterprise Information System, Volume I; 1999 March 27-30; Setubal, Portugal, 1999. pp 82-90, ISBN 972-98050-0-8.
- [163] KINI, SRINIVAS NARASIMHA, 1995, *Failure analysis and software implementation of air data computer*.A dissertation work carried out in NAL(National Aerospace Laboratories).Bangalore.
- [164] ROBERT LOSEE; 1997; "*A discipline independent definition of information*" Journal of the American Society for information Science; Volume 48; pp 254-269.
- [165] G. BHATTACHARYYA; 1997; "*Information: its definition for its service professionals*" Library Science with a slant to documentation and Information studies; Volume 34; No., 2; 1997 Paper C; pp 69-83.
- [166] ANATOL RAPOPORT; 1953; "*What is Information*"; ETC: A review of general semantics; Volume 10, No., 4; 1953. Also available in "Introduction to information science" compiled and edited by Tefko Saracevic published by R.R.Bowker Company NY and London ISBN 0-8352-0313-1.
- [167] ROBERT A FAITHORNE; 1967; "*Morphology of Information Flow*"; Journal of the ACM Volume 14; No., 4; October 1967. Also available in "Introduction to information science" compiled and edited by Tefko Saracevic published by R.R.Bowker Company New York and London ISBN 0-8352-0313-1.
- [168] KINI, SRINIVAS NARASIMHA; JACOB, POULOSE K. 2000. "*An approach towards quantifying information*" in the proceedings of the 8th International Conference on Advanced Computing and Communications, Recent Advances in computing and communications. Eds Bhabani P.Sinha et.al, 2000 December 14-16, Cochin, India, 2000; pp 291-294, ISBN 0-07-043548-0
- [169] NYQUIST, H.; 1924; "*Certain Factors Affecting Telegraph Speed.*"; Bell System Technical Journal, Volume 3, April, 1924, p 324.
- [170] NYQUIST, H.; 1928; "*Certain Topics in Telegraph Transmission Theory*" A.I.E.E. Transactions; Volume 47, 1928, p617.

- [171] CLAUDE E SHANNON; 1948; "*A mathematical theory of communication*". Bell System Technical Journal, Volume 27, July 1948;pp 379-423 Also available in "Key papers in the development of information theory" ed. by David Slepian 1973 IEEE Press ISBN 0-83942-027-8 (Cloth bound); 0-87942-028-6 (paper bound); 1974; pp 5-18
- [172] CHOMSKY N; 1956; "*Three models for the description of language*" ; IRE Trans. on Information Theory 2:3 pp 113-124; 1956.
- [173] CHOMSKY N.; 1959; "*Certain formal properties of grammar*"; Information and control 2:2 pp 137-167; 1959.
- [174] TERRENCE PRATT; MARVIN ZELKOWITZ Programming Languages, by, Prentice Hall, 4th edition, 2001. ISBN 0-13-027678-2.
- [175] W3C Specification for HTML from the site <http://www.w3.org/TR/1999/REC-html401-19991224/> (Last visited on 31-May-2002).
- [176] W3C Specification for XML from the site <http://www.w3.org/TR/2002/WD-xml11-20020425/> (Last Visited on 31-May-2002)
- [177] IAN GRAHAM The HTML 4.0 Sourcebook, 1998 Publisher: John Wiley and Sons ISBN 0-471-25724-9
- [178] DAVID MEGGINSON,1998 *Structuring XML Documents*, Publisher Prentice Hall ISBN 0-13-642299-3.

AUTHOR INDEX

A		F	
Alan Emtage.....	56	Faithhorne	67
Art St. George	55	Farooq	21
B		Foley	23
Barsalou	17	Fox	47
Bates.....	47	Frohlich	23
Belkin.....	20, 45	Fuhr	42
Billy Barron	55	G	
Bobrow.....	24	Gentner.....	19
Bookstein	46	Green.....	22, 23
Bowers	31	H	
Brajnik.....	13	Hammer.....	30
Brodie.....	31	Harper	42
Brown.....	28	Herrman	18
Bruner	17	Hobbs	43
Bruza	55, 66, 67	Hull	26, 27
C		Humphrey	18
Card.....	22	J	
Carroll	34	Jarvenpaa.....	24
Chaffin	18	K	
Chen	24	Keil.....	17
Codd.....	24, 31	Kieras	23
Croft	46	King.....	26, 28
D		Kwasnik	45
Daniel Drelinger.....	4	L	
Daniels	13, 20	Leveson.....	22
David Shenk.....	3	Lloyd.....	17
Davis	24	Lochovsky.....	25
Dominick.....	21	Luff	23
E		Lynch	54, 55, 60, 66
Ellis David	69	Lyytinen	14, 19

M		S	
Machesky	24	Sadananda	4
Macleod.....	30	Salton	46
Maryanski	26, 27	Schank.....	43
Medin	17	Schumacher.....	19
Miyake	19	Schwartz.....	60
Moores	67	Shannon.....	6, 88, 89, 94
Moran	21, 22, 23	Simmons	43
Murray.....	14	Smith	17
N		Soben.....	17
Neil Postman.....	3	Spark Jones	40
Nyquist.....	88	Srinivas Kini	64
O		Steve Kirsch	4
Oberquelle.....	14	Su	32
Ortony	17, 19	T	
P		Trueblood.....	28
Payne.....	22, 23	Tsichritzis.....	25
Peckham.....	26, 27	V	
Peter Deutsch	56	Van Dam	23
Polson.....	23	Van Rijsbergen.....	40, 42, 43
Potter	28	Vasconcelos Ana.....	69
Proper	55, 66, 67	Vickery.....	18
R		Vosniadou	17, 19
Reisner	21	W	
Robertson	40	Walker.....	43
Ron Larsen	55	Whitefield	14
Rosch.....	17	Winston	18
		Y	
		Yu.....	42

SUBJECT INDEX

- 3**
3-D graphs..... 78
- A**
agent..... 18, 59
AI..... 13, 18, 147
algorithmic specifications..... 21
AltaVista..... 53
analog..... 2
analogies..... 19
Analyzer..... 105, 123, 124, 127, 132
animations..... 78
Anomalous States of Knowledge..... 45
archie..... 56, 57, 59
architectural framework..... 2, 63, 86
Architecture of software systems..... 63
Artificial Intelligence..... 13
ASK..... 45
ASK structure..... 45
assembly segments..... 70, 78
atom..... 91
automated indexing..... 37
Automatic validation checks..... 75
autonomous indexing services..... 59
- B**
ballot boxes..... 70
BASIS..... 22
BBN..... 55
Behavioural Approach to the
 Specification of Information Systems
 22
bibliographic IR systems..... 20
binary language..... 93, 94
BNF..... 21, 22, 23
browser..... 90, 142
bulletin boards..... 52
- C**
canned query..... 77
categorization..... 17, 68, 69
category membership..... 17, 18
channel..... 95
charge..... 91
classification .. 18, 26, 28, 35, 54, 56, 60,
 61, 69, 72, 77, 82, 83, 146
Classifier and Categorizer..... 123
clean data..... 75
CLG..... 21, 23
cluster hierarchy..... 47
CNI..... 57
Coalition for Networked Information . 57
cognitive models..... 12, 15, 16, 33
cognitive psychology .. 12, 16, 18, 34, 66
Command Language Grammar..... 21
Component management..... 11
components 1, 11, 13, 14, 21, 26, 28, 29,
 33, 63, 71, 80, 83, 104, 105, 123, 124,
 125, 127, 129, 130, 131, 132
computer architectures..... 63
computer networks..... 4, 5
computer science..... 12, 63, 66
computer systems..... 1
computing power..... 2
computing technology..... 2
conceptual models..... 12, 15, 16
co-occurrence analysis..... 45
cosine correlation..... 40
counting tables..... 70
current flow..... 92
- D**
data entry..... 75
Data mining..... 71, 77
Data Smog..... 3
Data visualization..... 71, 131

data warehouse 71, 72, 73, 74, 75, 76, 77,
 79, 129, 130
 Data warehouse 71
 database structure 15
 database structures 15
 datalogical realm 25
 decomposition and structuring rules ... 11
 democracy 70
 design patterns 11
 design principles 11
 DIALOG 59
 digital 2, 4, 8, 66, 87
 digital form 4, 8, 87
 dimensions of representation 24
 directory of servers 58
 discipline of information 140, 145
 Discoverer .. 65, 124, 127, 128, 129, 130,
 132
 discovery process 113
 discrete mathematics 10
 Dispatcher 127, 128, 131, 132
 dissemination of information 7
 distributed information server
 architecture 58
 distributed systems 3, 9, 61, 69
 district head quarters 70
 DNS 11
 Document Object Modelling 131
 DOM 121, 131
 domain knowledge 49, 74, 136, 137
 Domain Naming System 11
 dynamic data 72, 73, 76, 79, 130

E

ECRS.... 6, 69, 70, 73, 77, 79, 80, 81, 82,
 84, 85, 86
 Election Counting and Reporting
 Software 6, 69
 election counting process 70
 electron 91
 electronic form 74
 electronics 2
 energy 92
 engineering practices 11

enterprise-wide systems 10
 Essence 60
 event driven 121, 141
 Excite 53
 EXPLAIN 58
 extended boolean retrieval 40, 42
 eXtensible Markup Language 5
 Extranet 61, 130

F

fault tolerance 72, 76, 79, 84, 85, 86
 feature-based technique 35, 43
 feedback techniques 50
 file organization 46
 finite-state diagram 23
 finite-state machine 23
 firmware 95
 first-order predicate calculus 43
 Forensic Systems 144
 formal language development 14
 forward chaining 43
 FTP 56, 57, 59
 Function Models 21
 fundamental information 5, 91
 fundamental information element 91
 fuzzy set 37, 38, 42

G

Gatherer 123, 127, 128, 132
 general communication system 88, 90, 94
 general information discovery system 90,
 91, 139
 Generalized Transition Networks 23
 Geographical Information System 4
 GILS 58
 GIS 4
 glass box 83
 Goals, Operators, Methods, Selection. 22
 GOMS 22, 23
 Google 53, 60
 Gopher 58
 Gopherspace 57

Government Information Locator
 Service..... 58
 grammar . 21, 22, 92, 93, 94, 95, 97, 105,
 120, 129
 granularity 61, 62, 66, 87, 100
 Graphical User Interface 133
 GTN 23
 GUI 133, 134

H

hardware. 1, 7, 9, 46, 63, 71, 83, 95, 114,
 131, 144
 Harvest 57, 59, 60
 historical data 73, 75
 HotBot 53
 HTML .. 4, 5, 53, 99, 100, 101, 103, 104,
 106, 107, 108, 115, 116, 119, 120,
 123, 140, 142
 humanity 3, 147
 Hyper Text Markup Language 5
 hyper-semantic 28

I

I³R 49
 ID ... 6, 54, 55, 61, 67, 68, 71, 73, 95, 97,
 133, 140
 ID front..... 133, 140
 IDLIS ... 7, 132, 133, 134, 135, 138, 141,
 145
 IDP 54, 64, 145
 IDS . 6, 7, 64, 67, 87, 100, 105, 121, 122,
 123, 125, 128, 129, 130, 131, 132,
 133, 135, 136, 139, 140, 141, 142,
 143, 144, 145, 146, 147
 IDS architecture 129
 IETF 57, 144
 IF 54, 67
 IID 143
 IM..... 13
 implementations 11
 infological realm 25
 information carrier 55
 information content..... 4, 123

information context..... 18
 information destination 89
 information discoverer67, 68, 69, 90, 95,
 105, 121, 123
 information discovery 5, 6, 54, 55, 56,
 57, 59, 61, 64, 66, 69, 87, 88, 90, 104,
 122, 133, 145
 Information Discovery6, 54, 64, 65, 121,
 126, 132, 133, 139, 140
 Information Discovery in Library
 Information System..... 7, 132, 133
 information discovery paradigm6, 61, 69
 Information Discovery Paradigm. 54, 64,
 65, 140
 information discovery system 64, 90
 Information Discovery System 7, 121,
 126
 information elements 122, 140
 information explosion 3, 6
 Information Filtering..... 54
 information flow 90, 92
 information glut 3
 information management techniques 4
 Information model 13, 14
 information model continuum..... 16, 91
 Information modeling 12
 Information modelling 14
 Information Retrieval..... 6, 54, 67
 information scarcity 3
 information science 12, 18, 66
 information scientists 5, 34
 information seeking 20
 information sink 90
 information smog 8
 information source 13, 60, 69, 89
 information space3, 8, 57, 64, 66, 67, 68,
 121, 122, 123, 128, 131, 147
 information storage 5
 information system..... 5, 11, 13, 15, 95
 Information Technology 88
 information theory 66, 88
 infotron..... 5, 91, 95, 96, 97, 98, 99, 100,
 105, 115, 117, 120, 121, 124, 128,

129, 130, 132, 135, 136, 137, 139, 144, 146, 148	
infotron dictionary ... 95, 96, 98, 99, 105, 121, 124, 128, 129, 130, 132, 135, 137, 139, 144, 146, 148	
infotrons . 5, 7, 91, 92, 93, 94, 95, 97, 98, 99, 100, 101, 102, 105, 106, 107, 108, 109, 111, 112, 113, 115, 119, 121, 122, 123, 124, 125, 127, 128, 129, 135, 136, 139, 140, 141, 145, 146, 147, 148	
Integrated Systems	51
intellectual cataloging	57
intelligent information discovery	128
interfaces	11, 34
Internet ... 3, 4, 6, 7, 9, 46, 47, 52, 53, 54, 55, 57, 60, 61, 64, 67, 89, 90, 130, 142, 143, 144, 147	
Internet Information Discovery	143
Internet2	53
Intranets	47
inverse document frequency weight ...	38
inverted file	39
IPv4	5
IPv6	5
IR.... 6, 13, 20, 35, 36, 37, 38, 46, 50, 54, 67, 77, 96, 122, 128	
IR techniques	35, 36, 37, 77, 128
IS 13, 16, 20, 21, 24, 25, 27, 28, 32	
IS design	20

J

Java	114, 131, 147
Java scripts	131

K

Keystroke-Level Model	22
KLM	22
knowledge engineers	5
knowledge intensive networks	49

L

language 5, 21, 53, 92, 93, 94, 95, 97, 99, 100, 105, 114, 120, 129, 131, 144, 145, 147	
learning by analogy	19
legacy systems	6, 141
Library Information System	7, 133
library science	66
linguistics	18
LIS	7, 133, 134, 135
Lycos	53, 60

M

mailing-lists	52
markup languages	99
mass	91
mathematical theory of communication	88
mechanical	2
mediated information retrieval	20
member functions	96
meta search engine	4
metadata	68, 71, 73, 76, 78
metaphors	19
microelectronics	1
microprocessors	1
modelling of information	6
modus operandi	144
Morse Code	89
multi processing	10
multi tasking	10
multi threading	10
multidimensional data relationships ...	77

N

naming scheme	11
NAND	94
natural language	43, 144, 148
Natural Language Processing	36
Network Operating System	71, 142

network servers	71
Networked computing machines	71
news groups	52
news-archives.....	52
NLP	36, 44
noise	53, 95
NOR	94
NSFNET	56
numerical taxonomy.....	43

O

object oriented analysis and design	96, 99
object oriented concepts.....	96
object oriented modelling	32
OCLC	57
OLAP.....	71, 77, 79, 131, 142
OnLine Analytical Processing	77
online visualization	78
OOAD	99
operator	75, 92, 93, 97, 146
organizational structure.....	18

P

parallel computing	1
parliamentary constituency	69
penta-tier architecture ..	6, 64, 70, 80, 81, 82, 86, 125, 129, 130, 132, 139, 142, 146
picture	78
political parties.....	74
polling booths.....	70
pragmatics	22, 24, 34
presentation layer	78
proactive.....	89, 90
probabilistic....	35, 37, 38, 40, 41, 42, 44, 46, 50
process of discovery.....	123
process of model building.....	14, 20
processor architecture	1
production systems.....	23
programming languages..	21, 63, 93, 147
protocols.....	9, 95

Q

Query gatherer.....	127, 129, 131
query logic formulation.....	46

R

ranking techniques	42
RD	66
RDBMS.....	123
reality mapping	14
Relational Model/Tasmanian	31
relational thesaurus	47
reliability	2, 72, 79, 84
Resource Discovery	66
retrieval technique	35, 38, 39, 40, 41, 42, 43, 45, 48, 49, 50, 51, 52, 61
RM/T.....	31
robots.....	52
rotate	78
RUBRIC.....	44

S

SAM*	32
Scanner.....	104, 105, 123, 124, 127, 129, 132
SDM.....	30
search engines	3, 4, 46, 52, 53, 60, 68, 87, 143, 147
Semantic Abstractions	26
Semantic Association Model-enhanced	32
semantic complexity	25, 26
semantic data models ..	12, 15, 25, 26, 28
Semantic Database Model.....	30
Semantic Hierarchy Model extended..	31
Semantic memory	16
semantic relations.....	18, 34
semantic structure	18
semantic web.....	61
semantics.....	21, 23, 32, 95
sequence-based analysis.....	77
session specific infotron dictionary...	123

SHM+	31
signal	53, 95
signaling theory	66
similarity	17, 18, 37, 39, 40, 43, 45, 47, 129
similarity measure	37, 39, 40, 43, 47
Simple Object Access Protocol	131
SIRE	51
SMART	39, 47
SOAP	131
software	1, 2, 5, 6, 9, 10, 32, 52, 63, 64, 65, 71, 76, 83, 89, 95, 114, 140, 144
software agents	52, 89
software architects	5
software architecture	2, 140
software components	2, 9, 71
software engineering	63
software field	1
software life cycle	1
software system	2, 6, 10, 32, 63
Space technology	4
special log file	76
spiders	52, 89, 128
spin	92
spreading activation	49
static data	73, 74, 75, 129, 130
statutory reports	70
stored data	71
stringent policy	75
Structure-Based Techniques	43
study of concepts	17
subcomponents	125
successful communication	95
supportive data	73, 86
symbols	95, 147
syntax	21, 23, 59
system architecture	58, 63, 71, 79, 81, 84
system engineering	11
system failure	124
system wide infotron dictionary	123, 124, 130
Systems engineering and design	11

T

TAG	22, 23
task-action grammar	22
telegraph speed	88
temporal logic	10
TeX	60
text compression	46
text-based systems	13
THOMAS	48
TOPNODE	57
truth tables	94
Turing Machines	10

U

UML	32
Uniform Resource Characteristics	57
Uniform Resource Locators	57
Uniform Resource Names	57
universal operations	94
URC	57
URI	87
URL	57, 87, 143
URN	57, 87
user interface	143
user models	13, 20
user specific infotron dictionary	123

V

vector space	37, 38, 40, 41, 42
Veronica	57
visualization	78, 79, 95
vocabulary	46, 47

W

W3C	61, 100, 144
WAIS	58
Web Services Discovery	143
web-based interface	123, 133
Web-walking systems	57
well formedness	123
Wide Area Information Server system	58
within document frequency weight	38
World Wide Web	4, 9, 61, 100

World Wide Web Consortium 61, 100
WWW 4, 7, 9, 11, 52, 57, 58, 60, 89,
104, 142, 143

X

XML5, 99, 100, 102, 103, 104, 106, 107,
109, 115, 117, 118, 119, 120, 121,

122, 123, 131, 133, 135, 137, 138,
139, 140, 141, 142, 143
XSL 142

Z

Z39.50 58

PUBLICATIONS OF THE AUTHOR

- [1] KINI, SRINIVAS NARASIMHA; SRIKANTHA; JACOB,POULOSE K. 1999. "A System for Analysis and Visualization of Domain Specific Data". Proceedings of the First International Conference on Enterprise Information System, Volume I; 1999 March 27-30; Setubal, Portugal, 1999.pp 82-90,ISBN 972-98050-0-8.
- [2] KINI, SRINIVAS NARASIMHA; BABY M.D.; JACOB,POULOSE K. 2000. "Information Discovery: A Paradigm for Information Exploration in a Networked Environment". Proceedings of the Seventh National Convention - CALIBER 2000, Information Services in a Networked Environment in India, R Vengan et al Eds; 2000 February 15-18; Chennai, India, 2000; pp 1.23- 1.29, ISBN : 81-900825-3-1.
- [3] KINI, SRINIVAS NARASIMHA; JACOB, POULOSE K. 2000. "XML: A New Way to Create Information Content on Internet". Proceedings of the Annual Seminar on Electronic Sources of Information I.K.Ravichandra Rao et al Eds; 2000 March 1-3; Bangalore, India, 2000; paper CL, pp 1-11.
- [4] KINI, SRINIVAS NARASIMHA; JACOB, POULOSE K. 2000. "XML for Creating Information Content on Internet" in *Information Studies* Volume 6 No 4, Neelameghan Ed.pp 241-254. ISSN 0971-6726
- [5] KINI, SRINIVAS NARASIMHA; JACOB, POULOSE K. 2000. "An Approach Towards Quantifying Information" Proceedings of the 8th International Conference on Advanced Computing and Communications, "Recent Advances in Computing and Communications". Bhabani P.Sinha et.al, Eds; 2000 December 14-16, Cochin, India, 2000; pp 291-294, ISBN 0-07-043548-0.

G 8533

