ABSTRACT

COMPLEX ADAPTIVE SYSTEMS BASED DATA INTEGRATION: THEORY AND APPLICATIONS

by

Eliahu Rohn

Data Definition Languages (DDLs) have been created and used to represent data in programming languages and in database dictionaries. This representation includes descriptions in the form of data fields and relations in the form of a hierarchy, with the common exception of relational databases where relations are flat. Network computing created an environment that enables relatively easy and inexpensive exchange of data. What followed was the creation of new DDLs claiming better support for automatic data integration. It is uncertain from the literature if any real progress has been made toward achieving an ideal state or limit condition of automatic data integration. This research asserts that difficulties in accomplishing integration are indicative of socio-cultural systems in general and are caused by some measurable attributes common in DDLs. This research's main contributions are: (1) a theory of data integration requirements to fully support automatic data integration from autonomous heterogeneous data sources; (2) the identification of measurable related abstract attributes (Variety, Tension, and Entropy); (3) the development of tools to measure them. The research uses a multi-theoretic lens to define and articulate these attributes and their measurements. The proposed theory is founded on the Law of Requisite Variety, Information Theory, Complex Adaptive Systems (CAS) theory, Sowa's Meaning Preservation framework and Zipf distributions of words and meanings. Using the theory, the attributes, and their measures, this research

proposes a framework for objectively evaluating the suitability of any data definition language with respect to degrees of automatic data integration.

This research uses thirteen data structures constructed with various DDLs from the 1960's to date. No DDL examined (and therefore no DDL similar to those examined) is designed to satisfy the law of requisite variety. No DDL examined is designed to support CAS evolutionary processes that could result in fully automated integration of heterogeneous data sources. There is no significant difference in measures of Variety, Tension, and Entropy among DDLs investigated in this research. A direction to overcome the common limitations discovered in this research is suggested and tested by proposing GlossoMote, a theoretical mathematically sound description language that satisfies the data integration theory requirements. The DDL, named GlossoMote, is not merely a new syntax, it is a drastic departure from existing DDL constructs. The feasibility of the approach is demonstrated with a small scale experiment and evaluated using the proposed assessment framework and other means. The promising results require additional research to evaluate GlossoMote's approach commercial use potential.

COMPLEX ADAPTIVE SYSTEMS BASED DATA INTEGRATION: THEORY AND APPLICATIONS

by

Eliahu Rohn

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

Department of Information Systems

January 2008

APPROVAL PAGE

COMPLEX ADAPTIVE SYSTEMS BASED DATA INTEGRATION: THEORY AND APPLICATIONS

Eliahu Rohn

Dr. Murray Turoff, Dissertation Co-advisor Distinguished Professor Emeritus of Information Systems, NJIT	Date
Dr. Michael Chumer, Dissertation Co-advisor Senior University Lecturer, Department of Information Systems, NJIT	Date
Dr. Jerry Fjermestad, Committee Member Associate Professor of Information Systems and Management, NJIT	Date
Dr. Robert Klashner, Committee Member Assistant Professor, Department of Information Systems, NJIT	Date
Dr. Vassilka Kirova, Committee Member Research Professor, Department of Information Systems, NJIT	Date
Dr. Denis Blackmore, Committee Member Professor, Department of Mathematical Sciences, NJIT	Date

BIOGRAPHICAL SKETCH

Author: Eliahu Rohn

Degree: Doctor of Philosophy

Date: January 2008

Graduate Education:

- Doctor of Philosophy in Information Systems
  New Jersey Institute of Technology, Newark, NJ, 2008
- Master of Science, Management
  New Jersey Institute of Technology, Newark, NJ, 1992

Major: Information Systems

Presentations and Publications:

Rohn, Eli and Pinnero, Andrew (2007) *Cutting Edge Cyber Fraud Prevention and Detection.* Internal Auditor, August 2007.

Rohn, Eli (2007) *A Survey of Schema Standards and Portals for Emergency Management.* Proceedings of the 4th International ISCRAM Conference, Netherland, May 2007.

Rohn, Eli (2007) *Developing SOX Compliant Software.* Ternton Computer Festival, The College of Trenton, New Jersey, April 29, 2007.

Rohn, Eli et. al., (2007) *Towards a Unified Public Safety Scale.* Proceedings of the 40th Hawaii International International Conference on Systems Science (HICSS-40), January 2007.

Rohn, Eli (2006) *Data Integration Potentiometer in DERMIS.* Proceedings of the 3rd International ISCRAM Conference, Newark, NJ (USA), May 2006.

Rohn, Eli and Klashner, Robert (2004) *Hidden Disorder in XML Tags*. Proceedings of the Americas Conference on Information Systems, New York, NY, August 2004.

Rohn, Eli (2002) *Predicting Context Aware Computing Performance*. Ubiquity, Volume 50 Number 3, ACM Press, February 2002.

Rohn, Eli (2002) *Usage of Low Cost Aviation Simulators*. IEEE Computer Society, North New Jersey Chapter, May 6th, 2002.

Rohn, Eli (2002) *XML: Practical Introduction Using Theory and Case Studies*. IEEE Computer Society, North New Jersey Chapter, March 19, 2002.

Rohn, Eli (2002) *XML Technologies and their Use in an E-Commerce Project*. ACM SIGAPP Symposium on Applied Computing, Madrid, Spain, March 2002.

Rohn, Eli (2001) *XML: Lessons Learned*. IEEE Computer Society, North New Jersey Chapter, April 19, 2001.

Rohn, Eli (2000) *Comments on the Electronic Signatures in Global and National Commerce Act*. New Jersey Technology Council, November 2000.

Rohn, Eli (2000) *XML Secrets for Managers and Engineers*. IEEE Computer Society, North New Jersey Chapter, September 14, 2000.

Rohn, Eli (2000) *Introduction to Multi-media Data Bases*. IEEE Computer Society, North New Jersey Chapter, May 11, 2000.

Rohn, Eli (1999) *Legal Pitfalls and Survival Techniques for Internet Users*. IEEE Computer Society, North New Jersey Chapter, November 9, 1999.

Rohn, Eli (1999) *Non-WWW Internet Services*. IEEE Computer Society, North New Jersey Chapter, August 10, 1999.

Rohn, Eli (1995) *Design and Implementation of Information Systems Using Adabas and Predict*, SPL, Or-Yehuda, Israel

Rohn, Eli (1988) *dBase for the Hebrew User*, Hod-Ami Publishers, Hertzelya, Israel.

Rohn, Eli (1987) *OpenAccess Self Study Guide*, Kalanit Publishers, Tel-Aviv, Israel

To my ancestors and to my parents

who endured the most horrific treatments devised by mankind

from Exile through the Inquisition to the Holocaust and back to the Promised Land

yet never gave up on their identity and purpose of life

providing a base of wisdom and endurance

upon which I now stand

TABLE OF CONTENTS

Chapter **Page**

TABLE OF CONTENTS
(Continued)

Chapter **Page**

TABLE OF CONTENTS
(Continued)

Chapter **Page**

TABLE OF CONTENTS
(Continued)

Chapter **Page**

TABLE OF CONTENTS
(Continued)

Chapter **Page**

TABLE OF CONTENTS
(Continued)

Chapter **Page**

TABLE OF CONTENTS
(Continued)

Chapter **Page**

TABLE OF CONTENTS
(Continued)

Chapter **Page**

LIST OF TABLES

Table **Page**

Table **Page**

LIST OF FIGURES

Figure **Page**

Figure **Page**

Figure **Page**

LIST OF SYMBOLS, NOMENCLATURE AND DEFINITIONS

ADABAS	A database management system made by Software AG.
CAS	Complex Adaptive Systems.
CSV	Comma Separated Values.
Data	Facts, measurements, or observations with or without context (Marakas 1999).
DBMS	Database Management System.
DDL	Data Definition Language. In this research it refers to syntax designed for the creation of data structures. This includes Cobol FD, Adabas, SQL, XML, XSD and many others.
EDI	Electronic Data Interchange.
Information	Data organized in such a manner as to be useful and relevant to a user (Marakas 1999). "A relationship between common sets of structured variety"(Buckley 1998) p. 41.
Knowledge	The application of rules, procedures, ideas and information to guide the actions of a decision maker (Marakas 1999).
Metadata	Structured, encoded data that describe characteristics of information-bearing entities to aid in the identification, discovery, assessment, and management of the described entities (ALCTS 1999).
MISMO	Mortgage Industry Standards Maintenance Organization.
SQL	Structured Query Language.
SWIFT	A banking standard used for foreign currency exchange and other transactions.

EXECUTIVE SUMMARY

Motivation

Data definition languages (DDLs) are used to create data structures. DDLs specify how to organize and interconnect related elementary pieces of data into useable structures. They come in three types: "structured," (e.g., Cobol, SQL) "semi-structured," (e.g., web pages, Word documents) and "unstructured" (e.g., images, voice). Data structures can differ on three aspects: their structure (which also implies level of details), field / tag names, and the syntax used to define the data structure. Organizations and individuals exhibit a need for data integration that has been growing since the early 1960's. Integration from independent and heterogeneous sources is becoming an expensive bottleneck, costing private, public and government organizations billions of US dollars a year in the US alone. Full automation of the task is highly desirable. However, achieving automatic data integration from such sources seems to be illusive as ever. Many careers have benefited from the rush to create a panacea to the need, only to be replaced by the next wave of hyperbole - more complicated and usually more expensive technology: additional layers or components are responsible for the added complexity; the need for additional communications bandwidth and processing power make the solution more expensive. For example, the introduction of XML in early 1998 was followed by the introduction of XLL (Extensible Linking Language) and XSL (XML Style Sheet) several months later. XMLNS (XML namespace) was released shortly thereafter, adding one more layer and processing time requirements. XPATH and XSLT (XSL Transformations) showed up as World Wide Web Consortium (W3C) recommendations in early 1999, again adding layers and requiring additional processing cycles. None of these technologies was able to

escape, circumvent or overcome the problem of language ambiguity. Most are poor in thorough analysis, often void of sound mathematical foundations, and littered with short lived solutions. This lead to what the Gartner Group calls the "hype cycle" - a model of the relative maturity of technologies in a certain domain (Knox and Abrams 2003; Knox 2004; Knox, Abrams et al. 2006). The solutions end up in some form of an electronic graveyard upon reaching the "disillusion stage" in Gartner's hype cycle. Such was the fate of the *Metadatabase* project, XLL, Xpath 1.0, XOP (XML-binary Optimized Packaging), OIL (Ontology Inference Layer), DAML (DARPA Agent Markup Language), DAML+OIL, CICA (Context Inspired Component Architecture), XVIF (XML Validation Interoperability Framework) and other proposed approaches and implementations, whose viability did not exceed roughly two years.

No theory of data integration exists. It is unknown what the theoretical necessary requirements are to fully support automatic data integration from autonomous heterogeneous data sources. Therefore, it is not possible to objectively evaluate if and how much new DDLs move towards meeting such theoretical requirements. Nor is it possible to suggest a better DDL that meets such theoretical requirements, because the requirements do not exist. One may view this research as a step towards the formulation of a mathematically sound data integration theory. It aims at getting a theoretic-based deep understanding of principle attributes that must exist in any DDL constructed for the purpose of enabling automatic data integration. This work proposes a data integration theory and uses it to assess the suitability of DDLs for automatic data integration using three basic constituents: Variety, Tension, and Entropy.

Research Questions

This research proposes to answer three questions:

1. What are the theoretical necessary requirements of a DDL built to fully support automatic data integration from autonomous heterogeneous data sources?
2. Has there been real advancement in DDL design towards such integration? (i.e., do new DDLs progressively meet the theoretical requirements?)
3. Is there a better way to approach the design of DDL that fully support such integration?

Contributions

This research proposes a data integration theory. The theory draws on multiple scientific fields, especially on complex adaptive systems (CAS) theory. The study of CAS holds that the dynamics of complex systems are founded on universal principles that may be used to describe disparate problems.

The proposed data integration theory is tested against thirteen DDLs representing different computing eras from the 1960's to date. It guides the approach to building an entirely new type of DDL that better serves automatic data integration. The new DDL is tested using the same method applied on the sample of DDLs. The theory and the test results are used to set a direction for future research.

Data Integration Theory

The theoretical necessary requirements for a DDL to fully support automatic data integration from autonomous heterogeneous data sources are:

1. Availability of a regulator that abides by the law of requisite variety (LRV).
2. The ability to create and sustain tension between (partially) mapped data structures
3. Behave like a noiseless communications channel (e.g., having entropy=1)

LRV comes from the field of cybernetics, a predecessor of CAS. To understand LRV it is necessary to understand, define and measure variety. Abstractly, variety of a given system is the number of meaningfully different states and disturbances the system has. The research harnesses Zipf's groundbreaking work to measure variety in data structures via the counting and graphing of data structure signifiers (e.g., fields, tags) usage distribution and meaning distribution.

Tension can be intuitively understood as a mechanical force that is developed when an anchored string keeps a suspended object from falling. The cables on the Golden Gate Bridge serve as a long lasting example. The Tacoma Narrows Bridge collapsed days after it opened to the public. Its cables exemplify short lived tension. In a non-mechanical reality tension is created when a mapping exists between two objects. A mapping between elements or groups in data structures connects those objects in a manner that preserves their meaning. Meaning is often in the eyes of the beholder. Per CAS, such mapping creates mental tension (e.g., cognitive connection). Otherwise the mapping is senseless and does not serve its purpose. Mapping can be precisely defined, classified, and measured mathematically.

Entropy: Mechanical mapping requires a mechanical conduit that serves as a connection channel. Higher level systems require higher level (more abstract) channels. The fewer obstacles a channel has, the better it functions. Shannon's Theory of Communications (or Information Theory) defines and measures some characteristics of connection channels, termed as communication channels. A channel that has no obstructions at all, that is, it is noise free, is said to have entropy that equals to one. The

noisier the channel, the less entropy it has, and the less efficient it is in channeling information (or energy).

For a mapping to occur it needs a channel through which the mapping's energy is conducted. For the energy to connect meaningfully (rather than be wasted), it needs to map to an object with identical meaning. The two objects could be represented differently, but they need to mean the same. This calls for a mechanism that can exercise selection criteria against which the variety of possible mappings from a system to its environment may be sifted into those variations in the system that more closely map the environment and those that do not. The sifting mechanism is termed "regulator" in cybernetics. Per LRV, it needs to be able to regulate (to channel) at least as much variety as the regulated system is able to handle.

The proposed data integration theory predicts that only DDLs designed to meet its three principles have the potential to support automatic data integration. The same principles apply to physical systems too. Case in point, the Tacoma Narrows Bridge did not meet the three criteria. Specifically, its regulation mechanism was unable to mediate some of the environment's variety (wind) and therefore it collapsed. The Golden Gate Bridge better meets the three criteria, resulting in its cables sustaining the mappings they were designed for. The same principles apply to more abstract systems, such as data structures.

Measuring Variety, Tension, Entropy

To measure variety we use Zipf Distribution of Words and Zipf Distribution of Meanings. Each one is a Ratio variable. The measure is carried out by counting words, counting meanings, plotting the results, and calculating a correlation coefficient for the

plot. Additionally, we qualify the variety using classifications of ambiguities and assigning those to data samples.

Tension is manifested by the existence meaning preserving isomorphism mappings between a data structure and other data structures in its environment that could be used to satisfy integration goals.

Entropy is a ratio variable calculated according to the formula proposed by Shannon's information theory.

Testing the Theory

If there is a DDL that satisfies all the theory's requirements yet doesn't support automatic data integration, the theory must be rejected. In the absence of such a DDL, it becomes necessary to build a new DDL that satisfies all the theory's requirements and evaluate its fitness for automatic data integration. It is desirable to describe the fitness using quantitative measures.

DDLs Sample Selection

The study uses 13 DDLs representing several computing generations – from the 1960's to date. Standards are represented by SWIFT and EDI. Structured DDLs are represented by COBOL FD sections, and by ADABAS data structures. Semi-structured DDLs are represented by numerous XML, DTD and XSD structures. Ontologies are represented by RDF and OWL structures. Gathering DDL from different vertical markets and written in two unrelated natural languages (English and Hebrew) minimize the risk of bias due to a specific market, a specific natural language, or the cultural background of data structure creators. Vertical markets represented in the data sample are Banking, Real Estate and the Travel industry.

Test Results (Partial list)

All correlation coefficients of Zipf distributions of words are almost identical, around 0.92. The Zipf distribution of meanings is also indistinguishable among DDLs, hovering around 0.97. The exceptions are EDI and SWIFT, whose design ensures one signifier per "entity" or "object" and one meaning per signifier.

No DDL in the sample has a tension creation mechanism. No DDL in the sample supports invertibility, vocabulary preservation, or structure preservation.

EDI and SWIFT have entropy equals one. All other DDLs have entropy slightly lower than one. Entropy of one is a desirable result, for several reasons: first, standards (e.g., EDI, SWIFT) have such entropy, and standards work well. Entropy of one is the most efficient communication channel, also a desirable characteristic as it implies no redundancy. Analysis of entropy over computing generations reveals that there is no significant difference in the entropy of DDLs.

Discussion and Implications

The research shows that existing data integration approaches to date do not implement a robust regulation mechanism that satisfies the law of requisite variety. Existing data integration approaches do not yield tension unless humans intervene in the mapping process and invest mental energy to keep the relations from falling apart when a data source changes its data structure. Such failures are due to the absence of a regulator that can successfully overcome the existing semantic heterogeneity, which in turn is a manifestation of the theoretically infinite variety that exists in the environment. Having identified common weaknesses in all DDLs, a DDL design approach that is consistent

with CAS requirements to support automatic data integration is proposed. The DDL is termed GlossoMote[1]. It is a mathematically sound solution for maximizing the effectiveness and efficiency of DDL as it relates to automatic integration. The GlossoMote provides a regulator that can handle all possible variety that can be created with this DDL; it provides for build-in homomorphism mappings, therefore producing the necessary tension to preserve meaning. Its Entropy equals to one, which means it is very efficient and does not require (nor allows for) redundancy. Additionally, the possibility of adding collaborative tagging (Folksonomies) to data structures for the purpose of improving their potential to create tension (homomorphism or isomorphism mappings) is discussed.

Future Research

GlossoMote assumes the existence of a small set of axiomatic facts, similar to Mendeleev's periodic table. Further research is required to develop an axiom set that addresses the complex environment we live in. What constitutes a "good" or an "ideal" quasi-periodic table, and what are its limitations remain open questions. The process of creating new DDLs or improving existing ones is time consuming and uses expensive resources. Thus it would be useful to build a mathematically sound DDL development framework against which any DDL design can be evaluated and tested before it is implemented even in the laboratory. It has been achieved with Codd's relational model (Codd 1970). Additional research is needed for our proposed evaluation framework to reach a similar level of practical simplicity.

---

[1] Glosso means "of the tongue", Mote means "a small particle" according to the Babylon.com dictionary.

CHAPTER 1

INTRODUCTION

Chapter one specifies the research problem, exposes opening assertions, and then makes non-trivial connections between some aspects of complex adaptive systems theory and computing constructs, both central to the entire research.

Chapters two through six provide in-depth background information, leading to the chapter dedicated to the research methodology, followed by its implementation. Results are then presented, followed by rigorous analysis and discussion of the findings. Conclusions are drawn in the last chapter, which ends with recommendations and sets the stage for future research.

1.1 Problem Statement

Designers and advocates of contemporary DDLs claim that recently proposed DDLs are better designed for, or entirely solve the challenge of, automatic data integration from heterogeneous sources. The questions this research addresses are:

1. What are the theoretical necessary requirements of a DDL built to fully support automatic data integration from autonomous heterogeneous data sources?
2. Has there been real advancement in DDL design towards such integration? (e.g., do new DDLs progressively meet the theoretical requirements?)
3. Is there a better way to approach the design of DDLs that fully support such integration?

This research asserts that data integration is not merely a technical challenge. Rather, it is a socio-technical phenomenon, a manifestation of morphogenic processes of which technology plays only a part. A quantitative assessment method based on Complex Adaptive Systems (CAS) theory is developed to assess the difference between numerous DDLs and thus determine if real progress in DDL development was made, and suggest improvements if the study concludes they are necessary for realizing automatic data integration.

1.2 A Brief Socio-Technical Background

This section explains the importance of data integration to organizations. It then addresses how data is organized technically, using data structures expressed in a variety of computing languages.

Mergers, acquisitions, and cooperation among autonomous organizations are morphogenic processes as defined and explained in CAS theory (Buckley 1967; Burrell and Morgan 1979), a topic expounded on later in this research. The need for data integration within the enterprise and across organizations increases as the enterprise grows. "*Integration at the [computerized business] systems level requires common standards and data definitions, and some means of synchronizing the communication between different software applications*" (Stohr and Nickerson 2003). Economic incentives are significant: imperfect interoperability costs the U.S. automotive supply chain at least $1 billion per year (Linden, Buytendijk et al. 2003); during the 1990's the US banking industry experienced heavy mergers and acquisitions. In the process banks expected to lose 10 percent to 15 percent of their customers due to integration problems. When First Union Bank merged with Wachovia in 2001, "*it was still reeling from the*

disastrous CoreStates merger. And Wachovia had recently fumbled the acquisition of two Virginia banks by attempting to convert both over the same weekend"(Dragoon 2004); emergency response management information systems must integrate data from heterogeneous and autonomous resources (Turoff, Chumer et al. 2004) because "*response to an emergency more often than not involves several organizations that under normal operations are loosely connected or entirely unrelated*" (Rohn 2006).

A data definition language specifies how to organize and interconnect data into a useable structure. DDLs are used to codify messages to be sent or received by computerized systems or their components. Hundreds of DDLs have been developed over the years. Examples of DDL include Cobol's structured File Description (FD) section; delimited flat files such as Comma Separated Values (CSV) and Data Interchange Format (DIF) for data exchange; Structured Query Language (SQL) for relational databases; Extensible Markup Language (XML) for semi-structured data; and, metadata and ontologies expressed in a variety of DDLs such as Resource Description Framework (RDF) and Web Ontology Language (OWL).

Computerized socio-technical systems require data integration to serve human needs. The goal of data integration is to synthesize data from different sources into a unified view termed as global schema. Integration of data can be carried out only if the corresponding data structures expressed in any DDL are first mapped to each other. Achieving automatic integration—a much more difficult problem—of heterogeneous data structures provided by autonomous sources has been the goal of academics, practitioners and industry for many years. It has not been achieved to date, despite enormous

investment of talent and resources in the quest for a working solution. It is the question of "why" and the search for a solution that motivates this research.

The computing industry's initial paradigm was centralistic and monolithic. That is, organizations that computerized some of their business processes, such as accounting or manufacturing, had a single computer that executed programs from a central location. Organizations modified their interaction methods by exchanging data electronically among their computers. Typically, the structure of the data to be exchanged was agreed upon by the businesses, and then data was exchanged by physically exchanging computer tapes (Rohn 1982). Electronic Data Interchange (EDI) pioneered non-physical exchange of data through Value Added Networks (VAN) that acted as switchboards (Unitt and Jones 1999).

When computers and networks became pervasive and thus less expensive, the paradigm changed to that of networked computing. Ubiquitous exchange of data directly between business units and organizations became the norm. The dependency upon humans mapping data structures that facilitates data integration has become a costly bottleneck. For example, imperfect interoperability costs the U.S. automotive supply chain at least $1 billion per year (Rohn and Klashner 2004). Automation of the data integration process became an acute issue for businesses and other organizations. The lack of real progress in automatic integration over thirty years of research efforts by academia and industry indicates that there might be an invisible "brick wall" that is a sociotechnical phenomenon. At the same time other areas of computing continued their rapid growth. The growth of the entire computing industry—a holistic system—is hampered by the lack of progress in data integration research.

Therefore, a systems research perspective is needed to categorize phenomena associated with the data integration problem. Using generic systems research approach, computerized systems are defined as ensembles or sets of components with a distinct boundary that evolves as the systems interact with each other via input, output and feedback. Many computerized systems have sub-systems, which themselves have identifiable boundaries, and the sub-systems are mutually interdependent. For example, an accounting system may have sub-systems such as General Ledger, Accounts Receivables and Accounts Payable. Computerized systems interact with each other by importing and exporting data, which they convert to useable information. These systems are influenced by the environment they operate in. For example, regulations such Public Law 104-191, which is better known as the Health Insurance Portability and Accountability Act (HIPAA) of 1996, and Public Law 107–204 (the Sarbanes Oxley act of 2002) caused the modification of computerized systems and the organizations using them. These modifications are a form of adaptation.

Data integration is a social communications system (for business, leisure etc.) implemented through computerized technology, allowing for the formalization of specialized language. Data integration is also goal oriented. The DDL provides syntax for communicating actions in and between organizations. Data structures (built using DDLs) can communicate illocutionary acts such as informing, ordering, warning, or undertaking and other speech acts of change (Searle 1969). Locutionary acts[2] (Searle 1969) are fundamental in data integration, as they facilitate informing other users or systems. If a

---

[2] Locutionary speech act is the act of communicating something.

recipient system "understands" the message and takes action (or avoids taking action) as a result, then the data integration performed a perlocutionary act[3] (Searle 1969).

Lack of progress in automatic integration of heterogeneous data sources over thirty years span suggests a reverse salient could exist in the underlying DDL. A reverse salient exists as a result of unforeseeable confluences, and will go undetected "*unless inventors, engineers and others view the technology as goal-seeking*" (Hughes 1983 page 80). Similarly to Hughes, this research does not take a technological deterministic stance, but rather a historical presentation of sociotechnical phenomena. Goal-seeking behavior has become more interesting to the information systems and software engineering research communities as purely technological solutions have failed in numerous venues such as requirements engineering. As defined by the sociologist Buckley, goal-seeking behavior is an attribute of CAS which he introduced as a synthesis of numerous theoretical perspectives (Buckley 1967).

Historically, technological progress is often the result of an attempt to respond to a reverse salient, when one has been detected. Countless inventions and technological progress have resulted from efforts to correct reverse salients (Hughes 1983). It is important to note that although this study looks at some historical data, it is not intended to forecast the future of automatic data integration. A forecast is a statement, usually in probabilistic terms, about the future state or properties of a system based on a known past and present. A conditional forecast states in probabilistic terms what the future will be if a course of action is taken. A forecast that states with a high degree of confidence what

---

[3] A perlocutionary act is any speech act that amounts to getting someone to do or realize something

the future will be is referred to as a prediction. This research is not concerned with forecasting or with predictions.

Data structures are ensembles in computerized systems that interact with their environment and have temporal, spatial and causal relationships with the environment and with components within the systems. Data structures get their shape from the data definition language used to materialize the structure. Our proposed framework maps DDL characteristics to CAS via measures drawn from Linguistics, Artificial Intelligence and Information Theory. Data structures are engineered using natural language which implies variety, a CAS characteristic. We measure the existence of variety by gauging and classifying inter-DDL and intra-DDL natural language ambiguities (Gardent and Webber 2001); for the same reason we use distribution of words and distribution of meanings (Zipf 1949), two measurements of natural language ambiguity. Tension is what CAS uses to maintain acquired variety. To measure if tension exists we use meaning preservation (Sowa 1999). CAS, as a morphogenic system, seeks to reduce its local entropy and increase order. The framework uses entropy (Shannon 1948) as a direct measure of the level of order achieved by utilizing a given DDL. Figure 1 has a pictorial summary of CAS characteristics and the disciplines used to measure them. This new treatment of DDLs in the IS research community may necessitate revised research agendas based on altered assumptions.

1.3 Chapter 1 Summary and Implications to the Research

Mergers, acquisitions, and cooperation among autonomous organizations are morphogenic processes as defined and explained in CAS. Computers put structure to organizational data via the usage of DDLs. When organizations cooperate or merge, at

least part of their data needs to be integrated, in order to serve human needs. Integration of data can be carried out only if the corresponding data structures expressed in any data definition language are first mapped to each other. The mapping must not break over time. The lack of real progress in automatic integration over thirty years of research efforts by academia and industry indicates the existence of a reverse salient, a socio-technical phenomenon.

Implication of chapter to study	How Implication will be used
Data Definition Languages (DDL)	Focuses the research on computing constructs used to organize data serving human goals
Complex Adaptive Systems Theory	The theoretical prism through which DDL and their suitability for automatic data integration are viewed, measured, and analyzed
Variety – a key concept in CAS	Quantify types and magnitude of variety in DDL, as these pose a difficulty in automatic integration
Mapping – a key concept in CAS	Assess mathematically the possibility for the existence of correct mapping between data structures in order to operate as intended by humans
Tension – a key concept in CAS	Assess the existence and strength of the force that maintains the mapping between acquired variety present in two or more data structures expressed in any DDL
Entropy – a key concept in CAS	Calculate the entropy of DDLs examined in the study, as a direct measure of organization (clarity) in data structures expressed in any DDL

Table 1: Summary of Chapter 1 Implications to the Study

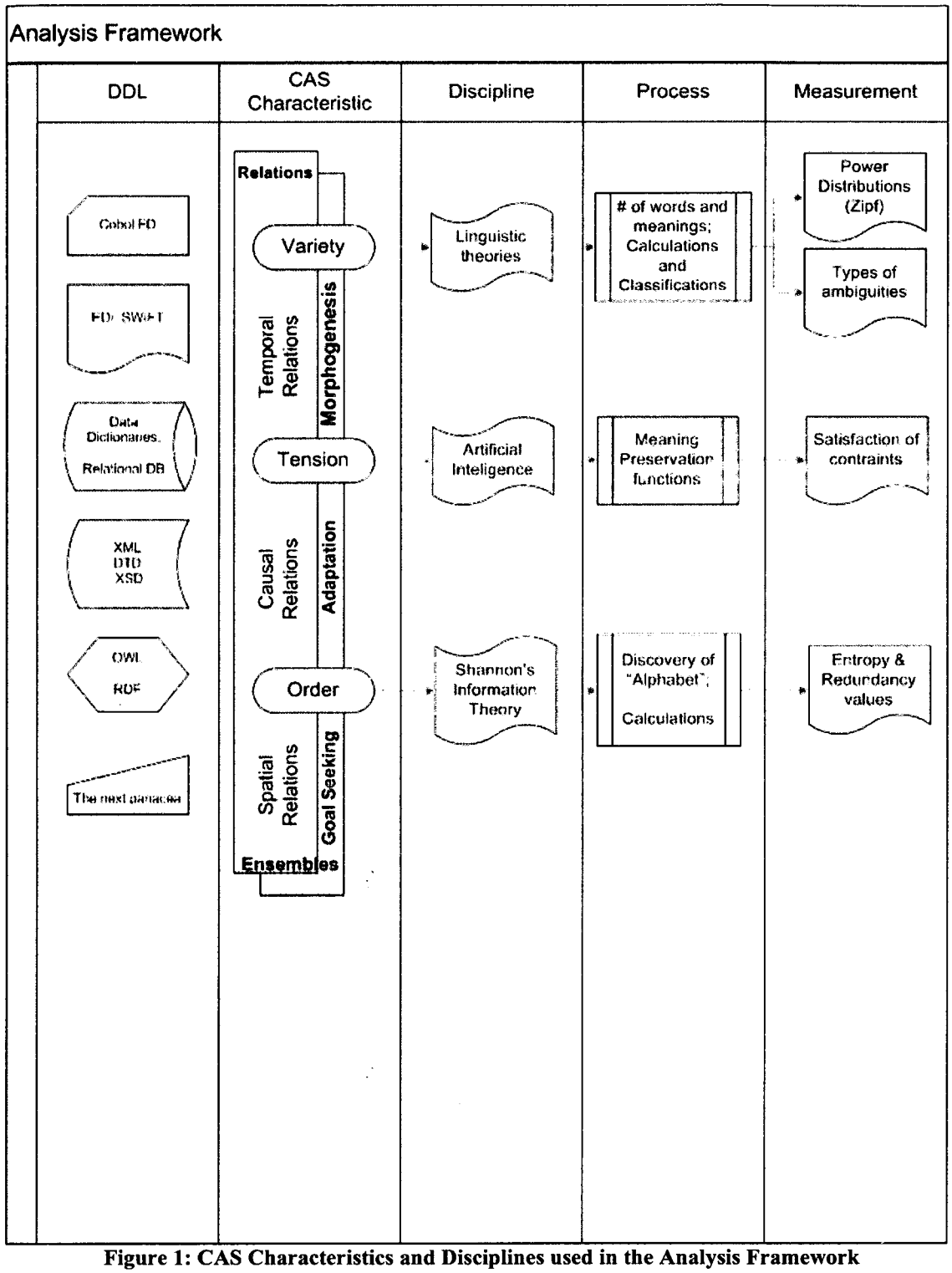


Figure 1: CAS Characteristics and Disciplines used in the Analysis Framework

CHAPTER 2

DATA DEFINITION LANGUGAGES

2.1 Introduction

Chapter 2 provides a temporal overview of the development of DDLs. It begins with motivation for having DDLs. Then it reviews the progression from structured DDLs such as COBOL that do no convey semantics to intermediate solutions such as EDI where meaning is pre-agreed. It describes progression towards database DDLs and continues with the introduction of semi-structured DDLs such as XML, their failure to convey semantics, and concludes with a variety of proposed solutions to convey meaning.

2.2 The Need for Data Structures

Data is processed for a reason, regardless of the type of data. All computerized data has some structure regardless whether the data pertains to business applications, real time applications, gaming, or any other domain, Data needs to be modeled and defined in a structured manner in order to accomplish further processing. Data structures are formed using DDLs. This research defines DDLs broadly: any syntax that can be used to create a computerized data structure is a data definition language. DDLs specify how to organize and interconnect data into a useable structure. DDLs are used to codify messages to be sent or received by computerized systems and their components. These are just a few of the reasons why a DDL is a fundamental computing construct. Data sources are "structured," (e.g., Cobol, database) "semi-structured," (e.g., web pages, Word documents) and "unstructured," (e.g., images, voice). A detailed discussion of structured and semi-structured DDLs appears in subsequent sections.

Information systems engineering activities should and usually do include the creation of an organized collection of information about the definition, structure, and use of data in an organization. This collection, often referred to as data dictionary, is an organizational asset that is carefully maintained by authorized personnel only. DDL materialize definitions from a given data dictionary (Date 1990). Multiple implementations exist to define data. This chapter covers two main families of DDLs: structured and semi-structured. Structured DDLs are usually more rigidly structured and are part of a program's source code; data and structure are stored separately from the data they describe. In contrast, early semi-structured DDLs did not include declaration of data types (e.g., alphanumeric, float, integer), a feature later enhanced with optional data typing. Semi-structured DDLs combine the data structure and the data itself, creating verbose files that can be traversed ("queried") directly, even when the structure is not known a priory.

2.3 Structured Data Definition Models

There are some structured DDLs that have been widely used, three of which are closely examined and analyzed in this research:

- Cobol's (COmmon Business Oriented Language) DDL implementation
- Data Interchange Format File (DIFF)
- Relational database DDL - part of the Structured Query Language (SQL)

2.3.1 COBOL Data Definition

The COBOL programming language includes formal syntax to define every group and data element it uses. Every COBOL program consists of four separate divisions, each

with a specific function. The DATA DIVISION describes the input and output formats used by the program. There is a FILE SECTION within the DATA DIVISION that provides a detailed description of files used for input and output (IBM 2000) as illustrated in the code fragment below:

```
DATA DIVISION.
FILE SECTION.
FD MyFile  Label records standard,
           value of file-id is FMyFile.
01  MyRec.
    02 PrimKey    pic x(45).
    02 MiscStuff  pic x(256).

FD PO Label record standard.
01 PO-Rec        pic x(126).
```

Figure 2: Cobol's Data Division code fragment

File structures defined inside a COBOL program are not exposed outside the program. There exists a technical option that allows the file definition to reside in a shared library, and copied from there into specific COBOL programs. The library serves as a reference standard. Therefore, the library is modified with great care by authorized personnel who have complete control over the library and its content to the last byte (Rohn 1983). In addition, COBOL has a list of reserved words that cannot be used for describing files or fields within files.

2.3.2 Data Interchange Format File (DIFF).

There existed in the 1970s a software vendor, named Software Art, that developed a mechanism to export and import data into its spreadsheet software named VisiCalc. The company named the file used for the mechanism Data Interchange Format File, or DIFF. It became a de-facto protocol for exchanging data among early PC applications, such as

VisiCalc (Brickley 1979), Lotus 1-2-3 (Bricklin, Kapor et al. 2003), and others. The DIFF protocol is materialized as a sequential file wherein fields are separated by a delimiter such as a comma, a tab, or by a special character. Contemporary typical DIFF file extensions are DIF and Comma Separated Values (CVS). The initial series of characters extracted from a digital file up to a pre-specified demarcation character(s) is commonly abstracted as a row structure. The first row (typically referred to as a record in the programmers' community) in the file is optionally filled with field names. DIFF do not support typing (e.g., integer, float, character). The programmer needed preexisting knowledge of the meaning and significance of the fields in the file in order to use a DIFF or CVS. Programmers customarily communicated the meaning and significance of these fields orally or in some form of documentation.

```
Listing No.,Address,House Type,Bedrooms,Bathrooms
1001,"1 Penn Plaza, Newark, NJ",Skycraper,N/A,N/A
1002,"100 Main St, Edison, NJ",Victorian,5,2.5
1003,"244 Crowells Rd, Red Bank, NJ",Historic,3,1.5
```

Figure 3: DIFF / CSV Sample

Case Study of DIFF Usage. Bank HaPoalim commissioned the development of a PC based application written in dBase in 1985. That dBase application was intended to aid the bank's sales staff in the collection, analysis and distribution of data pertaining to potential clients. The system became operational in the fall of 1985 and was used for about four years. Typical usage scenarios for Bank HaPoalim was the exchange of data between a mainframe application and other dBase applications at which time DIFF files were either created or read by the dBase Potential Client application. Programmers had to

negotiate ahead of time the format and meaning of each DIFF file. Many times the negotiation and agreement were done orally with only fragmented documentation available inside the dBase code itself (Rohn 1985).

2.3.3 Relational Database Data Definition Language

The Association of Computing Machinery (ACM) published a seminal paper by Codd (Codd 1970) on "*Relational Model of Data for Large Shared Data Banks*". Codd's approach became widely accepted as the definitive model for relational database management systems (RDBMS). During the 1970s, a group at IBM's San Jose research center developed "System R", a database system premised upon Codd's model (Blasgen, Astrahan et al. 1981). The data in System R was stored, manipulated, or retrieved by a computing language called Structured English Query Language ("SEQUEL"). The acronym SEQUEL was later condensed to SQL due to a trademark dispute.

The American National Standards Institute (ANSI) adopted SQL as a standard in 1986. The International Organization for Standardization (ISO) adopted SQL as a standard in 1987. SQL has a list of reserved words, clustered by function, such as:

- Data Retrieval, (e.g., Select From Where Group By)
- Data Manipulation (e.g., Insert, Update, Delete, Merge)
- Data Control (e.g., Grant, Revoke)

SQL has a set of words to define data, which comprise SQL's DDL. These words are: CREATE, ALTER, DROP.

```
CREATE TABLE table_name
(
column_name1 data_type,
column_name2 data_type,
.......
)
```

Figure 4: SQL code fragment

The CREATE command causes an object (a table, for example) to be created within the database. ALTER permits the user to modify an existing object in various ways; e.g., adding a column to an existing table. DROP causes an existing object within the database to be deleted.

2.4 Standards-Based Data Definition Languages

2.4.1 Electronic Data Interchange (EDI)

Electronic Data Interchange (EDI) established a common language for exchanging business related transactions via the creation and enforcement of a standard. EDI began in the United States around 1968, when two organizations started exchanging point-to-point information between their computers through private telecommunication networks. The first transactions to be transmitted were invoices and bills. Organizations transmitted these documents electronically through communication lines instead of writing and mailing them. Since the transactions did not transit through a physical mailbox, they were more rapid and the organizations saved both time and money. Prospective partners had to first agree upon the procedures of transmission and on the internal standards to be used when sending and receiving data.

Exchanging data was relatively easy when only two partners were involved but became more complicated when many partners using different computer systems, protocols and interfaces decided to exchange data through EDI (Emmelhainz 1990). By 1975, many vertical business groups interested in EDI were actively working on defining their own domain specific EDI standards. Such standards emerged in several market segments, such as:

- transportation industry represented by a consortium named Transportation Data Coordinating Companies (TDCC)
- food preparation and sales industry voluntarily governed by the Food Marketing Institute (FMI)
- government agencies providing Medicare benefits
- the International Association of Industrial Accident Boards and Commissions (IAIABC).

An EDI implementation requires skilled personnel to use specialized software to map data from the organization's native format to EDI and vice-versa. Once the mapping is complete and verified, relevant flow of information between trading partners needs no human intervention unless an error occurs in the process. EDI is a vibrant and still growing approach to data exchange and integration among autonomous and heterogeneous systems. However, all EDI implementations rely upon industry consensus regarding their voluntary governance through standards, wherein ad hoc or negotiated agreements between parties are exceptionally rare.

Case Studies of EDI Usage. Medicare is a federal health insurance program for people age 65 and older and for individuals with disabilities. During 2001, Medicare processed

157,306,245 electronic media claims using EDI, accounting for over 97% of all claims processed by Medicare. The number of EDI claims rose to 170,558,776 in calendar year 2004 (Medicare 2005). The IAIABC, an association of government agencies that administer and regulate their jurisdiction's workers' compensation acts creates, maintains and publishes EDI standards that are specific for workers' compensation insurance claims. "With over 300 members, the IAIABC represents a diverse group of workers' compensation professionals, medical providers, insurers, and corporate agencies" (IAIABC 2005). The latest IAIABC EDI standard, Release 3, was made available in 2004. In the State of Minnesota alone, there are approximately thirty (30) state jurisdictions that currently participate in or are planning to use EDI communications with their trading partners using the various IAIABC release standards (MDLI 2005).

2.4.2 Society for Worldwide Interbank Financial Telecommunication

The Society for Worldwide Interbank Financial Telecommunication (SWIFT) created a key "common language" for exchanging financial related transactions via the creation and enforcement of standards. The SWIFT runs a worldwide network by which messages concerning financial transactions, such as payments, letters of credit, securities transactions, foreign exchange, and others, are exchanged among financial institutions (Walmsley 1992). In the year 2000, the SWIFT carried about 1,200,000,000 messages. As of December 2001, the SWIFT linked over 7,000 financial institutions in 194 countries and carried payment messages averaging more than six trillion US dollars per day. On 30 June 2005, there were 11,487,827 messages processed, a new daily peak for the SWIFT. As of June 2005, there were 7,668 active SWIFT participating institutions in 203 countries (SWIFT 2005). The SWIFT website reads:

> *"Standards are an essential element of SWIFT's global offering. We are committed to the collaboration of efforts and convergence of standards, so that our community can benefit from potential cost savings, eliminate redundancies and smoothly expand into previously untapped markets."* (SWIFT 2005)

The SWIFT attempts to satisfy the needs of vertical markets. For example, in October 2001, the SWIFT announced plans to migrate its securities industry standards from ISO 7775 to ISO 15022 XML standard. The SWIFT planned that by year-end 2004 the majority of securities transaction messages, for the front and back office, would use ISO 15022 XML.

The SWIFT was appointed by the ISO to serve as the sole registration authority for ISO 15022 XML. In this role, the SWIFT maintains a Data Field Dictionary (DFD), and a Catalogue of Compliant Messages. The society has been given the mandate by its members and the ISO to create new fields as necessary. The SWIFT also enforces compliance of messages built directly by communities of users, who develop messages based on needs. Thus, the SWIFT is a vibrant and still growing organization. It determines the solutions to financial data exchange and integration among autonomous and heterogeneous systems worldwide. Implementation of the SWIFT standards relies on industry consensus based standards, where ad hoc and negotiated agreements between specific parties are unheard of.

SWIFT messages consist of five blocks of data including three headers, message content, and a trailer. Message types are crucial to identifying content. The blocks are identified by position; therefore they are not treated as independent and distinguishable data elements. SWIFT terminology is given from the perspective of SWIFT and not the user, with no exception. All SWIFT messages include the literal "MT" (Message Type).

This is followed by a 3-digit number that denotes the message type, category, and group. For example, "MT502" is an order to buy or sell a financial instrument via a third party. The first digit (5) represents the category. A category denotes messages that relate to particular financial instruments or services such as Precious Metals, Syndications, or Travelers Checks. The category denoted by 5 is "Securities Markets". The second digit represents a group of related parts in a transaction life cycle. The group indicated by "0" is a Financial Institution Transfer. The third digit ("2" in MT502) denotes the specific message type. "2" means "Third-Party Transfer".

Each SWIFT message is assigned a unique identifier. A 4-digit session number is assigned each time a user logs in. Each message is then assigned a 6-digit sequence number. These are then combined to form an Input Sequence Number (ISN) from the user's computer to SWIFT, or an Output Sequence Number (OSN) from SWIFT to the user's computer. The SWIFT Header Block is fixed-length and continuous with no field delimiters. Its format is as follows:

{1:	F	01	BANKBEBB	2222	123456}
(a)	(b)	(c)	(d)	(e)	(f)

Figure 5: SWIFT Header Sample

(a) 1: = Block ID (always 1)

(b) Application ID as follows: F = Financial Application, A = General Purpose Application, L Login

(c) Service ID as follows: 01 = FIN/GPA 21 = ACK/NAK

(d) BANKBEBB = Logical terminal (LT) address. It is fixed at 12 characters; it must not have X in position 9

(e) Session number. It is generated by the user's computer and is padded with zeros.

(f) Sequence number that is generated by the user's computer. It is padded with zeros.

SWIFT continues to grow and change as it adapts to its changing environment. For example, the European Commission's Markets in Financial Instruments Directive (MiFID) become law in November 2007 and has had a profound impact on the securities industry. For example, by mid 2007 SWIFT had fully MiFID-compliant message standards meeting the requirements laid out in the Directive (SWIFT 2007).

2.5 Semi-Structured Data Definition Languages

2.5.1 Introduction to Semi-Structured data

There are two extreme types of data organization. One is completely organized; the other is not organized at all. In between there are data that have some structure in them. In between the two extremes of structured and unstructured data files there exists semi-structured data. For example, this document is a case in point of a semi-structured data source. It has some structure, such as headings and paragraphs. However, headings may appear in any place the author sees fit; a paragraph could be of any length; it does not need to adhere to a schema to be processed, although it can be done voluntarily. Abiteboul loosely defines semi-structured data as: "...data that is (from a particular viewpoint) neither raw data nor strictly typed, i.e., not table oriented as in a relational model or sorted graph as in object databases" (Abiteboul 1997). Suciu gives an overview of semi-structured data. He writes:

> *"Research on semi-structured data started from the observation that much of today's electronic data does not conform to traditional relational or object oriented data models. Several applications store their data in non-standard data formats: legacy systems, structured documents like HTML*

> *or SGML etc. Another instance is the integration of heterogeneous data sources: often these sources belong to external organizations, or partners, not under the application's control, and their structure is only partially known, and may change without notice."* (Suciu 1998)

Semi-structured data has partially known pattern that is subject to change without notice. It may have missing attributes, multiple occurrences of a single attribute, or multiple attributes. Identical attributes may have different types such as integer or string. Related or similar data may be represented in different ways, and unrelated data may be represented in similar ways. Examples for semi-structured data are cooking recipes and genealogical records.

For instance, genealogical information on the web site named Ancestry.com (MyFamily.com 2006) has a different structure than the one supported by the Israeli Beith Hatfutzot (Diaspora House) organization. Beith Hatfutzot provides Hebrew names independently of secular ("European") names for the same person. In many genealogical cases, only the Hebrew name is available in Beith Hatfutzot files. In contrast, the concept of a Hebrew name does not exist in the Ancestry.com schema. Access to the semi-structured source data and structure might be limited, as in the case of Ancestry.com that imposes access fees. It is possible that detecting, removing or correcting imprecise or erroneous data might be required when one wants to merge data from these two independent sources. Several additional examples for data incompleteness or mismatch are provided in (Rohn and Klashner 2004).

The proliferation of networked information systems and specifically the Internet created a need for a universal data transfer language. Its function is to enable the merging of document-centric Web pages with a data-driven infrastructure. Consequently there

exists a variety of machine-processing enabled semi-structured DDLs. The concept of self-describing semi-structured data took center stage, giving rise to the eXtensible Markup Language (XML). This can be attributed to XML's fit to self-describing data representation. The following sections review major types of approaches for self-describing data representation languages. Some were specifically developed for advancing the Semantic Web, where "information is given well-defined meaning, better enabling computers and people to work in cooperation" (Berners-Lee, Hendler et al. 2001).

2.5.2 Object Exchange Model (OEM)

The Object Exchange Model (OEM) is one of the first and simplest information models that have been proposed for exchanging information on the Web by the database community, before XML took center stage (Papakonstantinou, Garcia-Molina et al. 1995). The main features OEM offers are object identity and nesting. The OEM model is a directed labeled graph (see Figure 6) i.e., a set of objects called vertices joined by arrows, in which every object has a distinct identity and a type. Such graphs are used for computerized representation of data structures. Apart of atomic types like integers and strings, OEM supports sets and lists i.e., similar to what one finds in graphical user interface (GUI) drop-down controls. OEM object graphs can be represented in a graphical notation and serialized (i.e., converted to a stream) using a simple text-based syntax. The Object Exchange Model (OEM) serves as the basic data model in Tsimmis and Lorel, two data integration projects that we will discuss later in this work.

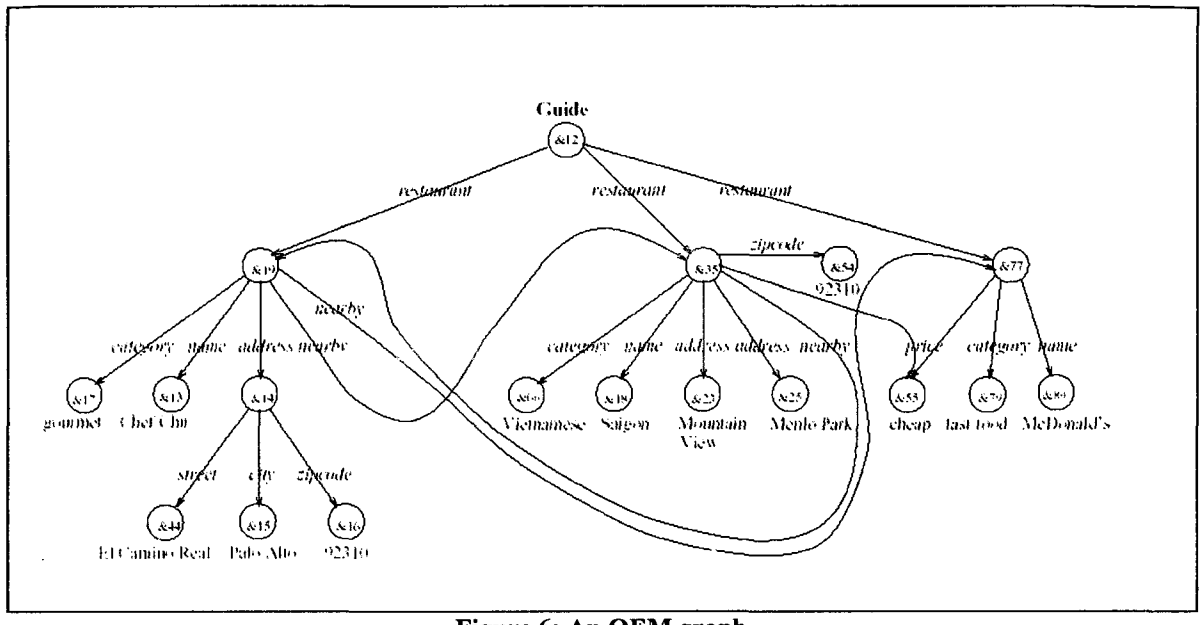


Figure 6: An OEM graph

2.5.3 Extensible Markup Language (XML)

Advancement in publishing from handwriting to mechanized printing created a need for a markup language. Authors and editors needed to communicate with typesetters about how to present the written material. Editors had to specify the representation of text and images, such as location, usage of bold letters, large letters, font size, tabs, paragraphs settings, and other considerations. No national or international standard was developed for such markups.

Similarly, many independent markup notations were developed as word processors replaced typesetters and typing machines. Word processor manufacturers had each their own proprietary markup language. Initially a computerized word processor was character based ("green screen") and the author had to insert special characters or combinations of characters to indicate to the text processor what style to use and where. The word processor was able to "translate" the special characters to printing commands, provided a

sophisticated printer was available. So, while on the screen the text looked like monotype embedded with strange markings, the printed result looked reasonably well formed. Using character based interface, there was no way for the author to see on the screen what would the printed version look like. When the Graphical User Interface (GUI) became ubiquitous, the monotype interface was substituted with "What You See is What You Get" or WYSIWYG, hiding the formatting markup from the user.

Competing word processor vendors had incompatible markup languages. Hence, authors were not able to exchange data and format with each other, unless they used the same version of a word processor, or used some markup conversion software.

Many years before GUI WYSIWYG three IBM researchers, Charles Goldfarb, Ed Mosher, and Ray Lorie, began working in the late 1960's on documents portability problems. They focused on legal documents created on disparate systems using proprietary formats. Their research brought to light three primary requirements for document portability: (a) There must be support for common document format (b) Document format is domain specific (e.g., legal, chemical, financial) (c) Document format has to follow specific rules. "This analysis of the markup process suggests that it should be possible to design a generalized markup language so that markup would be useful for more than one application or computer system." (Goldfarb 1973)

A new framework was created and named Generalized Markup Language (GML--also the initials of the three inventors). Eight more years of research and work with technical groups on GML yielded the Standard Generalized Markup Language (SGML) framework. SGML is based on the concept of document being composed of a series of parts, containing one or more logical elements. SGML clearly identifies the boundaries of

every part of a document. It requires marking up where the various elements of a text entity start and end, eliminating possible guesswork. SGML was adopted and approved by the International Standards Organization (ISO) in 1986 and is now overseen by ISO's JTC1/SC34 subcommittee. SGML is the "Grandfather" of all markup languages, including HTML and XML (Duschka and Genesereth 1977). The eXtensible Markup Language (XML) was developed by an XML Working Group in 1996. XML is a subset of SGML, having a fixed set of SGML features. XML's goal is "to enable generic SGML to be served, received, and processed on the Web". XML has been designed for ease of implementation and for interoperability with both SGML and HTML." (Bray, Paoli et al. 2000).

The following example shows an entire XML file (including data) relating to a contact's telephone information. Schematically, Figure 7 shows in graph format the concept "telephone information". Data components are the vertices and relationships are the connecting lines:

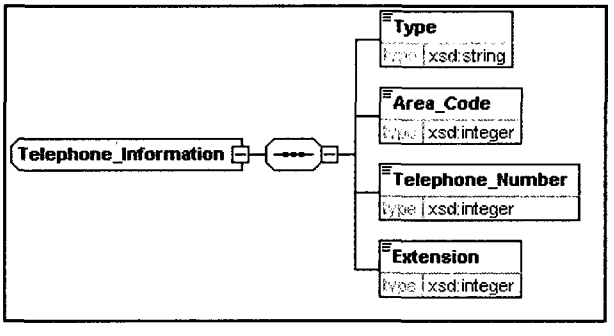


Figure 7: Contact Telephone Information (visual schema)

The first attribute *Type* may have data such as "home", "office", "fax", "mobile". The rest of the attributes, *Area_Code, Telephone_Number* and *Extension*, are self-explanatory. Figure 7 is expressed in XML syntax that yields Figure 8:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<Telephone_Information>
<Type>Office</Type>
<Area_Code>732</Area_Code>
<Telephone_Number>9803884</Telephone_Number>
<Extension/>
</Telephone_Information>
```

Figure 8: Contact Telephone Information (in XML syntax)

The sample XML document shown in Figure 8 is considered "well formed". That is, it adheres to all the basic XML syntax recommendations (Bray, Paoli et al. 2000).

2.5.4 Document Type Definition (DTD)

XML did not support a long lasting need for schema definition and standardization, as provided by data dictionaries embedded in databases. Therefore, the World Wide Web Consortium (W3C) implemented the SGML provision for schema definition by introducing a Document Type Definition (DTD). The purpose of a DTD is to define the building blocks of an XML document. It defines the document structure with a list of elements. A DTD can be declared inside an XML document, or as an external reference. An external DTD for the XML Sample in Figure 8 is given in Figure 9 below:

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT Telephone_Information (#PCDATA | Type | Area_Code | Extension | Telephone_Number)*>
<!ELEMENT Area_Code (#PCDATA)>
<!ELEMENT Extension EMPTY>
<!ELEMENT Telephone_Number (#PCDATA)>
<!ELEMENT Type (#PCDATA)>
```

Figure 9: Contact Telephone Information (in DTD syntax)

Note that it is possible to design a different DTD for the same XML file, in which the tree structure can be preserved. Line 1 in the sample DTD (Figure 9) is a declaration

that the file is an XML file. Lines 2 to 6 declare the vocabulary and typing in allowable DTD syntax. In this example, the types are EMPTY and #PCDATA.

DTD is not written in XML. Therefore, a DTD is not expressive enough to accommodate data modeling needs at a similar level of rigor enabled by a SQL DDL or even a COBOL DDL. To overcome these deficiencies, the W3C created a new recommendation, the XML Schema Definition (XSD), to enhance the DDL capabilities of XML. It also attempts to address lack of semantics. Semantics and additional structure are discussed next.

2.6 Extending DDL to Address Meaning

XML is mostly concerned with syntax but not with the meaning of each data tag. Unfortunately, syntax cannot make sense without semantics when data integration needs to take place. Therefore, new approaches aim at adding more semantic capabilities to XML. The following sections discuss some of these approaches.

2.6.1 Semantics inside XML using XSDL

Liu et al. proposed the XML *Semantics* Definition Language (XSDL) to express and preserve XML author's intended meaning (Liu, Pu et al. 2000). This acronym is not the same as the XML *Schema* Definition Language (XSDL), a W3C recommendation of May 2001. The XML Semantics Definition Language was introduced again by the same authors in 2005, (Liu, Mei et al. 2005). Liu's XSDL adds two features to XML: a formal language for semantics representation and a mapping language for mapping from XML constructs to the formal language.

Liu uses the Web Ontology Language (OWL) as the DDL because OWL has a formal logic foundation and it is an official W3C recommendation (OWL Recommendations 2004). A detailed discussion of OWL is provided later in this research.

Liu uses XML Path Language (XPath) for mapping XML constructs to OWL (XPath and W3C 1999). XPath is a W3C recommendation for traversing through parts of an XML document. It is worth noting that XPath is not expressed in XML syntax. Liu's XSDL was not adopted by the W3C to date.

2.6.2 Semantics inside XML using XML/M for Multimedia Integration

Kim and Park propose a multimedia data model called XML/M, which encompasses diverse types of multimedia data and captures semantic relationships among them (Kim, Park et al. 2005). The model addresses Media Objects that are the basic unit of multimedia data; Relationship Objects specify the relationships among media objects. Container Objects are clusters of semantically related media objects. XML/M unifies different types of multimedia data, but does not propose a novel approach, as it uses wrappers, termed adapters in the aforementioned paper. Wrappers have been used since 1981, and do not provide semantic content, not even in XML/M.

2.6.3 XML Schema Definition (XSD)

An XML Schema Definition (XSD) specifies how to formally describe elements in an XML document and typically has a file extension "XSD" as in *Telephone.XSD*. XSD can define, for instance, the ordering of elements, or what child elements a particular element may have. XSD has two advantages over DTD: (a) XSD is written in XML (b) XSD is more expressive, allowing for typing, restriction to given values, and allowing the formation of complex elements. Many XML parsers have the ability to verify that a given

XML file conforms in vocabulary, structure, and constraints to referenced XSD, when such reference is provided. On the down side, XSD is much more complex than XML or DTD. Figure 8 (XML), Figure 9 (DTD), and Figure 10 (XSD) demonstrate the differences.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xs:schema  xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified">
        <xs:element name="Area_Code">
                <xs:simpleType>
                        <xs:restriction base="xs:string">
                                <xs:enumeration value="201"></xs:enumeration>
                                <xs:enumeration value="609"></xs:enumeration>
                                <xs:enumeration value="732"></xs:enumeration>
                                <xs:enumeration value="908"></xs:enumeration>
                        </xs:restriction>
                </xs:simpleType>
        </xs:element>
        <xs:element name="Extension" type="xs:string"/>
        <xs:element name="Telephone_Information">
                <xs:complexType>
                        <xs:sequence>
                                <xs:element ref="Type"/>
                                <xs:element ref="Area_Code"/>
                                <xs:element ref="Telephone_Number"/>
                                <xs:element ref="Extension"/>
                        </xs:sequence>
                </xs:complexType>
        </xs:element>
        <xs:element name="Telephone_Number" type="xs:string"/>
        <xs:element name="Type" type="xs:string"/>
</xs:schema>
```

Figure 10: Contact Telephone Information in XSD syntax

XML supports namespaces. These are used similarly to table name prefixes in relational databases, such as *Employee.SSN* where *Employee* is a table name and *SSN* is a column. XML namespaces provide a method for qualifying element and attribute names used in XML documents by associating them with namespaces identified by URI

references. XML namespace notations require two lines. One for the identification of the source, and the second to use the source for element qualification, as illustrated in Figure 11. The first line in Figure 11 declares the namespace ("emp"). The second line makes use of the namespace when referring to the attribute SSN.

```
1. emp:http://localhost/myRDFile.rdf
2. emp:SSN
```

Figure 11: XML Namespace Sample

The designers of XSD had hoped that massive support of namespaces will be a significant step towards solving the problem of meaning. It appears they counted on data modelers, systems designers, and other practitioners using agreed-upon name spaces with meaning-providing content. Apparently those hopes did not materialize, as new W3C working groups were formed in 2001 to address the challenge of semantics. The three most notable efforts are groups known as WebOnt, OWL and RFD. Each one is expounded on in the following chapter.

2.7 XML Suggested Standard Schemas Proliferation

The website XML.ORG launched a Schema Registry in 1999. The US Government abandoned its effort in 2004 to create an XML Schema Registry and instead pointed to the OASIS Registry that was later moved to the XML.ORG registry (XML.GOV 2007). That registry is no longer in service (XML.ORG 2007). They do not provide an explanation, but it appears that the registry became useless. In a survey on XML Business Data Exchange Vocabularies (Kotok 2000), taken in January 2000, 124 different XML business vocabularies were either being planned, developed, or currently in use. Sources for the survey included the OASIS/Robin Cover Pages, XML.COM, Schema.Net, and IBM's alphaWorks. The survey also covered XML vocabularies registered with OASIS'

XML.ORG, Microsoft's BizTalk.org portal, and schemas managed by Data Interchange Standards Association (DISA). The survey focused on business to consumer transactions. The number of XML schemas suggested as industry standards grew exponentially from 1999 to 2003 and beyond. The Gartner Group summarized the growth by two comparative graphical depictions. Figure 12 depicts the situation circa 2000, and Figure 13 depicts the situation circa 2003. The written data and the graphical depiction indicate a growth process that went out of control.

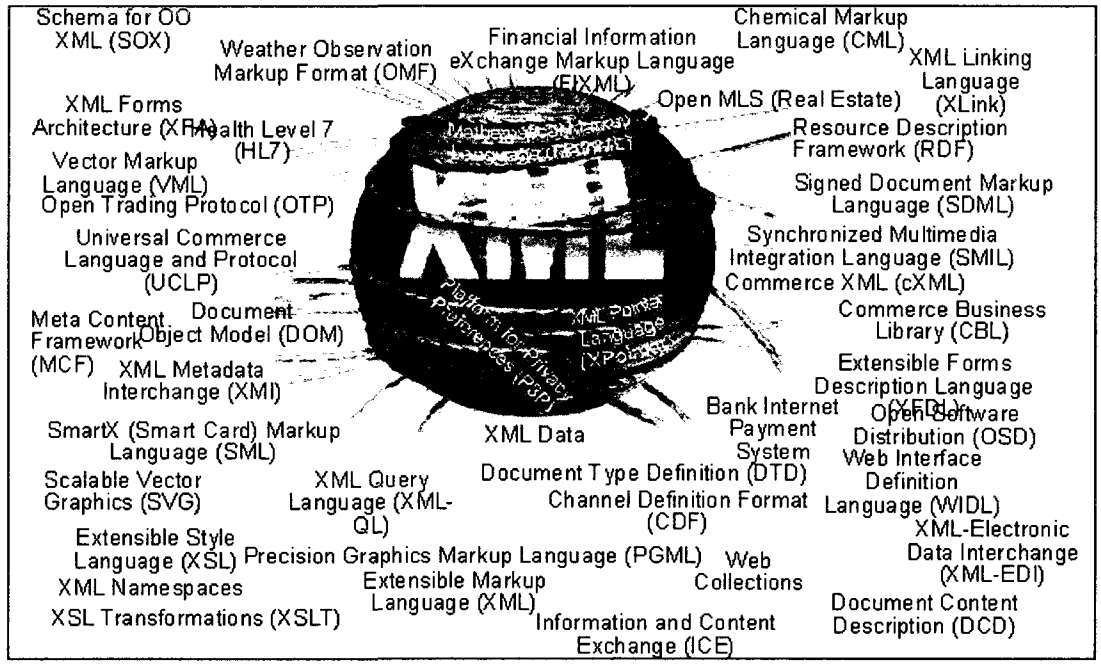


Figure 12: XML Proliferation in 1999 per the Gartner Group

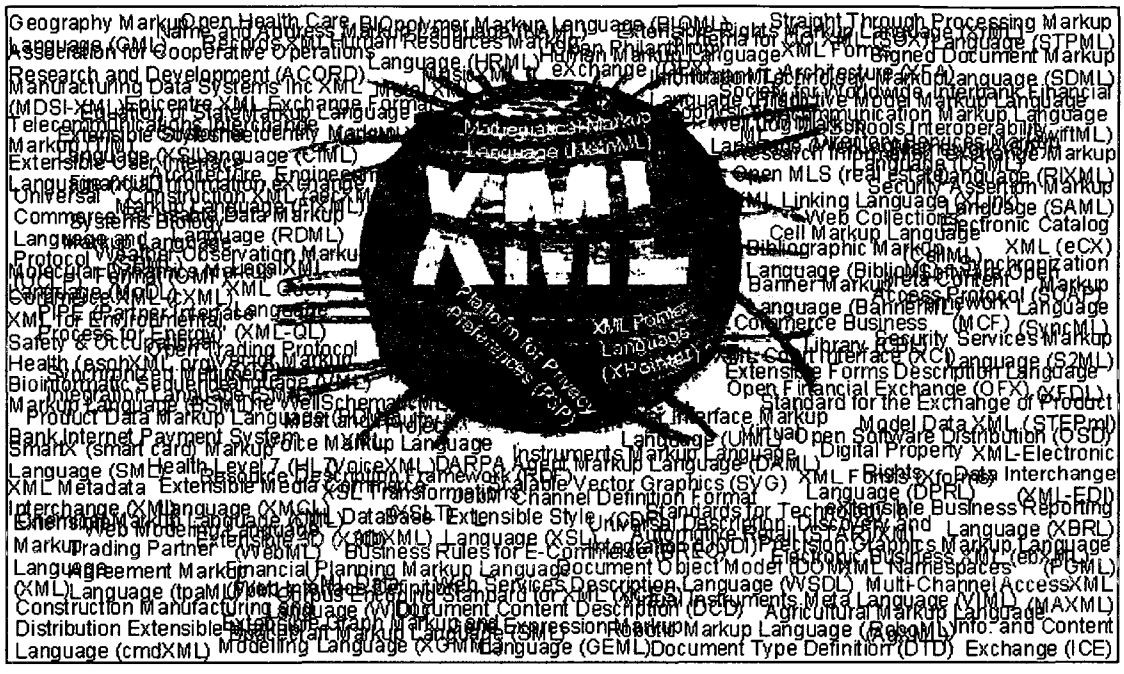


Figure 13: XML Proliferation in 2003 per the Gartner Group

2.8 Chapter 2 Summary and Implications to the Research

Electronic data processing using computers requires the data to be organized in some machine readable form. At the same time, the means of organization need to be human readable, at least to a degree that professionals can master without excessive training. For those reasons a variety of DDLs have been created, correlating with different generations of programming languages and computing paradigms. As socio-financial pressure to integrate autonomous and heterogeneous data grew, new DDLs that claimed support for such undertaking have been created. Different approaches to solving a fundamental problem of conveying and preserving meaning have been suggested. It appears that the challenge has not been overcome as of yet.

Implication of chapter to study	How Implication will be used
Structured DDL	Measure to what degree it supports the sifting through variety, reduces entropy and supports tension when new relation was formed with another data structure
Database DDL	Same as above
Semi-Structured DDL	Same as above
Extending DDL to convey meaning	Same as above

Table 2: Summary of Chapter 2 Implications to the Study

CHAPTER 3

COMPUTERIZED ONTOLOGIES

3.1 Introduction

Chapter 3 concentrates on various efforts to develop computer readable mechanisms to preserve and convey meanings, namely, computerized ontologies. The chapter begins with an explanation of what ontologies are and provides motivation for their need. It then reviews several approaches to designing and implementing computerized ontologies. Some of the difficulties ontologies create are explained in depth, along with several graphical illustrations.

3.2 Computerized Ontologies Explained

Many definitions of ontology have been offered. A commonly cited definition is one offered by Gruber: "An ontology is a formal explicit specification of a shared conceptualization" (Gruber 1993). In the context of knowledge sharing, ontology is a precise description (such as formal specification of a computer program) of concepts that exist in some area of interest and the relationships that hold among them.

Designing and building ontologies that are formal enough to support automated inference is difficult, time-consuming, and potentially expensive, for several reasons. There are a number of competing incompatible syntaxes for creating computerized ontologies. Ontologies require **consensus** across a community whose members may have drastically different visions of the domain under consideration. A variety of strategies exist for reaching consensus. At one extreme, small lightweight ontologies are developed by large numbers of people and then merged via mapping. Lest the small lightweight

ontologies are expressed by different syntaxes, some or all the ontologies in the group need conversion to a common syntax before mapping can occur. Such a conversion does not assure meaning preservation, which is a problem by itself and discussed in a later section. At the other extreme of possible strategies for reaching consensus is the formal and rigorous development of ontologies by consortia and standards organizations. Such organizations move slowly; the consensus view might be too much of a compromise to be effective for a large number of participants.

The next sections discuss data modeling languages for ontologies. Later there is a review of attempts to implement ontology mapping and merging.

3.2.1 Resource Description Framework (RDF)

Resource Description Framework (RDF) (Lassila and Swick 1999) and Resource Description Framework Schema (RDFS) which is the Schema Language for RDF (Brickley and Guha 2000) are both W3C recommendations for describing the content, characteristics and structure of concepts that are defined by the ontology creator. RDF is a language for representing information about resources in the World Wide Web using XML syntax. Such resources can be web pages, files, a specific location or part within a file. Figure 14 illustrates an RDF code fragment for defining some contact information. RDF can be used for defining classes, as shown in Figure 15 and then use the class definition to point to the meaning of a label. The class can be defined outside of a given RDF file. For example, in Figure 16 the meaning of the label "Service Name" exists in an external resource and the RDF points to it using the syntax "rdf:resource="

```
<rdf:RDF xml:lang="en"
        xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
        xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
        <rdf:Description ID="TelephoneInformation">
        <rdf:type resource="http://www.w3.org/22-rdf-syntax-ns#Property"/>
                <rdfs:domain rdf:resource="#AreaCode"/>
                <rdfs:range rdf:resource="#US_AreaCode "/>
        </rdf:Description>
</rdf:RDF>
```

Figure 14: RDF Sample code fragment

```
<rdfs:Class rdf:ID="Service">
<rdfs:label>Service</rdfs:label>
<rdfs:comment>
        A billable entity provided by Super Consultants Inc.
</rdfs:comment>
</rdfs:Class>
```

Figure 15: RDF Schema - direct declaration of the class "Service"

```
<rdfs:Property rdf:ID="serviceName">
 <rdfs:label>Service Name</rdfs:label>
 <rdfs:domain rdf:resource="#Service"/>
 <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#serviceName"/>
</rdfs:Property>
```

Figure 16: Using RDF to define a property of the class "Service"

> *"[RDF] is particularly intended for representing metadata about Web resources, such as the title, author, and modification date of a Web page, copyright and licensing information about a Web document, or the availability schedule for some shared resource. However, by generalizing the concept of a 'Web resource', RDF can also be used to represent*

information about things that can be identified on the Web, even when they cannot be directly retrieved on the Web." (Manola and Miller 2004)

The main modeling primitives defined in RDFS are classes and properties. A class is a resource and has a unique ID. Core classes in RDFS are: rdfs:Resource, rdfs:Property, and rdfs:Class.

An RDFS has modeling primitives for defining property constraints that restrict the range and domain of a property to certain classes. Core properties in RDFS are rdfs:type, rdfs:subClassOf, rdfs:subPropertyOf, rdfs:seeAlso, rdfs:isDefinedBy. RDFS core constraints are: rdfs:ConstraintResource, rdfs:ConstraintProperty, rdfs:range, rdfs:domain.

RDF is machine friendly, but it is not easily read and understood by humans. Some researchers have suggested annotating RDF with human language, for better readability (Jos, Kahan et al. 2001), (Von-Wun, Chen-Yu et al. 2003). "*To render RDF more friendly to humans, we propose to augment it with natural language annotations, or metadata written in everyday language*" (Katz and Lin 2002).

RDF defines three modeling primitives: a subject, a predicate, and an object. For example:

http://www.NJIT.edu/index.html has a **creator** whose value is **pilla**

The subject is the NJIT web page *index.html*; the predicate is the *creator*; the object is *pilla*. The predicate can be expressed by means of yet another resource. One may use a vocabulary published on the web to point to the predicate. For example, the Dublin Core Metadata Initiative (Dublin Core 2005) has an entry and a definition for the term creator in http://dublincore.org/2003/03/24/dces#creator

The object can also be expressed by means of yet another resource. For example:

http://www.njit.edu/phone_ldap/expand-entry.pl?uid=pilla

which points to the author's entry in NJIT's directory. Using the RDF triplet notation, the statement

http://www.NJIT.edu/index.html has a **creator** whose value is **pilla**

can be re-written as:

<http://www.NJIT.edu/index.html>
<http://dublincore.org/2003/03/24/dces#**creator**>
<http://www.njit.edu/phone_ldap/expand-entry.pl?uid=**pilla**>

Figure 17: Sample RDF Triplet

In essence, the predicate is a "data element" if we were to use XML schema terminology, or "column name" had we used relational data base terminology, or "field" in legacy data processing terminology. Unlike other DDLs, in RDF any part of the triplet can be a Uniform Resource Identifier (URI) - the address of an Internet resource. A URI is the unique name used to access a networked resource. It could point to a specific file location , yet it could also be a call to an application or a database.

Using RDF notations, the predicate "creator" in Figure 17 can be substituted with the following (quite obscure) URI: http://kegs.cs.tsinghua.edu.cn/ontology/travel#creator, which in turn can do one of the following:

- Provide a definition (in natural language) for the term Creator or
- Provide a resource for resolution (which can point to another URI ad infinitum) or
- Provide a URI for final resolution other than itself (e.g., the Dublin core http://dublincore.org/2003/03/24/dces#creator, or a less known source, such as http://xmlns.com/foaf/0.1/)

There exists the possibility that the ultimate URI for a specific predicate in two ontologies is the same one. All that is achieved is an expensive and complex standard,

rather than true semantic integration. That is, two Web pages point to a common "ancestor", as illustrated in Figure 18. Note that "floor number" of the Famous Real Estate web page is lost in the transition between intermediate URI, while "altitude" that's part of the vocabulary resolution is never used at Joe's Real Estate web page. Having a consensus ontology is desirable in many cases, but it is not a feature of the DDL. Using an agreed-upon ontology is subject to the designer's preferences or knowledge, office politics and even luck.

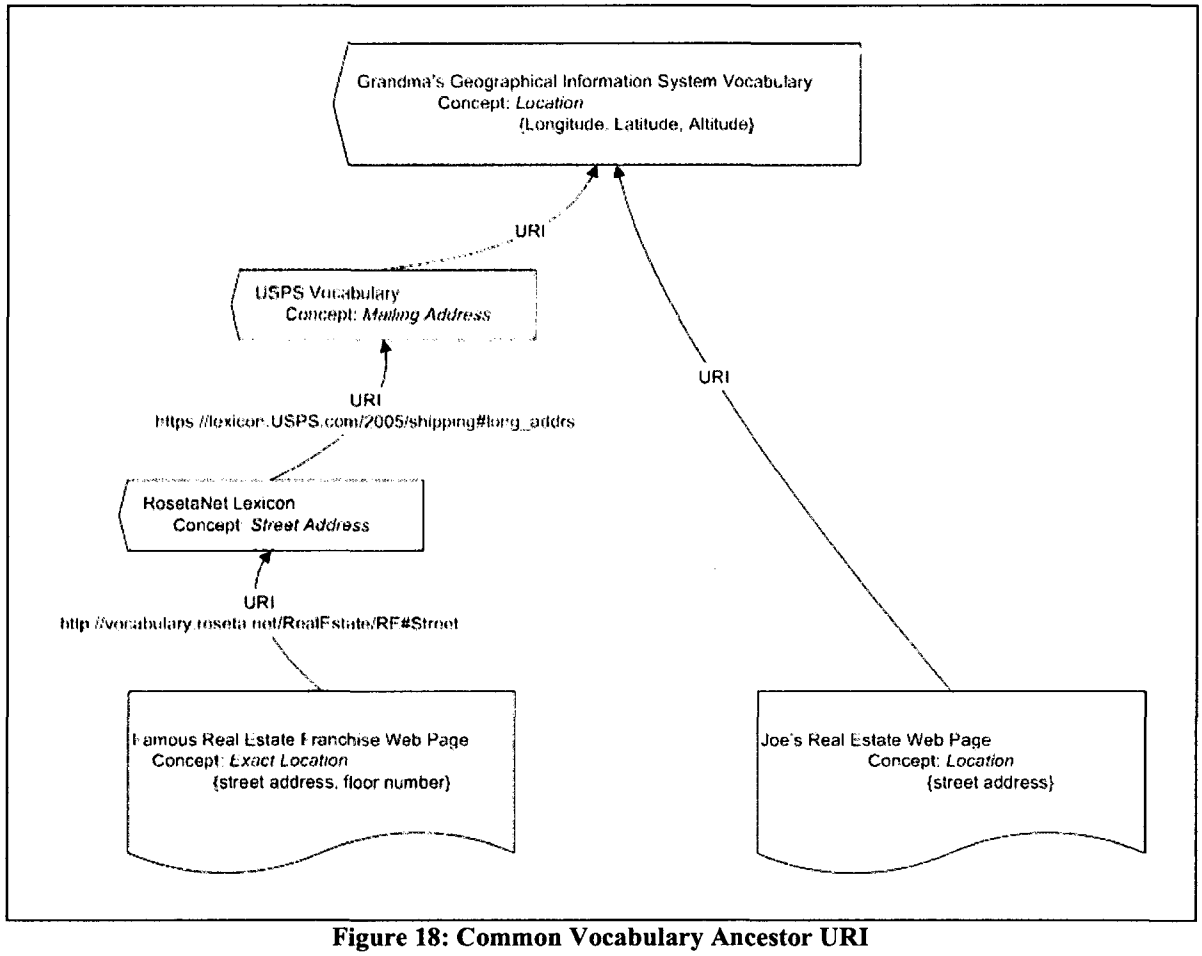


Figure 18: Common Vocabulary Ancestor URI

If the two predicates point to two different URI's, then the semantic barrier still exist. For example, in the TSINGHUA ontology the resolution for the predicate "name"

is found in the TSINGHUA ontology itself. In contrast, the concept *Author's Name* in the BlueSky Travel webpage points to the Dublin Core taxonomy as illustrated in Figure 19. The end result is two predicates, "name" and "creator" that are independent and need mapping to each other if they are equivalent.

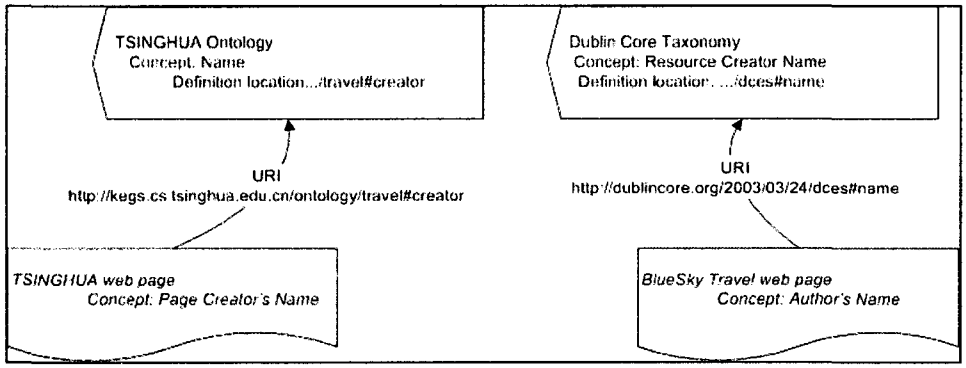


Figure 19: Sample for no-common URI

Ontologies that depend on other ontologies for resolution present a danger of having a circular reference, yielding an endless loop in lieu of a resolution to a given predicate. Figure 20 has an illustration of such a possibility. The designer of the Famous Real Estate Web page uses a concept termed "exact location" for real estate property. The concept is three dimensional, in that it includes the floor number in a high rise building. The ontology's designer points to the RosetaNet lexicon as a resource for resolving "street address", which is part of the "exact location" concept. In turn, RosetaNet points to the United States Postal Service (USPS) vocabulary for resolution of "street address". USPS points to a Mortgage Industry Standards Maintenance Organization (MISMO) like vocabulary. Unbeknown to the web page creator, the MISMO-Like resource points to RosetaNet's lexicon for the semantic resolution of "street address". This is not necessarily the result of a sloppy design. It is possible that when the web page was created there was no pointer from the MISMO-Like resource to RosetaNet. Rather, the pointer was created later.

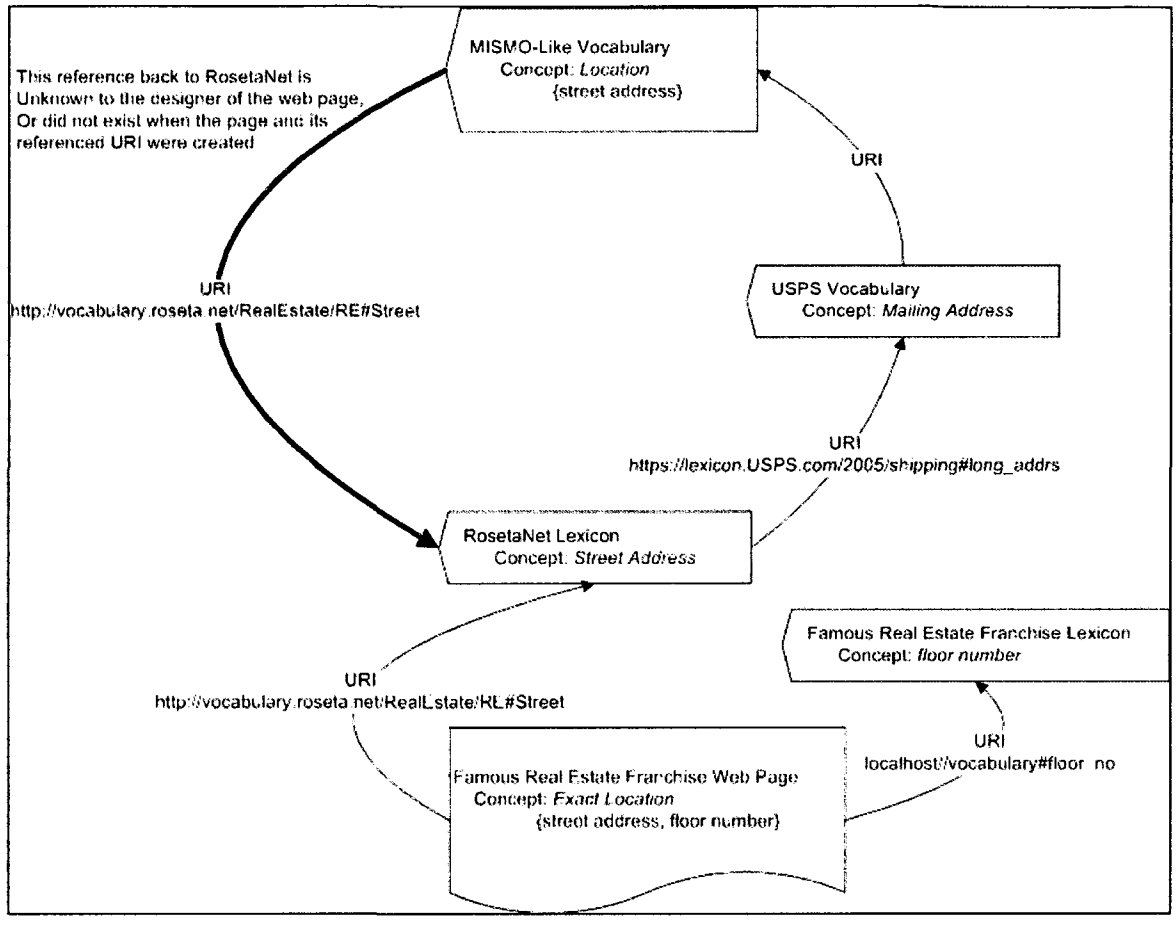


Figure 20: Circular Reference to an Ontology

In summary – RDF points to resources on the Web where an application may get the semantics of a given component in a semi-structured file. Users of RDF run the risk of having circular references. There exists the risk of referencing equivalent terms in two unrelated ontologies, where mapping is required for satisfactory resolution. There exists the possibility of indirectly referencing the same top-level resource, using a de-facto common standard, reached through a labyrinth of URI pointers.

3.2.2 SHOE

Simple HTML Ontology Extensions (SHOE) broadens HTML by allowing annotation of web documents with a machine-readable knowledge representation language. SHOE has

two categories of tags: Ontology Construction Tags and Web Page Annotation Tags. Ontology Construction Tags are expressed in XML syntax, using a set of predefined words. A mandatory Ontology Construction Tag that must appear in an HTML document enhanced by SHOE is <ONTOLOGY>, and there must exist a tag terminator such as <ONTOLOGY/> since XML syntax requires it. Every <ONTOLOGY> tag must have an ID and a VERSION associated with it. Ontology declarations may appear at the top level within the body of an HTML document, but they cannot contain HTML tags. The SHOE tags are used just as HTML tags, but they are not part of the HTML set of tags. For those extra tags to work one must first declare the META HTTP-EQUIV tag "SHOE" such as in <META HTTP-EQUIV="SHOE" CONTENT="VERSION=1.0">. One usage is augmenting HTML hyperlinks with a <RELATIONSHIP> tag that can tell a little more about the relationship between the link and the target. It is required to declare the relationship first, and use it later. For example, if there is a declaration of a relationship to an "advisor" (say, in a university) then one can use it as shown in Figure 21.

```
<RELATION NAME=" advisor">
        <ARG POS=TO VALUE="http://dir.njit.edu/list.asp?uid=klashner">
</RELATION>
```

Figure 21: Sample of SHOE extension

To use SHOE one must locate an applicable ontology or create one and then add SHOE tags in the web page (Luke, Spector et al. 1996; Heflin 2000) and hope that some other software will "understand" the ontology in use within the page when the page is consumed by that other software.

3.2.3 DAML, OIL, DAML+OIL

The DARPA Agent Markup Language (DAML) Program officially began in August 2000, a language for expressing more sophisticated class definitions than permitted by RDF. DAML's goal was to develop a language and tools to facilitate the concept of the Semantic Web, which is "*an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation*" (Berners-Lee, Hendler et al. 2001). The DAML group also developed the Ontology Inference Layer (OIL), in an effort to provide sophisticated classification, using frame based constructs. A frame (Minsky 1975) is a structure for representing a *concept* or *situation* such as "police station" or "being in a jail cell". The latest release of the language named DAML+OIL provided a rich set of constructs with which to create ontologies and to markup information so that it is machine readable and understandable. DAML+OIL had facilities for data typing based on the type definitions provided in the W3C XML Schema Definition Language (XSDL). In addition, a 2001 commissioned Web Ontology Working Group (WEBONT 2001) has taken on the task of producing an ontology language, with DAML+OIL as its basis. As a result the W3C has created OWL, a semantic markup language for publishing and sharing ontologies on the World Wide Web. The DAML+OIL project was closed in late 2003, and the working group dissolved.

3.2.3.1 ONTOLOGY INFERENCE LAYER (OIL)

The aforementioned DAML group proposed the Ontology Inference Layer (OIL) for web-based representation and inference layer for ontologies. The approach combines modeling primitives from frame-based languages with the formal semantics and reasoning services provided by description logics, a family of knowledge representation

languages that have been studied extensively in Artificial Intelligence over the last two decades.

OIL presents a layered approach to a standard ontology language. It is similar to the layers approach of the Open System Interconnection (OSI) seven layers model defining a networking framework for implementing communications protocols. (OSI has bastardized the notion of "open systems". The reader will recognize OSI is completely closed after reading the discussion on open systems in chapter 6). Each additional layer in OIL adds functionality and complexity to the previous layer. Agents (humans or machines) who can only process a lower layer can still partially understand ontologies that are expressed in any of the higher layers. Figure 22 sketches the relation between the OIL dialects and RDFS.

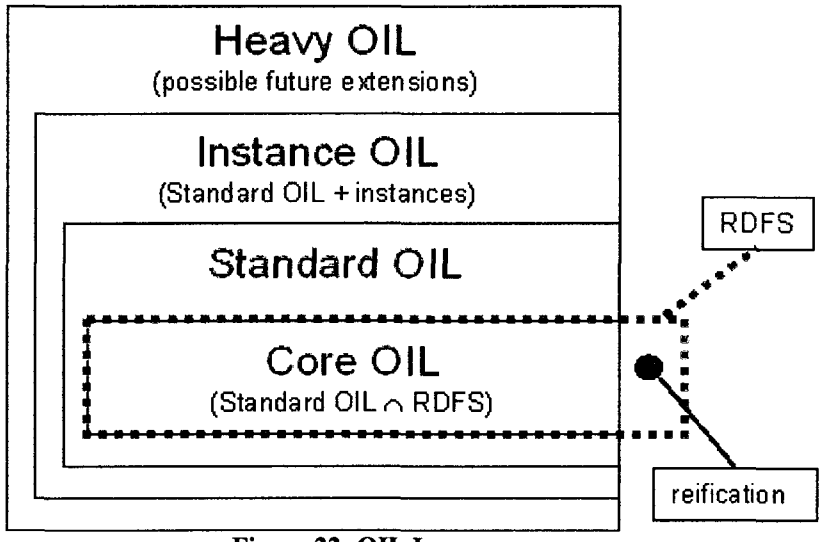


Figure 22: OIL Layers

Core OIL coincides largely with RDF Schema (W3C RDF Schema 2000). Simple software agents that can process RDFS should be able to process the OIL ontologies, and pick up as much of their meaning as possible with the agents' limited capabilities.

Standard OIL is a language intended to capture modeling primitives to provide expressive power and thus should allow semantics to be precisely specified and complete inference to be viable.

Instance OIL is supposed to provide individual integration capabilities by including full-fledged database capability.

Heavy OIL may include additional representational (and reasoning) capabilities. Its syntax is not yet defined, and it has no RDF Schema to support it as of yet. Since the entire DAML+OIL approach was dropped by DARPA and the W3C, it is unlikely these plans will ever materialize (Greaves 2004).

The DAML+OIL project was closed in late 2003, and its working group dissolved. DARPA and the DAML group moved on to develop the Web Ontology Language (OWL) based on RDF. Mark Greaves, DAML Program Manager articulated the new DAML directions for fiscal year 2004 in writing. "*Programs live and die at DARPA by their ability to continuously demonstrate to the DARPA director that they are... creating and nurturing revolutionary advances in technology... DAML has not been equally successful in fulfilling its other broad program objective: seeding this new capability so that others can pick it up. FY04 is, for all intents and purposes, the final year of the DAML program.*" (Greaves 2004).

3.2.4 OWL

The W3C Web Ontology Working Group develops the Web Ontology Language (OWL) as a replacement for DAML+OIL. OWL is primarily designed to represent information about categories of objects and how objects are interrelated. OWL is developed as a vocabulary extension of RDF and is derived from the DAML+OIL Web Ontology

Language. OWL is a formal language for representing ontologies in the Semantic Web. OWL has features from several families of representation languages, including primarily Frames (see page 52) and Description Logics (see page 52). OWL is a vocabulary extension of RDF (Horrocks, Patel-Schneider et al. 2003; Bechhofer, van Harmelen et al. 2004) and it has an RDF schema defining its vocabulary. The vocabulary consists of "reserved words". It takes the fact-stating ability of RDF and the RDFS class- and property-structuring capabilities, and extends them such that OWL specifies logical combinations as intersections, unions, or complements, as one finds in SQL DDL.

OWL documents (frequently called OWL ontologies) are RDF documents. The root element of an OWL ontology is an rdf:RDF element which also specifies a number of namespaces as shown in Figure 23 (OWL Document Header) below:

```
<rdf:RDF
   xmlns:owl ="http://www.w3.org/2002/07/owl#"
   xmlns:rdf ="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   xmlns:xsd ="http://www.w3.org/2001/XLMSchema#">
```

Figure 23: OWL Document Header

An OWL ontology may start with a collection of assertions grouped under an owl:Ontology element, and may have classes, as demonstrated in Figure 24 below:

```
<owl:Ontology rdf:about="">
   <rdfs:comment>Sample OWL Ontology</rdfs:comment>
   <owl:priorVersion
         rdf:resource="http://www.mydomain.edu/uni-ns-old"/>
   <owl:imports rdf:resource="http://www.nj_domain.gov/DMV"/>
   <rdfs:label>Dept. of Motor Vehicle Ontology</rdfs:label>
   <owl:Class rdf:ID="Driver">
         <rdfs:subClassOf rdf:resource="#NJ_TaxPayer"/>
         <owl:equivalentClass rdf:resource="#WorkingPeople"/>
   </owl:Class>
</owl:Ontology>
```

Figure 24: Simplified example of owl:Ontology and owl:Class elements

Two OWL versions are currently available: OWL-Lite and OWL-DL. The first is a subset of OWL and the later is a sublanguage of OWL that places a number of constraints on the use of the OWL language constructs, due to implementation difficulties of full blown OWL. Table 3 summarizes the vocabulary terms available for OWL-DL at the time of this writing (OWL Recommendations 2004):

OWL Vocabulary Terms		
Owl:AllDifferent	owl:allValuesFrom	owl:backwardCompatibleWith
Owl:cardinality	owl:Class	owl:complementOf
Owl:DatatypeProperty	owl:DeprecatedClass	owl:DeprecatedProperty
Owl:differentFrom	owl:disjointWith	owl:distinctMembers
Owl:FunctionalProperty	owl:hasValue	owl:imports
Owl:incompatibleWith	owl:intersectionOf	owl:InverseFunctionalProperty
Owl:inverseOf	owl:maxCardinality	owl:minCardinality
Owl:Nothing	owl:ObjectProperty	owl:one of
Owl:onProperty	owl:Ontology	owl:priorVersion
Owl:Property	owl:Restriction	owl:sameAs
Owl:sameClassAs	owl:sameIndividualAs	owl:samePropertyAs
Owl:someValuesFrom	owl:SymmetricProperty	owl:Thing
Owl:TransitiveProperty	owl:unionOf	rdf:List
rdf:nil	rdf:type	rdfs:comment
Rdfs:Datatype	Rdfs:domain	rdfs:label
Rdfs:Literal	Rdfs:range	rdfs:subClassOf
Rdfs:subPropertyOf		

Table 3: OWL Vocabulary Terms

3.3 Suggested Standard Ontologies Proliferation

At the time of this writing, ontologies have not become main stream technology. Further, it appears that vendors who invested significant resources in adopting XML and in an attempt to be market leaders do not exhibit the same level of enthusiasm and commitment to OWL. This may explain why we were unable to find Ontology Registries similar to the XML registries discussed in section 2.7 on page 39. Rather, there is an extensive registry sponsored by the PROTÉGÉ Ontology Editor (see page 78 for details), which is an

academic research project of Stanford University. The Ontology Registry (Protege 2007) lists 85 independent ontologies. The DAML Ontology Library lists 282 ontologies on April 30, 2004. However, the DAML project and its associated technologies were abandoned by the US government in the same year, making the list an anecdotal electronic graveyard, at most (daml.org 2004). The Open Biomedical Ontologies Foundry lists 67 life-sciences related ontologies along with a message asking for new ontology contributions (OBO 2007). It appears to have an overlap with the Plant Ontology Consortium repository of ontologies (POC 2007), which is work in progress.

Although we do not have dramatic graphical depiction of the proliferation of ontologies (as we have for XML), the trend is clear. The current status resembles XML in its infancy, with similar potential out of control growth characteristics.

3.4 Chapter 3 Summary and Implications to the Research

This chapter provided a comprehensive overview of the existing landscape of computerized ontologies. It explained some of the difficulties ontologies introduce, and illustrated how the key challenge of providing and preserving meaning is not yet fulfilled.

Implication of chapter to study	How Implication will be used
Computerized Ontologies	Measure to what degree it supports the sifting through variety, reduces entropy and supports tension when new relation was formed with another data structure
Mapping	Assess the degree of mapping possible in light of potential ambiguities

Table 4: Summary of Chapter 3 Implications to the Study

CHAPTER 4

APPROACHES TO DATA INTEGRATION

4.1 Introduction to Chapter 4

Data integration focuses on meaningful unification of distributed, heterogeneous sovereign data sources. It is an active area of research in the database community. Papers from as early as 1992 to recent years are a manifestation of both the interest and the difficulty this area of research exhibits. For example: (Wiederhold 1992; Arens, Chee et al. 1994; Garcia-Molina and al. 1995; Quass, Rajaraman et al. 1995; Levi, Rajaraman et al. 1996; Bayardo, Bohrer et al. 1997; Duschka and Genesereth 1997; Tomasic , Amouroux et al. 1997; Liu, Pu et al. 2000; Draper, Halevy et al. 2001; Thakkar, Knoblock et al. 2003; Kolaitis 2005; Yu and Popa 2005).

Now that various DDLs designs have been explained, the chapter reviews approaches suggested and implemented for integration using the DDLs described hereto. The chapter has three parts: integration of structured data, integration of semi-structured data, and integration of, or supported by, ontologies. There is a common problem thread in all approaches, namely the correct mapping of one data structured expressed in a given DDL to another data structure, usually expressed using the same DDL (although this is not a prerequisite).

4.2 Integration of Structured Data Sources

4.2.1 Standards Based Data Exchange

Data defined using any of the aforementioned structured DDLs (or similar ones) has been exchanged among systems and organizations. DDLs convey syntax, but do not convey semantics. For example, the Basic programming language (originally designed in 1963 by Kemeny and Kurtz at Dartmouth College) constraint naming data elements to only two alphanumeric characters. Cobol DDL supports syntax for hierarchies and longer data element names. However, the exact meaning of a data element is not to be found in the data structure. The meaning depends, among other things, on the context in which the data element plays a part. Therefore, semantics are exchanged either by negotiated mutual agreement between business partners or by an industry standard. In both cases the result is a set of data items named and a place in the structure in a manner that conforms to a prescribed set of constraints.

The following sections describe different data integration projects that use official industry standards or proposed standards.

Metadata for integration across computerized manufacturers. Industry leaders (ALCOA, DEC, GE, GM, IBM and others) sponsored the 1987 *Metadatabase* project through Rensselaer Polytechnic Institute's Computer Integrated Manufacturing Program. The Metadatabase project was a multi-year research effort seeking to "*develop concepts, methods and techniques for achieving information integration across major functional systems pertaining to computerized manufacturing enterprises*" (Hsu 1991). The research concentrated on:

1. heterogeneous, distributed and autonomous databases
2. information resources management
3. integration of information systems

One of the Metadatabase core components is the Global Information Resources Dictionary (GIRD) model. It is used as a unified representation of information-bearing entities characteristics to aid in the identification, discovery, assessment, and usage of such entities, better known as metadata. GIRD was used for the management of both data and knowledge, such as business rules and process control rules (Hsu and Rattner 1993). The researchers also hoped to develop "*a theory of information requirements for integration*" (Hsu 1991), which has not materialized, according to a 2003 Metadatabase web page (Metadatabase 2003). This website proposes a procedural model for data integration in a manufacturing environment. The research brought to light the need for a "common language" among the integrated systems, a way to model data and knowledge, a need for a global query formulation and processing mechanism. Hsu writes:

> *"Foremost is the need to incorporate contextual knowledge with databases because computerized manufacturing enterprises often include various knowledge-based systems that are an integral part of the overall integration and, because the functional contexts in which individual database systems contribute to enterprise-wide synergy must be sufficiently represented."*

Hsu makes a case that at least three major classes of contextual knowledge need consideration: Business rules and operating knowledge, controls and sequences of sequential interactions among systems and, decision knowledge for parallel interactions among systems. The latest paper relating to the project was written by Harhalakis et. al. in 1994. The paper has a rather narrow focus on "*rule-based implementation of*

specifications of integrated manufacturing information systems". Their approach uses Petri-Nets, a 'hot' research topic of the mid-1990's (Harhalakis, Lin et al. 1994). It appears that the Metadatabase project never materialized to a working industry standard. The project, with its rich metadata dictionary was abandoned.

Electronic Data Interchange (EDI). Hsu was not the first one to recognize the need for a "common language" among integrated systems. EDI established a common language for exchanging business related transactions via the creation and enforcement of a standard. EDI began in the United States around 1968, when two organizations started exchanging point-to-point information between their computers through private telecommunication networks. The first transactions to be transmitted were invoices and bills. Organizations transmitted these documents electronically through communication lines instead of writing and mailing them. Since the transactions did not transit through a physical mailbox, they were more rapid and the organizations saved both time and money. Prospective partners had to first agree upon the procedures of transmission and on the internal standards to be used when sending and receiving data.

Exchanging data was relatively easy when only two partners were involved but became more complex when many partners using different computer systems, protocols and interfaces decided to exchange data through EDI (Emmelhainz 1990). By 1975, many vertical business groups interested in EDI were actively working on defining their own domain specific EDI standards. Such standards emerged in several market segments, such as:

- transportation industry represented by a consortium named Transportation Data Coordinating Companies (TDCC)
- food preparation and sales industry voluntarily governed by the Food Marketing Institute (FMI)
- government agencies providing Medicare benefits
- the International Association of Industrial Accident Boards and Commissions (IAIABC).

During 2001, Medicare processed 157,306,245 electronic media claims using EDI, accounting for over 97% of all claims processed by Medicare. The number of EDI claims rose to 170,558,776 in calendar year 2004 (Medicare 2005).

The IAIABC is association of government agencies that administer and regulate their jurisdiction's workers' compensation acts. It creates, maintains and publishes EDI standards that are specific for workers' compensation insurance claims. "With over 300 members, the IAIABC represents a diverse group of workers' compensation professionals, medical providers, insurers, and corporate agencies" (IAIABC 2005). The latest IAIABC EDI standard, Release 3, was made available in 2004. In the State of Minnesota alone, there are approximately thirty (30) state jurisdictions that currently participate in or are planning to use EDI communications with their trading partners using the various IAIABC release standards (MDLI 2005). An EDI implementation requires skilled personnel to use specialized supporting software to map data from the organization's native format to EDI and vice-versa. Once the mapping is complete and verified, relevant flow of information between trading partners needs no human intervention unless an error occurs in the process. EDI is a vibrant and still growing

approach to data exchange and integration among autonomous and heterogeneous systems. However, *all* EDI implementations rely upon industry consensus regarding their voluntary governance through standards, wherein ad hoc or negotiated agreements between parties are exceptionally rare.

Society for Worldwide Interbank Financial Telecommunication (SWIFT). The SWIFT created a key "common language" for exchanging financial related transactions via the creation and enforcement of standards. The SWIFT runs a worldwide network by which messages concerning financial transactions, such as payments, letters of credit, securities transactions, foreign exchange, and others are exchanged among financial institutions (Walmsley 1992). In the year 2000, the SWIFT carried about 1,200,000,000 messages. As of December 2001, the SWIFT linked over 7,000 financial institutions in 194 countries and carried payment messages averaging more than six trillion US dollars per day. On 30 June 2005, there were 11,487,827 messages processed, a new daily peak for the SWIFT. As of June 2005, there were 7,668 active SWIFT participating institutions in 203 countries (SWIFT 2005). The SWIFT website reads:

> *"Standards are an essential element of SWIFT's global offering. We are committed to the collaboration of efforts and convergence of standards, so that our community can benefit from potential cost savings, eliminate redundancies and smoothly expand into previously untapped markets."* (SWIFT 2005)

The SWIFT attempts to satisfy the needs of vertical markets. For example, in October 2001, the SWIFT announced plans to migrate its securities industry standards from ISO 7775 to ISO 15022 XML standard. The SWIFT planned that by year-end 2004 the majority of securities transaction messages, for the front and back office, would use

ISO 15022 XML. It is a set of syntax and message design rules, a dictionary of data fields and a catalogue for present and future messages.

The SWIFT was appointed by the ISO to serve as the sole registration authority for ISO 15022 XML. In this role, the SWIFT maintains a Data Field Dictionary (DFD), and a Catalogue of Compliant Messages. The society has been given the mandate by its members and the ISO to create new fields as necessary. The SWIFT also enforces compliance of messages built directly by communities of users, who develop messages based on needs. Thus, the SWIFT is a vibrant and still growing organization. It determines the solutions to financial data exchange and integration among autonomous and heterogeneous systems worldwide. Implementation of the SWIFT standards relies on industry consensus based standards, where ad hoc and negotiated agreements between specific parties are unheard of.

4.3 Data Exchange Using Negotiated Agreements

When two organizations negotiate exchange of data, the two parties to the exchange need to reach an ad hoc agreement on the file structure and the meaning of each field in the structure. Alternatively, they can agree to use a pre-negotiated agreement, such as an industry standard file structure if one exists (Rohn 1983; Rohn 1985). The following sections describe several industry projects whose data exchange used agreed-upon data structured using compatible DDLs.

4.3.1 Car Insurance Data Exchange

Many Israeli government employees in the 1980's enjoyed a benefit of having part of their car insurance premium paid by the Israeli government. The Israeli Ministry of Defense paid such premiums directly to insurance companies. The Israeli automobile

insurance industry had a predefined file layout for such data exchange. An officer of an insurance company handed over in person the file layout along with written documentation to the Ministry of Defense Project Manager. The insurance company authorized one of its technical persons to provided oral explanations in an ad hoc informal training session given to the Ministry of Defense technical personnel. The file format was incorporated into custom written programs to extract relevant Ministry of Defense employee data for monthly data exchanges thereafter, using tapes that were readable by the receiving organization (Rohn 1982).

4.3.2 Intra-Company Data Exchange

Nabisco's Computerized Information Processing and Production System (CIPPS) project depended on negotiated data exchange between two systems within the same organization. CIPPS was designed and implemented between 1992 and 1995 on an IBM Mainframe s/370 using Software AG's Natural programming language and IBM's DB2 database. CIPPS supported multiple business processes. The *Formula* (recipe) sub-system was mainly used by Nabisco's R&D personnel for recording experimental formulas, and logging of the experiments' outcome. The sub-system was also used to maintain existing formulas used in production, and to archive formulas no longer in use. The *raw ingredients* sub-system assisted in enforcing FDA food safety regulations. The *pre-production* sub-system was responsible for calculations of batches tailored to the specifics of 54 different production lines in use, taking into account a mixer's capacity, oven temperature, and several other parameters. The *packaging* sub-system assisted in management of packaging, including optimization of box arrangement on pallets, depending on box physical measurements and the ability to sustain compression forces. A

different system at Nabisco, written mostly in Statistical Analysis System (SAS), was responsible for the calculations of label nutrition facts as required by the FDA. CIPPS needed to feed the *Nutrition Facts* sub-system with data extracted from the CIPPS *Formula* sub-system. The data was exchanged using a sequential file, whose layout and associated semantics were the result of several meetings between technical personnel from both systems. The file layout was modified numerous times while undergoing testing, to accommodate needs unforeseen during the first meetings (Nabisco Inc. and Rohn 1993).

4.4 Semi-Structured Data Integration

The objective of semi-structured data integration is to allow the unification and thus the examination of independent data sources as if they were a single source. Data originating from autonomous and heterogeneous sources will not necessarily have an agreed upon structure, be it an industry standard or a negotiated data structure. Therefore, semi-structured data integration has several phases: discovery of a source's internal schema, wrapping it with XML to create an external local schema, homogenization of the external local schemas and integration of the homogenized schemas into a global schema as illustrated in Figure 25. The data itself remain at the sources and is accessed by users via queries against the global schema using various query strategies. Figure 26 provides a more detailed overview of a typical integration system. There has been work published in each area that is relevant to this review. Subsequent sections expand on each of these steps.

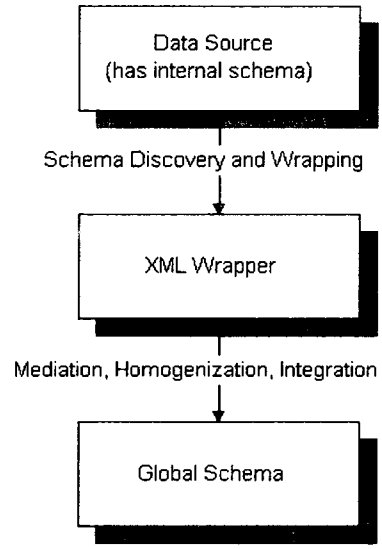


Figure 25: XML Integration Process Overview

4.4.1 The Integration Process

The objective of integration systems is to allow the unification and thus the examination of independent data sources as if they were a single source. Queries are formulated in terms of a global schema that the system translates into local schema terms. Re-written queries interrogate the local data sources, retrieve the results, and the system combines them into a unified response provided to the user.

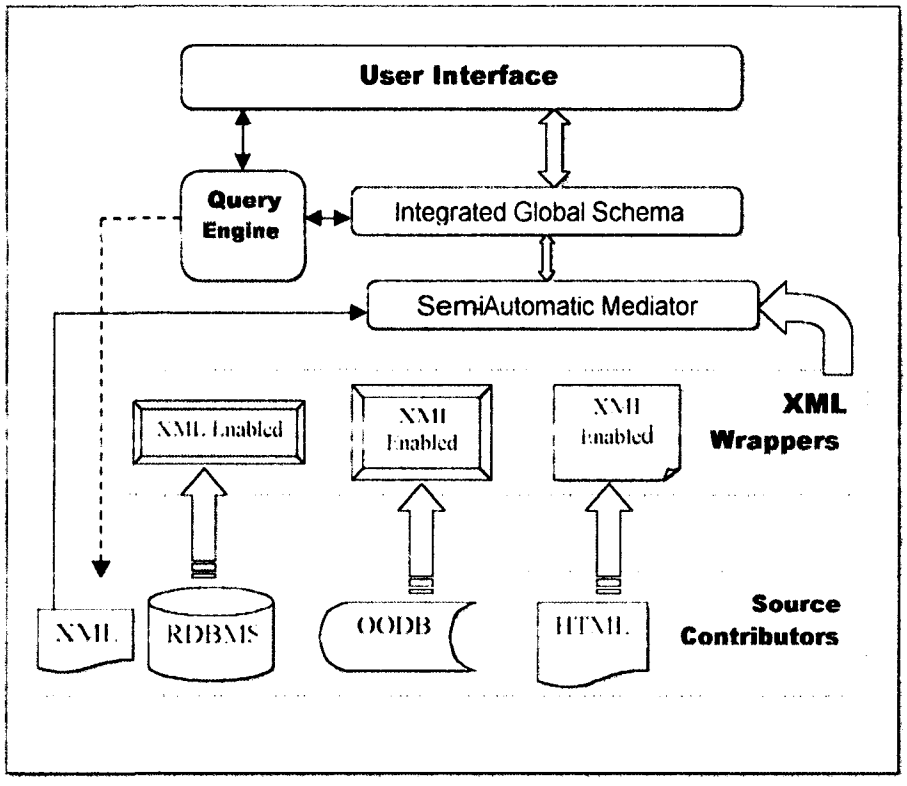


Figure 26: XML Integration Details

Generally, combining data from different sources is divided into two steps. The first step is homogenization of data and the second is integration of homogeneous data. Many researchers used this approach during the late 1990's when XML integration was at the center of interest in some research communities. Here are two examples: Yan at University of Alberta used a two-step approach to build a middleware, "AURORA", for electronic commerce that integrates the catalog information from a large number of data sources (Yan 1997). AT&T developed STRUDEL to manage semi-structured data on the Web (Fernandez, Florescu et al. 1997). STRUDEL uses wrappers and a global mediator

to integrate the data from various sources. These and other projects use a mediator between the XML wrappers and the Global Schema. A mediator is "*a complex software component that simplifies, abstracts, reduces, merges, and explains data*" (Tomasic , Amouroux et al. May 1997). STRUDEL is covered in detail later in this chapter.

4.4.2 Structure Extraction

It is necessary for non-XML sources to provide XML tags first. The goal is achieved through re-engineering the source by explicitly inserting XML tags around it. The result is called a "wrapper". There are three main methods for discovering a source's internal schema: Manual, Semi-Manual, and Automatic.

Hammer et al. describe a manual tool, which the user applies for hand-coding wrappers in their documents (Hammer, Garcia-Molina et al. 1997). The process was adequate as a learning step, yet its manual nature renders it unacceptable for volume processing.

NoDoSE is an open architecture, structural mining tool for plain text and for HTML documents (Adelberg 1998). The plain text part requires some user input, hence the semi-automatic notion. When NoDoSE processes HTML documents, it scans for HTML tags, font information and indentation and uses them to extract a document's schema. A human operator reviews the results and makes corrections when needed. The dependency on manual intervention renders NoDoSE unacceptable for volume processing.

WebView is a tool for retrieving internal structures and extracting information from HTML documents (Lim and Ng 1999). WebView constructs a semi-structured graph (already explained in chapter 3) of a given HTML document, and captures the internal structure of data embedded in the document and its directly and indirectly linked

documents. WebView also provides query-processing capability for evaluating SQL-like queries that are executed against the source documents.

An XML-enabled wrapper was build by Liu and her group at Georgia Institute of Technology. The software extracts the content from web pages and encodes it explicitly as XML tags in a wrapper document (Liu, Pu et al. 2000). The tool is limited to HTML sources with a known structure. If a structure changes, it requires changes to the tool. Non-HTML sources are excluded from the tool all together.

The University of Washington Tukwila project headed by Alon HaLevy proposes a data integration system with a focus on querying capabilities. "*The Tukwila data integration system is designed to scale up to the amounts of data transmissible across intranets and the Internet (tens to hundreds of megabytes), with large numbers of data sources*" (Ives, Halevy et al. 2001). The system depends on a global schema to represent a particular domain. Data sources are mapped as views over the global schema. The mediation process of schema sources into a global schema is done semi-manually.

4.4.3 Wrappers and Schemata Integration

There have been many research projects centering on data integration in the last eight years. This section reviews projects that have been widely cited: Garlic, The Information Manifold, Disco, Tsimmis, YAT, Ozone, Xyleme, WHIRL, Lorel and Strudel.

Garlic. Garlic's goal is to enable large-scale multimedia information systems by combining multimedia information from various systems while maintaining the independence of the data servers and without creating copies of their data (Carey, Haas et al. 1995). It allows query by content of any type of data. Garlic addresses complex issues

related to integration of multimedia information, such as dealing with different interfaces for several different data systems, locating the right system to handle each part of a query, optimizing the accesses to the various data systems and combining the retrieved results into a meaningful form for the user.

The Garlic system can process various data repositories including relational and non-relational database systems, file systems, document managers, image managers and video servers. For each data repository there exists a repository wrapper, which has two main functions: 1) exporting data types and description to Garlic's repository. 2) Translation of data access and manipulation request from Garlic's internal protocols to the repository's native protocol.

Garlic's image data query facility is a research prototype image retrieval system that uses the content of images as the basis of queries. The content used by the query includes the colors, textures, shapes, and locations of user-specified objects (e.g., a person, flower, etc.) or areas (e.g., the sky area) in images, and/or the overall distribution and placement of colors, textures, and edges in an image as a whole. Queries are posed graphically/visually, by drawing, or selecting examples of what the inquirer desires to retrieve. Figure 27 shows an example of visual query performed on Garlic and its results.

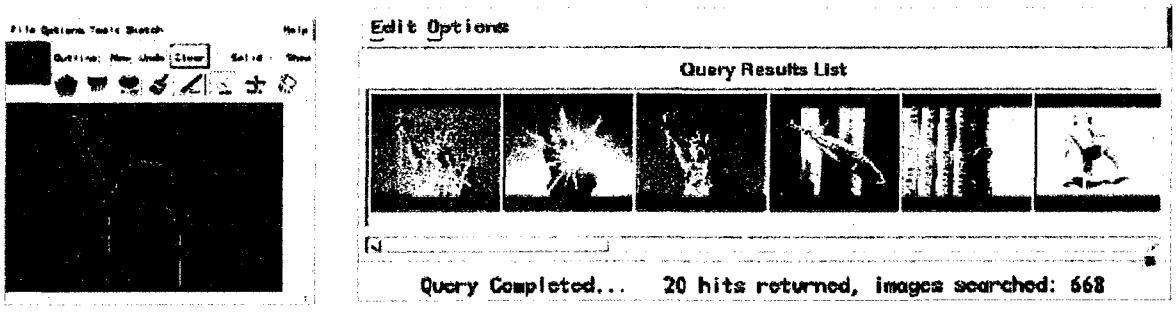


Figure 27: Garlic Query by Sketch Results Set

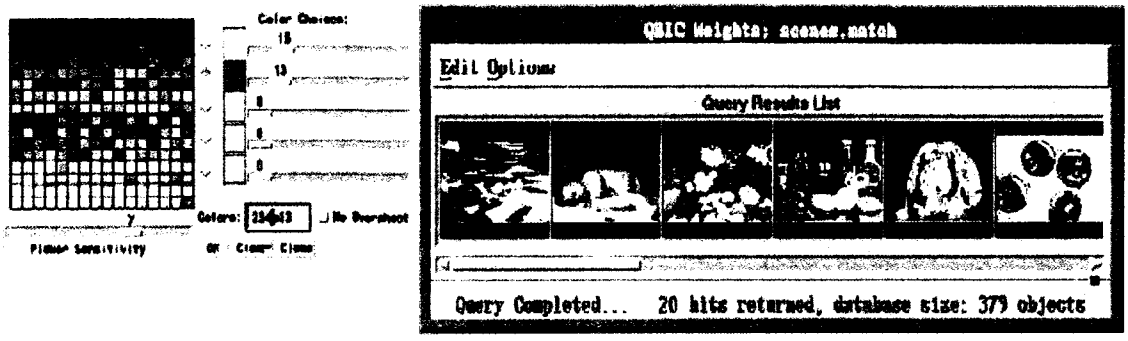


Figure 28: Garlic's Query by Color Histogram Palette

The Information Manifold. The Information Manifold is a system for browsing and querying of multiple networked information sources (Kirk, Levy et al. 1995). The system makes use of knowledge representation technology for retrieval and organization of information from disparate structured and unstructured information sources. Its second distinct feature is query optimization by accessing only the sources relevant to the query.

The Information Manifold supports multiple data source descriptions. It supports the description of complex constraints on the contents of a data source; it provides an editor for editing data sources without changing the descriptions of other sources. It reformulates queries to suite relations in the data sources. Despite its features the problem of data integration is by no means solved. The aforementioned Garlic project lists as its goals solving problem of name matching across sources when developing methods, information presentation, query optimization and execution, yet these have not been met in the Information Manifold either.

Distributed Information Search Component (DISCO). The Distributed Information Search Component (DISCO) focuses on three central research problems: transforming queries into sub-queries; the varying cost of queries due to variations in wrappers; crash

of systems when an attempt is made to access unavailable data (Tomasic , Amouroux et al. 1997).

DISCO's architecture includes two main components. The first is a mediator, which manages metadata and makes uniform the representation of data sources. The second component is a set of wrappers that execute queries on the local data sources. Interactions between the mediator and wrappers occur during data source registration and during the query-processing phase. During the registration phase, the mediator registers various wrappers; the wrapper represents the local schema in a manner that DISCO can process.

The mediator receives queries from the application during the query processing phase. It transforms the query into a plan consisting of sub-queries. Next the mediator executes the plan by issuing sub-queries against existing source-wrappers. The available wrappers process the sub-queries by communicating with the associated data sources and returning sub-answers, which are integrated and homogenized into an answer presented to the inquirer.

Tsimmis, the Object Exchange Model, and Ozone. The Stanford-IBM Manager of Multiple Information Sources (Tsimmis) main contribution was the introduction of a lightweight Object Exchange Model (OEM) that does not require strong typing and a corresponding query language whose syntax is the same as SQL (Papakonstantinou, Garcia-Molina et al. 1995). An OEM is self-describing, as it contains its own schema. It has a descriptive tag expressed in natural language, type (i.e., numeric, text, etc.) and the value. For example the set {RoomTemperature, decimal, 39.75} is an OEM tag. In Tsimmis the information extracted from a given source is converted into the OEM semi-

structured format. The authors "*believe that a self-describing object exchange model provides the flexibility needed in a heterogeneous dynamic environment*". They did not address the question how programs will "understand" the natural language tags without human intervention in the mediation process.

OZONE . OZONE was a research development effort that extended the structured Object Database Model (ODMG) (Cattell and Barry 1997) and its query language Object Query Language (OQL) with the ability to handle semi-structured data on the OEM model and the Lorel language (Lahiri, Abiteboul, & Widom, 2000). The goal was achieved by introducing XML wrappers to exiting technology, thus extending it. The research did not address a human-free mediation process.

YAT. YAT's main contribution was "a system that provides a means to build software components based on data conversion, such as wrappers or mediators, in a simple and declarative way." (Cluet and Siméon 1999; Cluet, Delobel et al. 2001). YAT provides tools for the specification and the implementation of data conversions among heterogeneous data sources using middleware, namely wrappers and a mediator. YAT offers a rule based framework and a user interface to express such mapping and transformations. The system does not support fully automated integration of data sources.

Xyleme. The Xyleme project started as an open, loosely coupled network of researchers. The Verso Group at the French national institute for research in computer science (INRIA) was at the origin of the project together with F. Bancilhon. The database groups

from Mannheim University and the CNAM-Paris as well as the IASI Team of University of Paris-Orsay were also part of the research network (Aguilera, Cluet et al. 2001).

The Xyleme system is an XML repository capable of performing various tasks such as access, store, classify, index, integrate, query and monitor on massive volumes of content and metadata. The main components of the Xyleme data model are a mediated schema and a simple query language. Once data is acquired using web crawlers, they re-visit the source website to find updates regarding the website's structure and information. If structure changes have been found then XML crawlers take note of new links and visit those URLs to acquire more information for indexing. Thus, XML indexing tables are updated and expanded by continuously adding new documents to the repository

The Xyleme project evolved to a start-up company in the year 2000, with headquarters in San Diego, California (Xyleme 2000). Xyleme was further developed to address semantic integration. However, this resulted in semi-automatically mapping of concepts and relations (Delobel, Reynaud et al. 2003).

LOREL. LOREL (Abiteboul, Quass et al. 1997) is an extension of the SQL/OQL language to query unstructured data. LOREL was designed specially to query Stanford University's prototype database Lightweight Object Repository (LORE). Being an extension query language of OQL, LOREL inherits many a features and abilities from SQL/OQL. LOREL's contribution lies in its flexibility to query without typing strict SQL/OQL commands, which is well suited to semi-structured data. LOREL has powerful

path expression capability, which allows the user to navigate and locate details even when the structure of the underlying data source is unknown.

Another noteworthy feature of LOREL is 'coercion' which allows for comparison of data rather than returning an error, as SQL would do. This ability comes handy when the data is untyped, irregularly typed or when data fields are missing. LOREL uses Object Exchange Model (OEM) graphs as part of its underlying mechanism.

STRUDEL. STRUDEL is a system for implementing data-intensive web sites, which typically integrate information from multiple data sources and have complex structures. STRUDEL allows users to manipulate the underlying data independently of where it is stored or how it is presented. STRUDEL's key idea is separating the management of a web site's data, the management of the site's structure and the visual presentation of the site's pages (INRIA 2002). STRUDEL uses a semi-structured data model with labeled directed graphs, which is a variation of the OEM data model. A STRUDEL graph is a set of nodes or objects, in which each object is either complex or atomic. A complex object is a pair {attribute, object}. Atomic objects only have basic values (e.g., integer, string, video stream). Hence, edges in data are labeled by attributes and leaves labeled with atomic values. This data model is best suited for STRUDEL because "*web sites are graphs with irregular structure and non-traditional schemas facilitating integration of multiple sources*" (Fernandez, Florescu et al. 2000).

Using STRUDEL, the site builder first creates an integrated view of the data that will be available at the site. The web site's raw data resides either in external sources or in STRUDEL's internal data repository. A set of source-specific wrappers translate the

external representation into the graph model. Second, the site builder declaratively specifies the web site's structure using a site-definition query in STRUQL, STRUDEL's query language. Third, the builder specifies the visual presentation of pages in STRUDEL's HTML-template language. The HTML generator produces HTML text for every node in the site graph from a corresponding HTML template; the result is a browsable web site (Florescu, Fernandez et al. 1998).

4.5 Ontologies Integration Approaches and Projects

4.5.1 Overview

One approach to overcoming semantic heterogeneity as a part of data integration in mediator systems is the use of metadata in the form of a vocabulary and relationships to represent domain knowledge explicitly. That is, not only structures or graphs become computer consumable, but the knowledge behind them becomes computer-consumable as well. "The holy grail is the achievement of fully automatic semantic interoperability" say Uschold and Gruninger, but they add that this "goal seems illusive" (Uschold and Gruninger 2004). Initial assumptions such that all members in a given exchange use a single language for representing ontologies, and that they all use the same ontology, are not realistic. Such assumptions must be relaxed for ontology integration to succeed (Uschold and Gruninger 2004). Numerous research projects have addressed specific aspects of the overall challenge of ontology integration.

Noy gives an overview of techniques for finding correspondences between ontologies, declarative ways of representing these correspondences, and use of these correspondences in various semantic-integration tasks for readers not very familiar with ontology research (Noy 2004). Hakimpur et al. (Hakimpour and Geppert 2005) bring the

topic of integration to a full circle – approaching the challenge of database schema semantic heterogeneity utilizing formal ontologies to represent similarity relations for the generation of a Global Schema, first introduced by (Motro and Buneman 1981) using the term *superview*.

4.5.2 Differential Ontology Editor

Bachimont et al. assert that existing ontology editors do not have complete guidelines to help users organize key components of ontologies (Bachimont, Isaac et al. 2002). As a result developers create ontologies that are very difficult to use or integrate. They propose a methodology to normalize the meaning of concepts. A key feature in their proposal is a "semantic commitment". Their approach has several steps:

1. Semantic Normalization – in which all parties involved reach an agreement about the meaning of labels used to name concepts.
2. Knowledge Formalization -- the transformation of concepts to formal primitives of reference ontology.
3. The representation of the ontology in computer-consumable form, such as DAML+OIL or OWL.

The authors build an ontology design tool named Differential Ontology Editor (DOE). It has limited capabilities because its sole purpose is to demonstrate the feasibility of implementing their idea. Subsequent ontologies built with DOE during experiments were exported to other ontology editors using RDF as the mediating syntax. The target ontology editors are PROTEGE-2000 (Noy, Sintek et al. 2000), OILED (Bechhofer, Horrocks et al. 2001), WebODE (Corcho, Fernández-López et al. 2002) and ONTOEDIT

(Sure, Erdmann et al. 2002). Earlier ontology editors such as ONTOLINGUA (Farquhar, Fikes et al. 1997), ONTOSAURUS (Swartout, Patil et al. 1997) and WEBONTO (Domingue 1998), were not involved in this Ontology transformation experiment, probably because they were considered dated or feature-poor (Troncy, Isaac et al. 2003).

4.5.3 WebODE

Most of the aforementioned ontology editors have been built as isolated independent tools that are incapable of interoperating, and are incompatible with their "rivals". Consequently, ontologies owned by different organizations, built using different tools and implemented in different languages, are difficult to exchange among ontology platforms. Corcho et al. proposed an integrated ontological engineering workbench. They have build a complex tool named WebODE (Corcho, Fernández-López et al. 2002) WebODE is an integrated ontological engineering workbench for representing, reasoning and exchanging ontologies. The tool has the capability of assisting in integration of ontologies via a user interface. WebODE supports export and import of XML ontologies, as long as their DTD schema is used. WebODE also supports direct export and import of ontologies expressed in RDF, RDFS, DAML+OIL, and OWL. However, integrating two or more ontologies must be done manually.

4.5.4 Norm Dynamics and Ontology Mapping

The Italian Ontology and Conceptual Modeling Group assumes that ontology integration is less complex if local ontologies are built with reference to other shared ontologies. They first identify terms and their semantic relationships in normative texts, such as Italy's bank regulations. The findings are transformed into local ontologies, one for each source. The next step is to compare and integrate the local ontologies into a Global

Ontology for further processing. The group calls the process Ontology Mapping (Gangemi, Pisanelli et al. 2001). Ontology Mapping is the task of finding semantic relationships between entities (i.e. concept, attribute and relation) of two or more ontologies. Schema mappings requires attributes and relations to be mapped (Doan, Madhavan et al. 2002). As with any mapping, it could be a simple one to one mapping, or complex partial mappings among sub-concepts, super-concepts, and different representations of concepts resulting in substantially dissimilar graphs and paths. Doan et al. concentrate only on one to one mapping.

(Mota and Botelho 2005) propose a translation approach from OWL representation to an ontology representation they believe is better suited for software agents, namely the O3F framework. Their approach to mapping is "bottom-up", starting with mapping of basic concepts, then using those to map more complex concepts in a recursive manner. The authors do not address how the initial mapping is achieved. Outside of these authors it appears no one has made use of the O3F framework. (Mota, Botelho et al. 2003).

4.6 Chapter 4 Summary and Implications to the Research

From CAS perspective, integration is a manifestation of a morphogenic process, where new, more complex structures, are created. It is of interest to assess the level of success such processes have achieved in the field of data integration. Progress or lack thereof could reveal longitudinal patters, and common attributes that inhibit or enable integration.

Implication of chapter to study	How Implication will be used
Standards (EDI, SWIFT, Etc.)	Assess if and how they differ from other DDLs in their ability to preserve meaning, manage entropy, and manage ambiguity
Integration of ML based DDL	Assess their ability to preserve meaning, manage entropy, and manage ambiguity
Semantic Matching Techniques and Algorithms	Assess if any of the techniques developed and used may serve this research in assessing degree of meaning preservation as proposed by SOWA

Table 5: Summary of Chapter 4 Implications to the Study

CHAPTER 5

METHODS FOR SEMANTIC HETEROGENEITY RESOLUTION

5.1 Introduction

Semantic Heterogeneity is the term used by Halevy to describe the differences in same domain database schemas developed by independent parties. The schemas will almost always be quite different from each other. (Halevy 2005). Researchers and practitioners alike have recognized the need to resolve semantic heterogeneity automatically, since human intervention precludes its application to large processing volumes. Research on semantic heterogeneity resolution dates back to 1991 (Vincent 1991) and even earlier, using different terminology, all the way to our times (Eduard, Clement et al. 2006). It is a strong indication that the challenge has not been overcome satisfactorily.

Data systems must understand each other's schema in order to cooperate. "*Without such understanding, the multitude of data sources amounts to a digital version of the Tower of Babel*" (Halevy 2005). Reconciling semantic heterogeneity is a must for data integration to succeed. The continued growth in the number of XML schemas offered in multiple domains over the years as demonstrated in section 2.5.3 strongly supports the assertion that semantic heterogeneity is not going away any time soon. The same problem manifests itself in the growing number of ontologies that are available. Ontologies pose a more complex problem: for XML there is a single syntax; yet many ontology representation syntaxes have been proposed and are used, as reviewed in section 3.2. None of the syntaxes used or proposed has been examined for its applicability to

integration using a set of necessary and required attributes. The entire development process is typified by expensive trial and error approach.

This chapter focuses on methods for resolving semantic heterogeneity. If a complete and reliable solution exists, then a major barrier to automatic data integration will have been overcome. Even if such a perfect solution does not exists, approaches to analyzing semantic heterogeneity are of interest for this research, since these aspects are important when searching for measurements of ambiguity and for the existence of meaning preservation.

5.2 Resolving Semantic Heterogeneity in Federated Databases

Hammer and McLeod presented what appears to be a seminal approach to "*accommodating semantic heterogeneity in a federation of interoperable autonomous heterogeneous databases... while at the same time honoring the autonomy of the database components that participate in the federation*". (Hammer and McLeod 1993). Their paper has been cited over 100 times, and served as the basis for integration projects such as the aforementioned TSIMIS.

Semantic resolution, according to Hammer and McLeod, is the determination of the relationships between objects that model similar information, and the detection of possible conflicts in their representations that pose problems during the unification of shared data. Their approach consists of three components. First, a function that returns meta-data about objects in a remote database. Meta-data of interest includes, type instances, value, etc. Second, a local lexicon which is a list of terms and their relations to other terms. Examples of relations are *Identical, Equal, Compatible, Kind of, Assoc, Has* and others. It is used in the fashion of <term> *RELATION* <other term>, such as <Bed &

Breakfast> *Kind of* <Hotel>. The lexicon associates real-world meaning of all shared terms in order to complement the results of the meta-functions. Data schemas are exposed or exported along with their local lexicon. The third component is a semantic dictionary, created and maintained by a "sharing advisor", a software tool that obtains all data pertinent to a new incoming concept deemed necessary by the software's designer. The authors briefly explain that any component that wishes to participate in the data sharing must first undergo a registration process in which the sharing advisor participates. The authors conclude that "*it is not at present possible to completely automate the tasks of the sharing advisor, particularly in supporting new components. In consequence, in practice one or more humans will likely be required to assist*" (Hammer and McLeod 1993)

5.3 Word-based Heterogeneous Information Representation Language

Word-based Heterogeneous Information Representation Language (WHIRL) is a quasi-database management system which supports similarity joins based on text similarity metric (Cohen 2000). WHIRL addresses the problem of combining information from relations that lack common formal object identifiers. WHIRL's novelty is that it combines logic-based and text-based representation. WHIRL gives a numerical value of weight to each term it encounters, as well as weights to indicate co-references, such as in "Wall-Mart" and "Sam Walton". WHIRL retains the reasoning for the similarity of pairs of names, using statistical measures of document similarity that have been developed in the information retrieval community. WHIRL suffers from a known string-join problem: given two large sets of strings and a predefined string-distance threshold SD, finding all pairs from the two whose distance is no greater than *SD* is resource intensive and yields poor results. That is, pairs related by the system are not necessarily related in the world. It

is worth noting that the same problem arises with data mining applications, data cleansing tasks, and in data integration tasks as seen here.

5.4 Similarity Relations

Similarity relations are generalizations of equivalence relation (Zadeh and Desoer 1963). Rather than relying on schema element names, or relying on the structure of schemas, (Hakimpour and Geppert 2005) utilize formal ontologies that contain definitions of terms. Similarity relations are discovered by a reasoning system using a higher-level ontology. Relations are then used to derive a Global Schema. The justification for this approach is cost reduction for the generation or regeneration of global schemas in a tightly coupled federated databases environment. The approach seems to be limited because such databases might have a significantly reduced level of "Semantic Heterogeneity" (Halevy 2005).

Williams et al. attempt to create "local consensus ontologies" as they term their integrated Global Schema, via means of similarities based on a syntactic similarity and semantic equivalence (Williams, Padmanabhan et al. 2005). Their approach is similar to lightweight ontologies (see section 3.2.2 for example) in that they break the problem into small chunks. Experiments using their algorithm show that as the number of input ontologies increases, the number of concepts that are correctly mapped decreases.

5.5 Integrated Multi-Match Strategies

A tool named COMA++ utilizes a composite approach to combine different match algorithms, uniformly supporting XML schemas and OWL ontologies. According to its authors, COMA++ can be used "*also to comparatively evaluate the effectiveness of*

different match algorithms and strategies" (Aumueller, Do et al. 2005). COMA++ matches two schemas or ontologies using a central vocabulary to which and from which all mappings occur. They term it Mediator Taxonomy. The mediator taxonomy links between the source and target of the matching task. The tool uses similarity matching strategy, aided by structure analysis to arrive to the "best" match. Figure 29 (taken from the original article) illustrates the usage of a mediator taxonomy for beer. It appears that the main shortcoming of the tool is its dependency on pre-existing mediation taxonomies for ontology integration. Further, the tool heavily depends on user involvement.

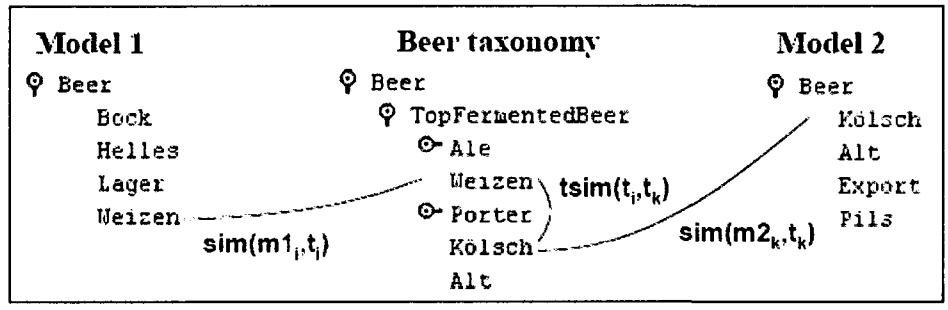


Figure 29: COMA++ Taxonomy Mapping

The reader probably recognizes this approach as a reincarnation of the "advisor sharing" tool proposed by Hammer and McLeod more than a decade earlier.

5.6 Chapter 5 Summary and Implications to the Research

This chapter provided an overview of early and contemporary methods for resolving semantic heterogeneity. All approaches attempt to reason about the meaning and resemblance of heterogeneous objects in terms of either their linguistic or structural representation or a combination of the two. All approaches succumb to the need for a human to provide semantic resolution services.

From a CAS perspective, resolving semantic ambiguity is in essence the creation of a meaning preserving mapping, or *relation,* between two terms (or objects) such that their

meaning is preserved. Consequently, ambiguity, and therefore *entropy*, are reduced or even eliminated. Relations that last for the duration that they are needed are considered to have sufficient *tension* to hold them together.

Implication of chapter to study	How Implication will be used
Semantic Heterogeneity	The research evaluates if, and to what degree, new proposed DDL have solved the challenge of semantic heterogeneity, as it is clear that all tools reviewed are merely assisting humans in resolving semantics.
Resolution Strategies	Resolution strategies can be viewed as attempts to build cybernetic regulators. Failure to resolve all ambiguities indicates they do not satisfy the law of requisite variety. These aspects are explained and discussed in chapters six and thirteen.

Table 6: Summary of Chapter 5 Implications to the Study

CHAPTER 6

COMPLEX ADAPTIVE SYSTEMS

6.1 Introduction

This chapter gives an overview of Complex Adaptive Systems (CAS) theory, introduces CAS terminology and relates them to some aspects and constructs of information systems. CAS terminology of interest includes Closed Systems, Open systems, Morphogenesis, Relations, Variety, Tension, and Entropy.

The chapter begins with a short introduction to complex systems, and a brief review of Cybernetics, a significant component of CAS. The next section reviews CAS in a nutshell, followed by a review of paradigm changes in computing, explaining the changes through CAS prism as morphogenic processes. The next section makes a connection between CAS and data integration. The chapter then relates CAS variety to DDL, tension and DDL and finally entropy and DDL.

6.2 Brief Introduction to Complex Systems

Complex systems occur naturally. For example, atoms form molecules that in turn may form living organisms. Families, Government, the Weather, are all examples of complex systems. The constituents of a complex system could be simple, but once they interact or their existence is intertwined, they become complex. It is not merely the number of parts that make a system complex – it is the relations and interactions among its parts. The field of study of complex systems holds that the dynamics of complex systems are founded on universal principles that may be used to describe disparate problems. (Bar-Yam 1997)

Complex systems have several properties in common (Bar-Yam 1997; Hannon and Ruth 1997; Polderman and Willems 1998):

- An environment in which the complex system exists
- Constituent elements. These are usually finite and countable.
- Interaction among elements and with the environment. The interaction may vary in strength.
- Variety (number of distinguishable states or elements)
- Operation (could happen in different time scales)
- One or more activity having some goal.

Parts of complex systems could be complex themselves, although it is not a necessary condition. Case in point is a molecule of water formed from two atoms. The qualities and characteristics of a water molecule are not to be found in any of its simpler parts. Further, a single molecule of water does not suffice to exhibit (nor to study) the behavior of a body of water. It is the collective behavior of water molecules that is complex. This is termed *Emergent Complexity* (Bar-Yam 1997) – the complex behavior that emerges from the interaction of a system's simple parts.

To study complex systems, one cannot concentrate on the system's parts alone. They need to be studied in the context in which they exist; this must include their relations and interactions. Emergent complexity can be studied by examining the parts in the context of the system as a whole (Bar-Yam 1997; Hannon and Ruth 1997).

Complexity can be measured quantitatively. A system's complexity is "the amount of information needed in order to describe it." (Bar-Yam, 1997 page 12). "Information" in this context is explained in detail in chapter 9 (Information Theory). Therefore, we

shall briefly set the stage for the discussion to come. The level of detail in complex systems is scale dependent (Frame and Mandelbrot 2003). Once the scale is determined, the system could have many possible differentiable and countable states (for that scale). To specify in which state the system is in, one can point to the state's number. The number of binary digits (bits) required to specify the number of states is related to the number of states. Suppose that the system has Ω states. The number of bits of information (I) needed to describe all its states is $I = \log_2(\Omega)$. The representation of each state requires as many numbers as there are states. For a string of N bits there are 2^N possible states. Therefore $\Omega = 2^N$. From here we can see that N and I are the same. Meaning, the number of states and the information of the system (its complexity) are one and the same. This is an important characteristic that should be remembered when reading the next section, as well as when reaching the methodology chapter and the outcome of DDL measurements chapter.

6.3 Brief Review of Cybernetics

CAS was preceded by cybernetics, which in turn emerged out of partially overlapping research carried out by independent researchers such as Ross Ashby and Norbert Wiener, two well-known names in the field. Ross Ashby is attributed by some to have written the first modern paper on cybernetics, before it was termed as such by Wiener. (Macrae 1951; Pickering 2002)

Ashby's 1940 paper explores several fundamental concepts that were ported to CAS. *Adaptation* is one of them. Adaptation to the environment, asserts Ashby, is equivalent to "the behavior of a system in equilibrium" (Ashby 1940; Ashby 1947). Equilibrium or a *steady state* means an object (machine, system, etc.) maintains its functionality without

breaking up. An object of interest may respond to variety (the number of distinguishable states) through some regulation mechanism. The ability to reach a steady state depends on the amount of external factors the regulation mechanism can respond to.

An adaptive system is one that changes its behavior in response to changes in its environment. The adaptive change that occurs is often relevant to achieving a goal or objective.

6.3.1 The Law of Requisite Variety

The variety of a given system is the number of meaningfully different states and disturbances that a system has. Variety, in relation to a set of distinguishable elements, is used to mean either (i) the number of distinct elements, or (ii) the logarithm to the base 2 of the number of distinct elements, which in essence is a bit (Ashby 1956, p. 126). For example, the variety of a card deck having 52 cards is $\text{Log}_2 57 = 5.7$ bits. The variety of a data structure that has 16 data elements is $\text{Log}_2 16 = 4$ bits.

Ashby gives the following example for the law of requisite variety: "*If, for instance, a press photographer would deal with twenty subjects that are (for exposure and distance) distinct, then his camera must obviously be capable of at least twenty distinct settings if all the negatives are to be brought to a uniform density and sharpness*" (Ashby 1956).

The variety of a system's regulator is the number of meaningfully different responses the regulator can respond with, vis-à-vis the environment's variety. Therefore, the number of distinct inputs that a system's regulator can apply to the system during the course of its operation determines the extent to which the system can react and adapt to its changing environment. For example, a voltage regulator maintains a steady level of

voltage in an electrical line by eliminating power surges and spikes harmful to sensitive electronic devices. Casti expounds on the notion of a regulator: "*Ashby showed that the ability of any regulator to control a given system is severely constrained by a control-theoretic version of the second law of thermodynamics, the so-called Law of Requisite Variety*" (Casti 1985). For a regulator to transmit into the system all its variety, the communications channel between the systems and the regulator must be capable of transmitting all that variety.

Standards such as EDI provide a regulation mechanism of some sort. EDI enabled IS are designed to process any EDI compliant message that is provided by a trading partner ("the environment"). The EDI processor (e.g., "regulator") can reject some or all messages and accept others for further processing. That is, an EDI engine also provides a selection mechanism driven by goal oriented business rules. Every data item in a system that originated in EDI can be traced back to EDI. The same applies for SWIFT. However, the variety standards such as EDI and SWIFT offer is very constrained. Figure 35 illustrates what happens (or doesn't happen) when there is no regulator for data integration: Schema-2 cannot adapt to what Schema-1 offers. Computerized ontologies attempt to provide a means to build regulators that support much more variety compared to standards. Computerized ontologies can be viewed as anchorage points supporting suspended bridges. Rather than tensioned cables attached to each side of an anchorage, there are data elements mapped to the ontology. Meaningful mapping creates the tension for the links ("cables") to maintain the relationship between a data structure and the ontology (anchorage point). The problem with this approach is analogous to the problem the Tacoma Narrows bridge had: ontologies are unstable to the point they can easily

break mappings, just as is the case with all other DDLs. Section 3.2 reviews some of the instability characteristics of ontologies as anchor points, along with illustrations (Figure 18, 6, 7). Section 6.6.1 of this chapter further develops the connection between DDL and Variety.

(Casti 1985) proves a theorem that states "*The variety in a system's input equals the variety in its output if and only if the system is completely reachable and completely observable*" (e.g., if the system is canonical). Therefore, if the system is not completely reachable and completely observable it is not possible for a regulator to fully regulate it. To understand the statement the reader needs to have an agreed upon understanding of *controllability, reachability* and *observability.* Controllability concerns whether a given initial state X_0 can be directed back to the origin in finite time using the input u(t). A state X_0 is said to be controllable if there exists a finite time interval [0, T] and an input {u(t), $t \in [0, T]$} such that x(T) = 0. If each and every state of a system is controllable, then the system is said to be completely controllable. A state X' = 0 is said to be *reachable* from the origin if, given X(0) = 0, there exist a finite time interval [0, T] and an input {u(t), $t \in [0, T]$} such that X(t) = X'. If each and every state of a system is reachable, the system is said to be completely reachable. If all outputs from a given derivation rule in a system are independent, then the system is said to be completely *observable.* Casti explains: "*Just like its physics counterpart, the second law of thermodynamics, the law of requisite variety imposes an upper bound on the information that can be transmitted from a sender (the input) to a receiver (the output), with the maximum transfer being achieved by a canonical system*" (Casti 1985). Chapter 13 (discussion) will make further use of Casti's theorem and explanation.

Standards such as EDI and SWIFT are canonical for each of their business domains. They are completely reachable and completely observable. Therefore, they provide for maximum variety transfer. In contrast, all other data structures created by data modelers and even ontologies (serving as pseudo-standards) are not canonical. This point is further explored in chapter 13 (discussion).

Constraint is a key concept associated with variety. When the variety that exists under one condition is less than the variety that exists under another in a related set, the smaller variety is said to be constrained. (Ashby 1956 p.127). For example, a traffic light has Red, Yellow and Green lamps. Each can be turned on or off, independently of each other. The total number of combinations the set has is $2^3 = 8$. Traffic engineers in Europe decided to use only four combinations: one allows the red and yellow to be on at the same time, and the remaining three allow only one light to be on at a time. American traffic engineers constrain the possible variety to only three possibilities – one lamp only turned on at a time. In both cases the number of used combinations is less then the possible eight, therefore constraint is manifested.

EDI, SWIFT and similar standards were devised as sociotechnical constraint. First, each such standard limits the infinite expressive variety data modelers and systems developers have at their disposal. Second, each standard limits the scope of concepts and objects that can be exchanged. It is analogous to limiting the carrying capacity of a communication channel. It is also analogous to a noise reduction filter. In contrast, other DDLs do not impose any such constraints. Computerized ontologies have been suggested as a constraining mechanism. However, the proliferation of ontologies is uncontrolled

and exhibits the same growth pattern as XML when it was considered the panacea for automatic data exchange in the late 1990's.

Degrees of Freedom is closely associated with the concept of constraint. It is the measure of the number of independent components (or elements in a set or in a vector) of the range that would give the same variety as the constrained variety. Using the traffic light example, two components suffice to give the same variety of four states as the constrained variety generated by the three components: there is one choice of three for the first light, one choice out of two for the second light, and no choice for the third light, yielding two degrees of freedom.

6.3.2 Homomorphism and Isomorphism Machines

Any set with one or more rules for combining its elements is an algebraic structure (Pinter 1982). Consider several alcohol beverage bottles in a wedding party, where the content of any two bottles is used by a bartender to mix a drink. The mix operation produces a new drink. This may be conceived as an algebraic structure. This algebraic structure has several mathematical characteristics. For example, this algebraic structure is commutative: mixing from the first then the second bottle is the same as mixing from the second then the first. Here's an additional example: consider two members of an expanded family attending a wedding party who are asked to pick the closest genealogical relative to both of them. Here too we have a set and an operation, therefore this may constitute an algebraic structure as well. It is important to note that these two algebraic structures are different in substance (beverage, people) but identical in form.

The set of guests in the example above can be made to coincide with the set of bottles if we were to superimpose the first person with the first bottle, the second person

with the second bottle, and so on, as pictorially illustrated in Figure 30. The one-to-one correspondence which algebraically transforms people into bottles (or from a group named A to a group named B) is a structure preserving mapping (or function) and it is called a *monomorphism.* A monomorphism that in addition has a one-to-one correspondence from group B to group A is called an *isomorphism*, as depicted in Figure 31 (Colton and Duffee 1940; Pinter 1982).

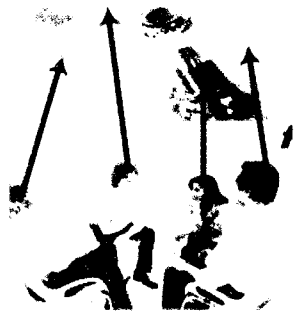


Figure 30: monomorphism function

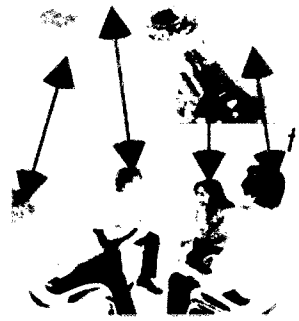


Figure 31: isomorphism functon

If G and H are some arbitrary groups, it may happen that there is a function which transforms G into H without having a one-to-one correspondence. For example, let G be the additive group of all integers modulo 12, all inclusive (customarily denoted as $\mathbf{Z}_{12}$). Let Modulo 5 be a transforming function. (Modulo 5 always yields the remainder of the division of a number by 5). This transformation maps $\mathbf{Z}_{12}$ onto $\mathbf{Z}_5$. Figure 32 lists the remainder next to the number that was divided by 5.

Number	Mod(5)
0	0
1	1
2	2
3	3
4	4
5	0
6	1
7	2
8	3
9	4
10	0
11	1

Figure 32: Modulus 5

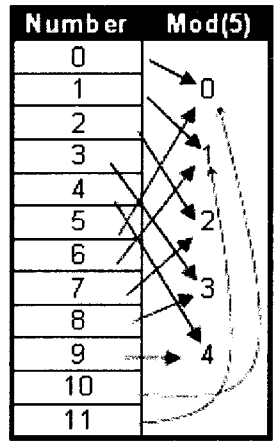


Figure 33: Modulus 5 condensed

The observant reader can see that the numbers under the column entitled *Mod(5)* repeat. The same function depicted in condensed form is shown in Figure 33. The function Modulus 5 maps G onto H but there is no one-to-one correspondence. This type of mapping is termed simply a *homomorphism*, or sometimes an *epimorphism* because it is onto. An homomorphism is also a structure preserving mapping from one set to another. However, it is not bijective. Meaning, one cannot map back from H to G and preserve the structure (Colton and Duffee 1940; Pinter 1982). Homomorphisms and isomorphisms are of particular interest for this discussion.

The study of some phenomena can be done by direct observation and direct measurement. Some studies require indirect observation and measurements, but these should be at least homomorphism to the original, and isomorphic at best. In other words, an observer can see the input into a machine, and can see the output from the machine, yet cannot see the machine's details. Based on the observation, the observer can hypothesize on the manner in which the machine operates. Ashby terms such a machine "black box".

Ashby discusses in his 1956 book the black box machine, which is "*a sealed box that has terminals for input... and terminals for output, from which [one] may observe what [one] can*" and deduce what one can of its contents. "*Though the problem arose in purely electrical form, its range of application is far wider*" (Ashby 1956). The study of a black

box gives the experimenter information up to a certain amount and no more – up to a canonical representation that identifies the mechanism "up to an isomorphism".

> *" 'Isomorphic means, roughly, "similar in pattern". It is a concept of the widest range and of the utmost importance to all who would treat accurately of matters in which "pattern" plays a part. Let us consider first a few examples merely to illustrate the basic ideas. A photographic negative and the print from it are, so far as the pattern of the picture is concerned, isomorphic. Squares in the negative appear as squares in the print; circles appear as circles; parallel lines in the one stay as parallel lines in the other. Thus certain relations between the parts within the negative appear as the same relations in the print, though the appearances so far as brightness is concerned are different, exactly opposite in fact. Thus the operation of changing from negative to print leaves these relations unaltered.* (Ashby 1956, page 94)

One can view a computer application as a black box, whose terminals are its exposed data structures, expressed in any DDL. Further, one can view any computerized data integration process, which is a specialized computer application, as a black box. Its goal is to be the regulator between a system and the varied environment it is attempting to integrate with through some other data structure. A successful integration yields some homomorphism between a computer application's exposed data structure and some external data source. An ideal regulator (or mediator, in computer science terminology) would create some isomorphism. This is demonstrated in Figure 34: a mediator should be able to create an isomorphic mapping between the object entitled "Dog1" and the object entitled "Car". The object "Dog2", which went through a variety-reducing transformation form a "complete dog" to something less than that, is merely homomorphic to the object

"Dog1". Therefore, no mediator would be able to create a complete isomorphic relation between "dog1" and "dog2".

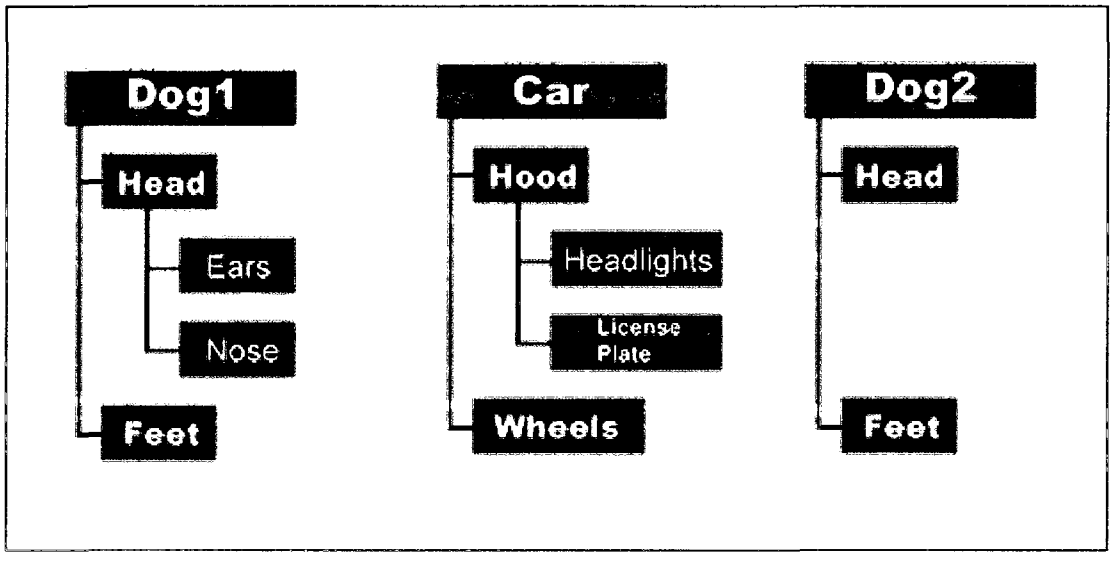


Figure 34: Isomorphism and Homomorphism Examples

The utility of an isomorphism with a black box machine is often used as a research methodology when attributes or concepts cannot be measured directly, but rather by an isomorphic proxy.

6.3.3 Behavior and Purpose

Rosenblueth, Wiener and Bigelow published "Behavior, Purpose and Teleology" in January of 1943 with two goals in mind: the first was to "*define the behavioristic study of natural events and to classify behavior. The second is to stress the importance of the concept of purpose*" (Rosenblueth, Wiener et al. 1943). The paper defines output as "*any change produced in the surroundings by the object" and defines input "as any event external to the object that modifies the object in any manner*". It then defines the study of the object's behavior as the examination of the output of an object of interest, and the relations of the output to the object's input.

Rosenblueth et al. assert that a purposive object must be connected to some features of the environment, and must be "oriented to and guided by" the goal. Purpose testing is made by changing the environmental conditions. An object behaves purposefully if it continues to pursue the same goal by changing its behavior as conditions change. Wiener's cybernetics stemmed from the study of governors and aiming devices mainly focused on damping disturbances as a means of controlling relatively closed and centric apparatus. Churchman and Ackoff comment that cybernetics "*makes in a new way a point which has already been made in various guises in contemporary science and philosophy*" and add that "that teleological concepts are extremely fruitful in the study of neurological and machine behavior, and that such concepts can be treated experimentally." (Churchman and Ackoff 1950).

Churchman and Ackoff argue "that the social sciences require a more refined analysis of function and purpose than one which suits cybernetic purposes" (Churchman and Ackoff 1950) and propose a refinement: (a) the object and the environment should not be rigidly specified as some physical objects; (b) a purposive object displays a selection process among choices present to it; (c) purpose can be only studied over time; and (d) a purposive object needs to, potentially at least, produce some end-results. The treatment of open adaptive distributed control systems became centric in a development that stemmed from cybernetics, namely CAS, as the next section describes.

6.4 Complex Adaptive Systems Theory in a Nutshell

6.4.1 From Closed to Open Systems

The transition from closed to open system is closely related to the transition from mechanical systems to adaptive, information processing, systems. An open system

doesn't simply engage in interchange (of information in my case) with its environment; this interchange is an essential factor underlying the system's viability, continuity, and its ability to change further. "Information is inherently relational, a mapping between two or more sets of objects or events" (Buckley 1998, page 185).

Interaction takes place only when two objects act on each other – either one sided or reciprocally. Information Systems (IS) exists within an "environment". If the IS does not interact with the environment it is said to be "closed". If the IS interacts with the environment it is said to be open. The degree of openness is related to the amount of interaction. Openness is also subject to an observer's viewpoint or bias, as brilliantly explained by Abe Mowshwitz in his article describing approaches to the study of social issues in computing (Mowshowitz 1981).

Interaction of the parts comprising a mechanical system is expressed in the physical concept of energy. CAS energy is manifested at much higher levels, such as stress in animals, or psychic energy in humans. Closed systems break down or reach new equilibrium when facing an intrusion from the environment. In contrast, open systems typically react to obtrusions by changing some of their structure to a more complex level, capable of adjustment to new relationships. The interrelations characterizing higher levels progressively depend more on the exchange of information. The carrier of the information is insignificant (e.g., XML, Cobol); **it is the mapping of the structured variety to some other repertoire of meaningful structure that is important.**

Extending the closed (technical) systems viewpoint of Information Systems (IS) with a theory of organizations as open, adaptable systems is the goal and function of sociotechnical theories. The sociotechnical view addresses the indispensable interactions

of the technological core of the information system with the environment, such as users, organization, legislation, and so forth. According to these theories, IS are open sociotechnical systems. IS processes involve technology that transforms raw materials into output and a social structure that links the human operators with both the technology and each other. Sociotechnical theories combine these subsystems with the IS development and use processes so that technical and social goals of organizational adaptability and survival are achieved and reinforced (Lyytinen 1987). Of all the DDLs reviewed in this research, computerized ontologies appear to be a component that increases complexity, because it requires the creation of more mappings than what is necessary when two data structures map to each other. The reason is simple: one data structure maps to the ontology, and also the other data structure maps to the ontology, which acts as an intermediary. Unfortunately, such an increase in the number of relations is not a morphogenic process. Ontologies do not provide or use feedback to adjust the mapping if any of the three components changes (e.g., two data structures and the ontology). Ontologies are not self-regulating and not self-organizing, just as Cobol data structures, XML or the EDI standard are not.

Contemporary systems theories focus on the principles of organization (arrangement) and its change. Morphogenic tendency of processes to elaborate or change systems' form, structure, or state are of particular interest. *Morphogenic systems* increase complexity and decrease their local disorder; they are open systems with feedback loops; they are goal seeking, self-regulating, self-directing, and self-organizing. They are made of spatially, temporally, or causally related ensemble or set of elements, states, or events.

The theory of CAS put forth by Buckley "is a sophisticated modern systems theory that cleverly integrates Cybernetics, General Systems Theory, and Sociology" (Burrell and Morgan 1979). Buckley builds on the assertion that general systems theory helps identify varied relationships in sociocultural systems; it emphasizes processes of information and communication; it is integrative; and it views the world in dynamic terms. Buckley reviews historical approaches to social systems, namely mechanical and organic, points to their shortcomings, and proposes a new approach that extends general systems theory – the sociocultural system. The three theories differ in the way they work and in their degree of complexity and instability. Buckley demonstrates that systems can be described in terms of the degree to which they are open or closed. Closed systems are entropic in nature, thus they tend to break down. Open systems are negentropic[4]. That is, open systems tend to elaborate structures and tend to respond to a greater range of vacillation in the environment than closed systems do. Sociocultural systems are purposive and goal-seeking due to their capacity to receive feedback from their environments. Feedback, a key concept in Cybernetic and in General Systems Theory, allows analysts to take into account change, growth, friction, and evolution in their studies of social systems. Moreover, systems theorists emphasize the importance of a system's evolution of structure during its development. This internal process is termed *morphogenesis*. Of no less importance is *morphostasis*, a system's process of retaining its structure by filtering external forces and admitting only those changes that do not threaten the system's existing structures.

---

[4] The physicist and Nobel laureate Schrödinger introduced the concept when explaining that a living system exports entropy in order to maintain its own entropy at a low level. Of anecdotal interest is his dissertation: *On the conduction of* ***electricity*** *on the surface of insulators in moist air*

6.4.2 Tension in CAS

According to Buckley, tension in physical systems can be expressed as interaction of parts in a mechanical system, measurable in some units of energy. Suspension bridges provide a prime example of tension. The tension on their cables is supposed to preserve the relation between the state of the bridge and some aspects of its environment. Compared to the Golden Gate Bridge, the Tacoma Narrows Bridge that collapsed in a wind of 42 mph, demonstrates what happens when tension fails. The mechanical tension of the bridge's cables gave way to changes in the environment in a manner that was not designed for. Vines provide a natural example of a cable-like mechanism that seeks support in the environment. A vine would interact with some elements in the environment and not with others. For example, it may attach itself to a tree or a fence, but not to a spider's web. Humans can use "social vines", by participating in a support group, for example. One may view data integration between autonomous heterogeneous sources as a form of "computerized vine" seeking a matching counterpart for the purpose of importing, exporting or exchanging data.

Raw energy in CAS is replaced by "more complex form of organized and directed motive force" (Buckley 1967 page 55). A CAS whose internal organization responds to some specific parts of its surrounding variety, while ignoring other parts of the available variety, the CAS has in fact mapped part of the available variety and constraints from the system's surroundings into its internal structure. This mapping is isomorphic, at least in some aspects, to the original variety. The mapping occurs, as a minimum, on a semi-permanent basis. **The degree of sensitivity towards its environment is *tension*; it preserves, at least temporarily, the relation between the inner state of the CAS and**

some aspects of its environment. The mapping corresponds "closely with the current conception of 'information', viewed as the process of selection of a variety [which] has meaning" (Buckley 1967 page 64). The connection between DDL and tension is further developed in section 6.6.2 in this chapter.

6.5 Paradigm Change in the Computing Industry

The computing industry's initial paradigm was centralistic and monolithic. That is, organizations that computerized some of their business processes, such as accounting or manufacturing, had a single computer that executed programs from a central location. Organizations modified their interaction methods by exchanging data electronically among their computers. Typically, the structure of the data to be exchanged was agreed upon by the businesses, and then data was exchanged by physically exchanging computer tapes (Rohn 1982). Electronic Data Interchange (EDI) pioneered non-physical exchange of data through Value Added Networks (VAN) that acted as switchboards (Unitt and Jones 1999).

When computers and networks became pervasive and thus less expensive, the paradigm changed to that of networked computing. The number of interacting components grew astronomically, compared to the days of the mainframe. Ubiquitous exchange of data directly between business units and organizations became the norm. The dependency upon humans mapping data structures that facilitates data integration has become a costly bottleneck. One of the many examples brought in Chapter 1 is the financial consequences of imperfect interoperability costing the U.S. automotive supply chain at least $1 billion per year (Rohn and Klashner 2004). Automation of the data integration process has become an acute issue for businesses and other organizations.

Chapter 4 demonstrated that in spite of serious and costly efforts to automate the data integration process, this goal has not been achieved or even come close to. This lack of real progress in automatic integration over thirty years of research efforts by academia and industry indicates that there might be an invisible "brick wall" that is a sociotechnical phenomenon. At the same timeframe other areas of computing continued their rapid growth. The growth of the entire computing industry—a holistic system—is hampered by the lack of progress in data integration research.

Computer systems are made of ensembles or components with a distinct boundary that evolve as systems interact with each other via input, output and feedback. Computer systems have subsystems that are mutually interdependent. (e.g., Accounting: General Ledger, Accounts Receivable, Accounts Payable); they influence their social and technical environments, and influenced by them. In addition, they have observable behavior within a sociotechnical paradigm. These are the same characteristics that typify CAS (Buckley 1967). This particular framework allows us to discuss information systems components in a philosophically grounded way, a novel aspect that is rarely found, if at all, in the technical discussions about data integration.

6.6 Data Integration and CAS Properties

Data integration, whether done entirely by humans, or semi-automatically, or by an entirely automatic process, combines two or more non-identical structures as a means for attaining a goal – some form of data exchange.

If the combined structure is the lowest common denominator, the result is a homomorphism to the more complex objects. The smaller structure has a smaller variety and has no regulator that enables it to interact with the greater variety of the other data

structure. This yields a system that is relatively closed and entropic, where the richness of the larger structure is lost.

The results of an unsuccessful integration attempt are characteristic of a homeostatic system - maintaining its given structure within pre-established limits. Figure 35 illustrates a situation of two heterogeneous schemas describing shoes. Schema-1 has more variety than Schema-2, as it has a larger number of data elements. The three thin dotted lines map identical element names. The thicker lines map identical concepts, expressed differently. It is worth noting that neither data structure provides for a mapping mechanism. Regardless of how the mapping is achieved, it is clear that Schema-2 has no means to incorporate variety from Schema-1 that it doesn't have in its own already. For example, colors in Schema-1 are expressed as individual data elements (<black/> <white/> <blue/>), but as attributes of a single data element in Schema-2 (<available-colors>). Conceptually (at least from a shoe's view point) the two are one and the same. Schema-2 is homomorphic to Schema-1. In addition, Schema-2 is entropic because it failed to incorporate anything new from Schema-1. The mapping achieved in Figure 35 creates tension between the two schemas, but it merely maintains Schema-2's structure, rather than evolve it.

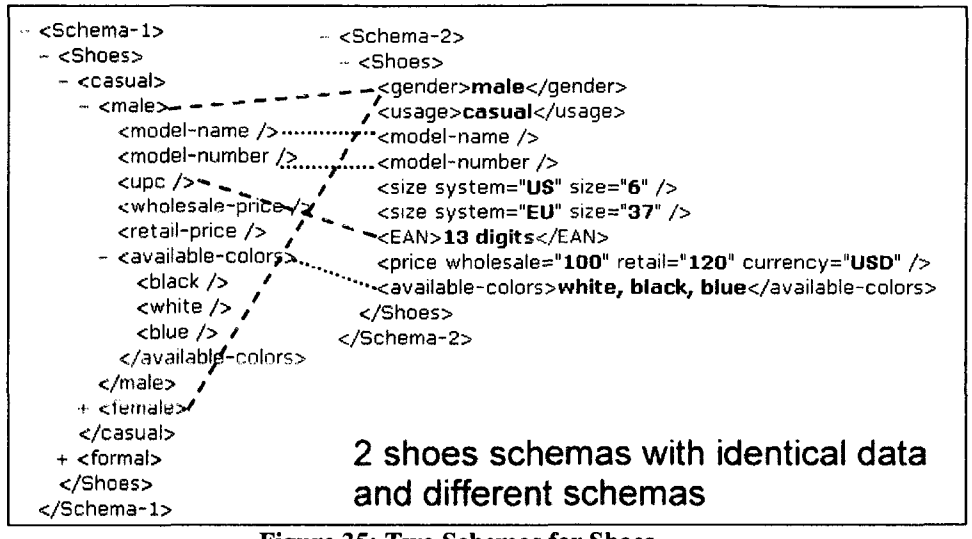


Figure 35: Two Schemas for Shoes

Successful data integration creates more complex structures due to its ability to adapt by responding to variety from the environment. The magnitude of the adaptation is related to how much "sense" the variety makes. The more meaningful it is (in other words, the less noisy it is), the more complex the structure and the system it serves become. Morphogenic systems evolve, and in the process create more complex structures. New structures are more than the sum of their parts. The growing structure is maintained by tension; without it there will be no retention of new concepts.

6.6.1 Variety in DDL

Within the CAS framework the environment is seen as a set of components or ensembles having distinguishable elements, states and events, external to the observed system. These can be discriminated in terms of temporal, causal and spatial properties or relations. Such differences are generally referred to as *variety* - a fundamental concept in CAS.

Data Modelers create data structures using natural language, except when the DDL in use constrains naming conventions to very short strings, as was the case with the Basic

programming language. The creation of any set of signifiers and signified constitutes a language, even if none of the signifiers can be found in a dictionary such as Merriam-Webster or the Oxford dictionaries, including 2-letter data names, as was the case with Basic. The thousands of data structures that are publicly available, or made available by special agreements, or even accessed by means of deceit, all comprise the variety that exists in the environment. Linguistic theories and tools, such as ambiguity classification (Gardent and Webber 2001) and Zipf's distribution of words and meanings (Zipf 1949) are used for gauging the nature and amount of variety that exists in data structures constructed with different DDLs.

6.6.2 Tension among DDL Constructs and Meaning Preservation

Successful mapping between a system's data structure, expressed in any DDL, to an autonomous and heterogeneous data structure that exists in the environment, is a selection process where the external data structure represents a constrained variety that exists in the environment. The mapping needs to "make sense", in other words - preserve the meaning of the external variety vis-à-vis the internal structure of the system whose goal is the integration. The constrained variety mapped between such two ensembles is communicated through some mechanism "of coding and decoding such that the original variety and its constraints remain relatively invariant at the receiving end." (Buckley 1967 page 64). This includes "symbolically mediated sociocultural systems, and overshadows the mapping of the physical environment" (Buckley 1967 page 64).

Therefore, one may safely conclude that a successful mapping between two data structures expressed in any given DDL will produce tension. The reverse is also true – if tension is measured, say, via means of meaning preservation, one can conclude that a

successful mapping was achieved. Absence of meaning preservation means there is no tension, which leads to the conclusion that no stable, although perhaps temporary, mapping has been achieved. This research later adopts Sowa's approach to establishing the existence of meaning preservation (Sowa 2001). This research inquires if contemporary DDLs designed with data integration in mind provide the means necessary to create and maintain tension.

6.6.3 DDL and Entropy as a CAS Property

The physicist and Nobel laureate Erwin Schrödinger uses entropy in the context of thermodynamics to assert that living organisms maintain their organization (structure) as follows:

> *It feeds upon negative entropy, attracting, as it were, a stream of negative entropy upon itself, to compensate the entropy increase it produces by living and thus to maintain itself on a stationary and fairly low entropy level. If D is a measure of disorder, its reciprocal, l/D, can be regarded as a direct measure of order. Since the logarithm of l/D is just minus the logarithm of D, we can write Boltzmann's equation thus: -(entropy) = k log (l/D). ...thus the device by which an organism maintains itself stationary at a fairly high level of he orderliness really consists continually sucking orderliness from its environment* (Schrodinger 1944).

Von Bertalanffy demonstrated the physical equivalence of thermodynamics entropy and information theory entropy (Raymond 1950).

Buckley quotes George A. Miller's words (Miller 1953) to clarify the relation between entropy and order, as follows: "A well organized system is predictable – you know almost what it is going to happen before it happens. A perfectly organized system is completely predictable and its behavior provides no information" that you did not

already have about the system. Organization (e.g., arrangement) "can be visualized as a number of elements, where each element has its own set of alternative interactions with other elements. Each element has some freedom of choice of interaction, but also some constraints" (Buckley 1967, p. 87). A *complexion* is any specific set of choices out of all the possible sets, made by each element. The number of complexions in an arrangement is the number of possible alternatives one can choose from. This is equivalent to ensemble of variety in Shannon's information theory (Shannon 1948), and the entropy measure "H" thus can be used. If the elements were entirely independent and had no interaction constraints then all combinations have equal probability, resulting in maximal entropy and zero organization. On the other hand, if the constraints are such that only one set of complexions is allowed, then there is zero entropy and maximum organization. However, in dynamic organizations such as CAS, the set of complexions is not limited to one but is significantly less than the maximum entropy. Specifically, sociocultural systems are "a set of elements linked by way of the intercommunication of information" (Buckley 1967). While in lower level systems information is energy, in higher level systems information "is a relationship between sets or ensembles of structured variety" (Buckley 1967). Organization measures the amount of constraint introduced to a collective. It is possible to represent two ensembles of complexions (collectives, groups, societies, data structures) by x and y. The entropy H(x) represents the organization of x and similarly H(y) represents the organization of y. The amount of common or compatible organizational interaction is represented by "T", as shown in Figure 36 below. Sociocultural systems that differ substantially would be expected to have a narrow

common area, whereas sociocultural systems with very similar organization would be expected to have a very large overlap, or a large "T".

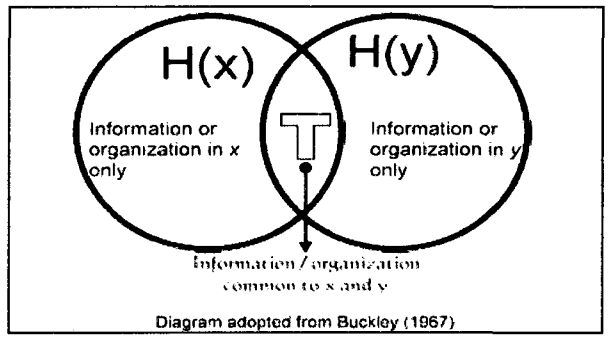


Figure 36: Information and Organization

Standards such as EDI and SWIFT represent a situation where the information common to two organizations that use such a standard (almost) entirely overlaps. "T" is huge; H(x) and H(y) are tiny if they exist at all. Usage of computerized ontologies may produce the same result, but so could any two data structures expressed in any DLL, provided an agreement of the minds between the owners of the autonomous data sources exists. Naturally, such an agreement between two parties is in direct opposition to the goal of human-free automatic data integration of heterogeneous data sources.

6.7 Chapter 6 Summary and Implications to the Research

This chapter introduced several CAS concepts originating in cybernetics. Combined with CAS related terms from linguistics and information theory introduced in previous chapters, this chapter connects between CAS, DDLs, and data integration among heterogeneous and autonomous sources..

From a CAS perspective, data integration is in essence the creation of a meaning preserving mapping, or *relation,* between an ensemble and its external constrained variety. Such mapping preserves the meaning of the variety vis-à-vis the IS, whose goal

is to integrate at least some external data. Chapter 4 (Approaches to Data Integration) reviewed prevailing approaches aimed at creating such mappings. Mappings, or relations, that last for the duration that they are needed, are held together by tension. Chapter 4 shows that existing data integration approaches to date do not implement a robust regulation mechanism, and do not yield tension unless humans intervene in the mapping process and invest mental energy to keep the relations from falling apart when a data source changes its data structure. In part such failures are due to the semantic heterogeneity discussed in chapter 5. Semantic heterogeneity is a manifestation of the theoretically infinite variety that exists in the environment. "Resolution" of semantic heterogeneity is CAS terminology is an attempt to constraint the variety. Each proposed method is a form of a regulator.

Tension in symbols-mediated CAS can be measured by formal meaning preservation requirements. In subsequent chapters it is shown that Sowa's prescription for meaning preservation (explained in chapter 7) could be relaxed somewhat and still create mappings that have tension.

The level of organization created by a specific set of relations out of all the possible sets (*complexion*) is measurable by using entropy. This is explored in more detail in chapter 7.

Implication of chapter to study	How Implication will be used
CAS	The advancement of data integration requires the recognition that IS are socio-technical systems. Its research requires a thorough grappling of CAS concepts for it to overcome the reverse salient it currently faces.
Variety	Autonomous heterogeneous data structures expressed in any DDL are constrained variety external to an IS that seeks to combine data from such sources. The variety needs to be mapped to the internal structure of the IS in a manner that is meaningful to the receiving end.
Tension	Inquire whether or not contemporary DDLs designed with data integration in mind provide the means necessary to create and maintain tension.
Entropy	Data integration creates some measurable order between the IS and its variety. This research measures entropy of a variety of DDLs in its assessment if they provide mechanisms to increase their internal order and therefore enable improved mappings.

Table 7: Summary of Chapter 6 Implications to the Study

CHAPTER 7

MEANING PRESERVATION

Chapter 6 explained the importance of mapping at least part of the available variety and constraints from its surroundings into the internal structure of a CAS. Further, the mapping should be isomorphic, at least in some aspects, to the original variety. Using information systems terminology, the CAS is a computerized application (information system) whose data is represented by one or more structures expressed in one or more DDLs. Data structures belonging to other organizations (or other independent information systems) and are published on public or privately accessible networks are the variety that is available in the environment. Correct mapping between (at least part) of an internal data structure and (at least part of) a relevant external data structure must occur for automatic data integration to take place. Correct mapping means that what is signified by the external variety is mapped to the same concept represented differently (or even identically) inside the information system that initiates the data integration. In other words, the signifier preserved its meaning regardless of its representation. Figure 37 provides an example of two data structures, each representing some aspects relating to timepieces. The item that provides energy is a common attribute in any timepiece. It is signified differently in each data structure, however. One data structure uses "Power_Source" the other uses "Makorkoach". This non-trivial mapping is correct because the meaning of the signified object is preserved.

This chapter explores the formal requirements for schemas mappings to preserve meaning, and provides examples where and how meaning preservation has been used in literature.

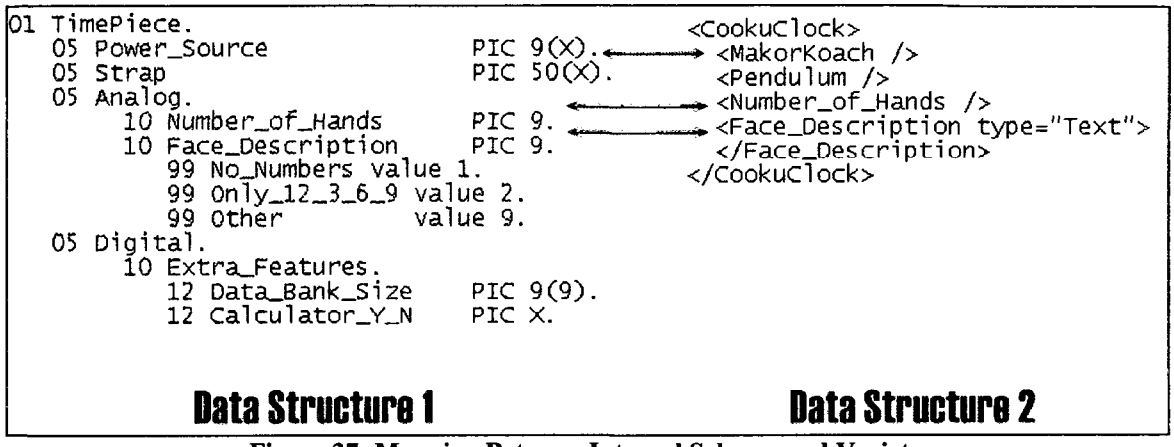


Figure 37: Mapping Between Internal Schema and Variety

7.1 The Essence of Meaning Preservation

The process of mapping one data source to another data source is a form of translation (Shu, Housel et al. 1975; Shu, Housel et al. 1977). A formal treatment of *meaning-preserving translation* from a language L_1 , to a language L_2 , may be defined as a function that satisfies the following constraints (Sowa 2001): Invertible, proof preserving, vocabulary preserving, and structure preserving.

- To satisfy *invertibility*, the translation function f must have an inverse function g that maps sentences from L_2 back to L_1. For any sentence s in L_1, $f(s)$ is a sentence in L_2, and $g(f(s))$ is a sentence in L_1.
- *Proof preservation* is satisfied when a sentence s in L_1 is translated to $f(s)$ in L_2 and back again to L_1 using the function $g(f(s))$.
- *Vocabulary preservation* is satisfied if s is translated from L_1 to L_2 and back by means of $g(f(s))$, the same *content words* or symbols that represent categories, relations, individuals etc., must appear in both sentences s and $g(f(s))$.
- *Structure preservation* is satisfied if, when s and g(f(s)) are normalized, they each contain the same number of negations and existential quantifiers, nested in semantically equivalent patterns.

Techniques that were accepted in peer reviewed publications all use relatively simple qualitative methods for evaluating meaning preservation. The following three examples are typical of such approaches:

1. Lee and Malone discuss computer-facilitated communication. Their primary concerns have to do with translating structured (or semi-structured) objects into other structured objects that can be processed by a receiver's computer system. They ignore most of the complexity and define **meaning preservation** in the following operational way: *The meaning of a message or an object is preserved in a translation if the recipients of the translated object can perform the same range of actions they would have wished to perform had they received and "understood" the original expression. We leave undefined the term 'understood' and appeal only to an intuitive sense of what it would mean for receivers to "understand" the original expression.*" (Lee and Malone 1990)

2. Wu and Wong propose some improvements to machine translation of natural languages. The authors "introduce a generalization of Wu's method with the objectives of improving **meaning-preservation** accuracy" (Wu and Wong 1998). The article that has been cited 23 times does not provide any metrics for meaning-preservation let alone measurement of improved accuracy. The authors use empirical examples from test runs to translate simple sentences from English to Chinese and then compare the output to a "standard" corpus. Accuracy seems to be judged by comparing the Chinese characters of the output and the standard corpus.

3. Bohan etc al "present a corpus-based method to evaluate the translation quality of machine translation (MT) systems" (Bohan, Breidt et al. 2000) with minimal user involvement. They evaluate meaning preservation using a single ordinal variable that assumes the values "good", "understandable" and "bad". They do not explain who makes the judgment call and under what guidelines.

7.2 Meaning Preservation in the IS Literature

A software agent is a program that intelligently performs its duties without human interaction, on behalf of a human or a system. It is autonomous (or semi-autonomous) proactive and reactive. For example, software that assembles news reports according to criteria specified by its owner. Software agents interact with networked data sources. Typically the data sources are on the World Wide Web. When an agent's ontology differs from the source it examines, even when the same natural language is used to define data structures, miscommunications can occur due to dialect differences. For example, American English uses "elevator" while British English uses "lift" to denote the same concept. Campbell suggests an Ontology Mediator that "learns" terminology present in web resources by asking its user questions. The Ontology Mediator builds its own representation of the source and the owner's ontology. The mediator's ontology is then used to find ontological translation candidates for further processing. The task of preserving meaning or finding meaning equivalences is entirely left to the human user (Campbell 2000).

Ontologies play a key role in making databases interoperate via the Internet. A database shared schema is intended for sharing meaning. Verheyden et al. argue that

meaning can be preserved via ontological commitments. The authors "*make precise the structure of this commitment layer by defining Ontology Base Reference and IDea Language (Ω-RIDL), a new type of commitment language*" (Verheyden, De Bo et al. 2004). A commitment language is the semantic coupling of attributes and relations to preserve meaning.

The challenge for meaning preservation is not confined to the field of integration only. Machine translation research is primarily concerned with meaning preservation. It is a concern for multi-lingual machine translation and for monolingual machine translation as well (Bateman 2002; Diab, Resnik et al. 2004; Andrés, Navarro et al. 2005; Hovy and Nirenburg 2005; Kishida 2005; Zhang, Gong et al. 2005).

Program restructuring is also an area of research that is concerned with meaning preservation (Griswold and Notkin 1995; Lee and Kim 2005; Liu, Mei et al. 2005) Our research is not concerned with program or query restructuring. This topic is mentioned for good measure only.

7.3 Meaning Preservation and CAS Properties

Data structures are incorporated into a global schema in the integration process. Queries from the global schema are sent to the source. This process poses a challenge similar in magnitude to the efforts in translating text from one natural language to another. For meaningful translation meaning must be preserved.

The mappings among components of a system vary along multiple dimensions. Figure 38 demonstrates such variations along a continuum of order, spanning from organized simplicity to chaotic complexity. Each state is developed hereafter.

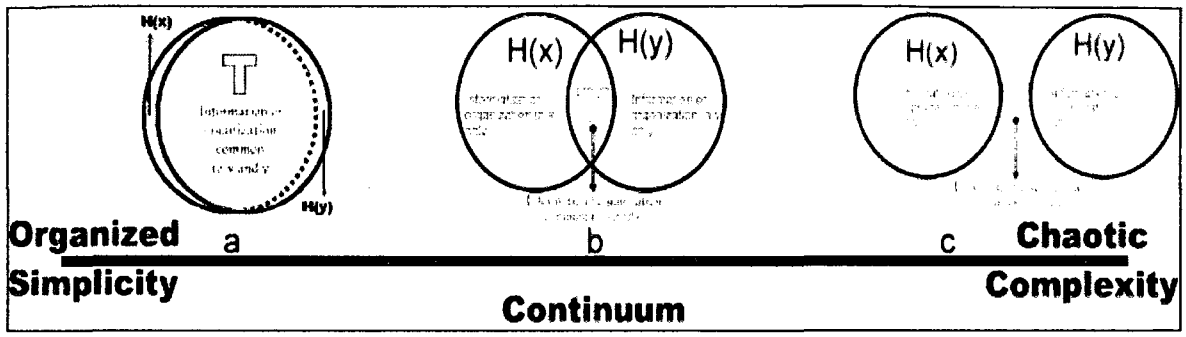


Figure 38: Systemic Relations

Interrelations in organized simplicity have very few degrees of freedom, their ability to change is rigid and restricted. On the opposite extreme we find chaotic complexity - an abundance of degrees of freedom in the relations among components such that the system can be described solely by statistical means, and there is little stable structure. Such structures have high tension, according to Buckley (page 48).

7.3.1 Organized Simplicity

On the extreme of organized simplicity there is total overlap of systems; there is no variety, there is total constraint and no degrees of freedom; there is nothing new X and Y can contribute to each other. X and Y share all meanings and there is no entropy at all. Therefore, Sowa's meaning preservation constraints are satisfied.

- *Invertibility*: The most important requirement upon which the others depend on is having f(x) and g(f(x)) for every (x). Since there is no variety and no degrees of freedom for every (x) there exists an f(X) and a g(f(X)). Therefore, invertibility is satisfied.
- *Vocabulary preservation* can be satisfied only if f(x) and g(f(x)) are satisfied. Since there is no variety and no degrees of freedom there exists a complete set of f(x); there are no concepts in X that do not map into Y; there are no

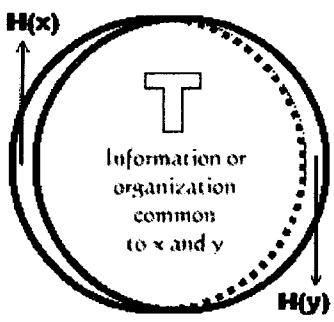
concepts in Y that don't map to X. Therefore, vocabulary preservation is satisfied.

- *Structure preservation* is satisfied if, when (x) and g(f(x)) are normalized, they each contain the same number of negations and existential quantifiers, nested in semantically equivalent patterns. Since there is no variety and no degrees of freedom there exists an f(x) for every concept (x). The same holds true for every concept in Y. Therefore, structure preservation is satisfied.
- *Proof preservation* is satisfied when all elements x in X are mapped by $f(x)$ into Y and back again to X by the function $g(f(x))$. Since there is no variety and no degrees of freedom there exists an f(x) for every concept (x); the same holds true for every concept in Y. Therefore, proof preservation is satisfied.

Strict standards are examples that fit into the Organized Simplicity extreme. EDI and SWIFT are two independent implementations of strict standards.

7.3.2 Almost Organized Simplicity

Moving slightly away from the Organized Simplicity extreme, to the point designated as

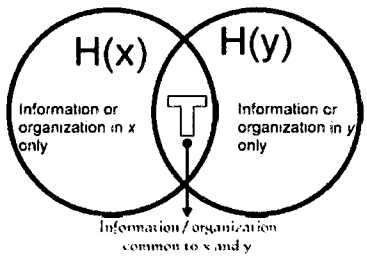


"a" in Figure 38 shows a state where there exists an area in ensemble (or system) X that has no overlap with ensemble (or system) Y, and vice versa. This state allows for some limited variety, and very little degrees of freedom in the constraints that exist between the systems. X can contribute a little bit to the evolution of Y, and if Y is a CAS it may choose to adapt some or all of that variety, as part of its goal seeking behavior. Along with the variety there is some entropy. The variety in X implies there are concepts (data elements) in X that do not exist in Y.

Since there is some variety, a complete set of *f(x)* cannot exist. Sowa's meaning preservation constraints cannot satisfied because the most important requirement for having *f(x)* and *g(f(x))* for every (x) is not achievable. The same holds true for *f(y)* and *g(f(y)).*That is due to the variety in X that precludes *f(x)* from being satisfied, unless the Y system evolves. X and Y have a lot of common meanings but that is less that absolute. This holds true for a state where Y is a complete subset of X, and also holds true for *f(y)* and *g(f(y)).*

7.3.3 Middle of the Road

Moving further away from the organized simplicity extreme toward chaotic complexity to a state illustrated by point "b" in Figure 38. Ensembles (or systems) X and Y have some information or organization that is common to both. This state allows for substantial variety, and significant degrees of freedom, although constraints exist between the ensembles (systems). X can contribute a little bit to the evolution of Y; if Y is a CAS it may choose to adapt some or all of that variety, as part of its goal seeking behavior. The variety in X implies there are concepts (data elements) in X that do not exist in Y.



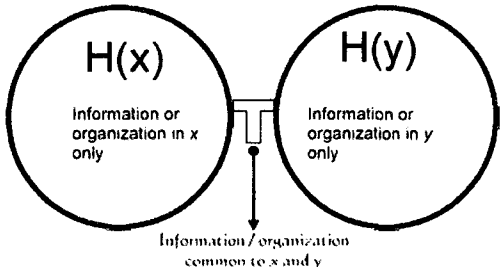
Sowa's meaning preservation constraints are not satisfied:

- *Invertibility*: The most important requirement for having f(x) and g(f(x)) for every (x) is not satisfied due to the variety in X that precludes f(x) from being satisfied. Although X and Y have a lot of common meanings, it is much less than absolute commonality, as we have seen in the organized simplicity case.

This holds true for a state where Y is a complete subset of X. Therefore, invertibility cannot be satisfied without a goal-seeking system evolution.

- *Vocabulary preservation* can be satisfied only if f(x) and g(f(x)) are satisfied. In this case we do not have a complete set of f(x); there are elements in X that do not map into Y; this precludes the possibility that symbols representing categories, relations, ideas and so forth, appear in both x and Y ensembles or systems.
- *Structure preservation* is satisfied if, when (x) and *g(f(x))* are normalized, they each contain the same number of negations and existential quantifiers, nested in semantically equivalent patterns. However, since *f(x)* has limited (partial) existence, structure preservation cannot be satisfied. The same holds true for *f(y)* and *g(f(y)*.
- *Proof preservation* is satisfied when all elements *x* in X are mapped by *f(x)* into Y and back again to X by the function *g(f(x))*. However, since *f(x)* has limited (partial) existence, proof preservation cannot be satisfied. The same holds true for *f(y)* and *g(f(y)*.

7.3.4 Chaotic Complexity



Moving all the way to chaotic complexity is illustrated by point "c" in Figure 38. The ensembles (or systems) X and Y have no information or organization in common. This state allows for vast variety, and virtually infinite degrees of freedom, where NO constraints exist between the ensembles (systems). X can

contribute any or all of its part to the evolution of Y; if Y is a CAS it may choose to adapt some or all of that vast variety, as part of its goal seeking behavior. The entire variety in X consists of concepts (data elements) do not exist in Y, and vice-versa.

Sowa's meaning preservation constraints are not satisfied:

- *Invertibility*: The most important requirement for having *f(x)* and *g(f(x))* for every (x) is not satisfied due to the variety in X that precludes *f(x)* from being satisfied even for a single (*x*). X and Y have no common meaning.
- *Vocabulary preservation* can be satisfied only if *f(x)* and *g(f(x))* are satisfied. In this case f(x) is an empty set; there are no elements *x* in X that can map into Y; this precludes the possibility that symbols representing categories, relations, ideas and so forth, appearing in both x and Y ensembles or systems.
- *Structure preservation* is satisfied if, when (x) and *g(f(x))* are normalized, they each contain the same number of negations and existential quantifiers, nested in semantically equivalent patterns. Since *f(x)* does not exist, structure preservation cannot be satisfied. The same holds true for *f(y)*.
- *Proof preservation* is satisfied when all elements *x* in X are mapped by *f*(*x*) into Y and back again to X by the function *g*(*f*(*x*)). However, since *f(x)* does not exist, proof preservation cannot be satisfied. The same holds true for *f(y)*.

7.4 Chapter 7 Summary and Implications to the Research

Compared to techniques that were accepted in peer reviewed publications, this research will utilize a method that appears to be quite more sophisticated. It will evaluate meaning preservation based on formal requirements: the research uses Sowa's measurements to

conclude whether or not meaning preservation constraints can be (or is) satisfied in any DDLs examined in this work.

Implication of chapter to study	How Implication will be used
Bijective mapping	Correct mapping between two signifiers that represent the same signified is a prerequisite for any data integration. For automatic data integration correct mapping must be achieved without human intervention. Assess if any DDL in this study supports such mechanism by design.
Meaning Preservation	Used to examine and conclude if any DDL supports meaning preservation as part of its design and / or attributes.

Table 8: Summary of Chapter 7 Implications to the Study

CHAPTER 8

DISTRIBUTION OF WORDS AND MEANINGS

Chapter 6 introduced the concept of variety and its importance to CAS, and along with chapter 7 established that successful data integration creates more complex structures as morphogenic systems adapt by combining variety from the environment. Therefore, assessing if variety exists at all, and if so to what degree, is significant to a research that utilizes CAS theory. Methods Zipf developed provide a tool to quantify variety.

8.1 On the Economy of Words

George Kingsley Zipf (1902-1950) was a Harvard linguist who in the 1930s noticed that the distribution of words adhered to a regular statistical pattern (Zipf 1949). The same pattern is a characteristic of many natural and human phenomena as well. Zipf also made observations relating to the economic value of languages, and its how economic value influences the usage of language. Zipf also made observations as well as on the growth and shrinkage of languages.

Central to Zipf's research is counting of words, and counting the number of meanings for each word, calculate their distributions, analyze the findings and draw conclusions. This research utilizes these methods on samples taken from ontologies and data schemas expressed in a variety of DDLs.

The following passages examine Zip's Principle of Least Effort, Zip's Distribution of Words and Zip's Distribution of Meanings.

8.2 Zipf's Principle of Least Effort

Zipf uses two imaginary people who communicate using words to illustrate his point. One person is talking. The other one is listening. The communication's originator has the task of selecting the meaning and the words to convey the meaning using words. From the originator's point of view, the most economical vocabulary consists of one word that would mean whatever the originator wants it to mean. Therefore, if there are m dissimilar meanings to convey, that single word would have m different meanings. Consequently, the message originator would be spared the effort necessary to obtain and maintain a large vocabulary, select the particular words from it to convey a particular meaning.

The recipient's task is to decipher the meaning of the communication. He would face the impossible task of determining the particular meaning of the single word in a given situation. Having m words for m meanings would be ideal for the receiver, saving efforts required to determine a particular meaning of a communication.

These two aspects of communications represent two different economies: that of the transmitter and that of the receiver. The two economies are in extreme conflict. Each economy represents a force: one is a unification force that tends to reduce the size of a vocabulary to a single word; the other is a diversification force, aiming at expanding the vocabulary to a one-to-one ratio between words and meanings. Zipf wrote "the unification force acts in the direction of decreasing the number of words to one, while the diversification force acts in the opposite direction of increasing the number of different words, while decreasing their average frequency of occurrence to one"(Zipf 1949).

8.3 Zipf's Distribution of Words

Zipf studied word frequencies in corpora in an attempt to see if vocabularies are able to balance the two forces and reach equilibrium of some sort. He found that texts in many different languages throughout history produced strikingly similar results. When those results were graphed on a double logarithmic scale they were always close to a straight line. Mathematically, it is a decaying function because it decreases at a rate proportional to its value. The line also represents a probability distribution that can also be expressed in a form of a *Power Law*. A power law relationship between two scalar quantities x and y is any such that the relationship can be written as $y=ax^k$ where "a" and "k" are constants.

It is possible to rank-order the words and count the frequency of each word appearing in a given text. The word that occurs most often is assigned the rank r = 1; the second most frequent word has rank r = 2, and so on. Zipf observed that a word's rank *r* multiplied by it frequency *f* yields a constant *C*. In mathematical notation it is expressed as: r * f = C

Zipf noticed that the distribution of words reflects the probability of a given word to be used. The probability P of a word is the number of occurrences of that word, divided by the total number of words in the text. Zipf's empirical law of word frequencies in its simplest form is:

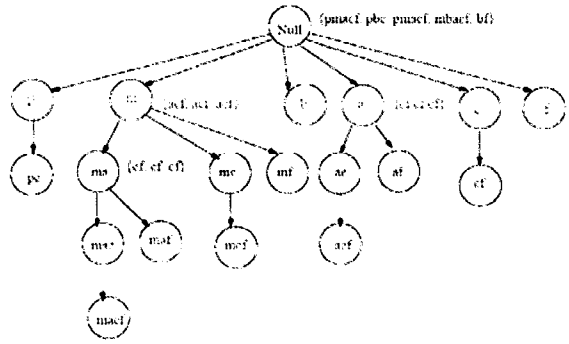
$$P \propto 1/r$$

Where P is the probability of a word to appear in rank "r". P(word) is proportional to one over the rank of that word in the text.. The Zipf power law of word frequency is near

perfectly hyperbolic on a regular (non-logarithmic) scale. Mandelbrot found in it an underlying scaling property (Frame and Mandelbrot 2003) expressed as follows:

$$P \propto \frac{F}{(r+V)^{1/D}}$$

In Mandelbrot's version P is the probability of a word, F is the Frequency of the word in the text, r is the rank of the word, V is the vocabulary D is a scaling dimension. The formula represents a lexicographical tree, as illustrated on the right. Lexicographical trees are self-similar where each branch is in some way a reduced-scale version of the whole tree.



An object is said to be self-similar if it looks "roughly" the same on any scale. Self-similar objects with parameters N and S are described by a power law such as $N=S^d$ where "d" is defined as ln(N)/ln(S). Natural languages are not perfect lexical trees with self similarities, but perhaps engineered languages are. This needs additional investigation in the future.

8.4 Zipf's Law of Meaning Distribution

Zipf pondered further and discovered that words at the top of given distribution lists also have multiple meanings. Zipf used a 1938 work done by Irving Lorge & Edward Thorndike titled *A Semantic Count of English Words* as an objective source to determine meanings of words. Further empirical investigation lead Zipf to conclude that there exists a distribution of meanings expressible as follows:

$$m_r = \sqrt{F_r}$$

Where m_r is the number of meanings associated with the word whose rank is r, and F is the frequency of the word whose rank is r.

8.5 Usage of Word Distribution in Information Systems

Information Retrieval (IR) systems use word (or term) distributions as a key tool to estimate the relevance of a document to a given query. IR systems rely on automatic indexing and term weighting to make a relevance assessment. Three major approaches have emerged in IR: (a) the probabilistic model; (b) the vector-space model; (c) the latent semantic indexing model. Each one is explained in the following sections.

8.5.1 Probabilistic models

A query in IR systems seeks to estimate the probability that a specific document will be judged as relevant with respect to a given query. There is an assumption that terms are distributed differently within relevant and non-relevant documents. In order to estimate this probability the system regards the distribution of terms in the document. The IR system counts, ranks and clusters words and terms of inspected documents. A basic IR indexing method was first suggested by (Maron and Kuhns 1960) The indexing weight of a word or a term in a document is an estimate of the probability of relevance of this document with respect to a query using a given descriptor. The method has many variants, and has been used to date (Berger and Lafferty 1999; Dalvi and Suciu 2005).

8.5.2 Vector-space model

The vector-space model (Salton and Lesk 1965) extends the aforementioned probabilistic model. Statistical data for term frequency and distribution is used to create vectors in multi-dimensional space. The length of the vector represents the importance of the term. Vector similarity is used to rank relevance of retrieved documents to the query. The vector-space model was first used in the SMART Information Retrieval System developed at Cornell University in the 1960s headed by Gerard Salton (Salton and Lesk 1965; Salton, Wong et al. 1975; Salton 1991). The vector space model is used to date, in applications such as spam detection (Zhan, Lu et al. 2005) and Internet searches (Sinka and Corne 2005). One method for spam detection uses a calculated term weight that's a multiplier of Term Frequency (TF) and Inverse Document Frequency (IDF). Term Frequency counts the frequency of a term inside a given document, such as email. The weight of the term in TF is proportional to its frequency in the document. Inverse Document Frequency counts the frequency of a term among all documents in a system or in a given inbox of an email account. The weight of the term in IDF is inversely proportional to its frequency among all documents in the system. It is worth noting that words with high IDF weight are not useful to discriminate between messages because they are "too popular". The calculated index term weights with relation to each document and to the spam detection query are stored in vectors. The correlation between the document vector and the query vector measures document to query relevance. The correlation is the cosine of the angle between the two vectors. Cosine values range from [–1 to +1]. The spam filtering system can set a threshold somewhere in between the two values and flag as spam those emails that are above the threshold.

8.5.3 Latent semantic indexing

Latent semantic indexing is based on co-occurrence clustering of terms (Deerwester, Dumais et al. 1990). For example, both "lawyers" and "attorneys" are likely to belong to the same cluster with related terms such as "courts", "trial", "judge", "sentencing", and "jury" but not with terms like "referee" or "umpire". Latent semantic indexing (LSI) improves the way in which the problem of multiple terms referring to the same object has been dealt with. LSI replaces individual terms as the descriptors of documents by independent "artificial concepts" that can be specified by any one of several terms or combinations thereof. LSI characterizes and identifies relevant documents that do not contain the terms of the query, or whose contained terms are qualified by other terms in the query but not both. Some improvements of this approach are researched and proposed to date, such as in (Husbands, Simon et al. 2005) and (Dasgupta, Kumar et al. 2005). The improvements are technical in nature and out of scope of our proposed research.

8.6 Chapter 8 Summary and Implications to the Research

DDLs enable the creation of data structures that use relations (e.g., hierarchy) and signifiers. Signifiers are words in natural language or other combinations of alphanumeric symbols. This research measures the degree of variety allowed (or disallowed) by DDLs. For that it is using Zipf's approach as part of its method to assess if real progress has been made in DDLs towards enabling automatic data integration from heterogeneous sources and to discover potential underlying fundamental characteristics of DDLs.

Implication of chapter to study	How Implication will be used
Distribution of Words	Measures the external degree of variety allowed (or disallowed) by DDLs.
Distribution of Meanings	Measures the degree of inner variety in data structures expressed in different DDLs.

Table 9: *Summary of Chapter 8 Implications to the Study*

CHAPTER 9

INFORMATION THEORY

This chapter gives a brief overview of Information Theory and then focuses on entropy and its relevance to CAS and to the study of DDL. CAS theory references entropy in various ways. For example, morphogenic systems export their entropy (disorder) so they can maintain their internal organization. Entropy's complement, redundancy, is an indication of constraint. Therefore, entropy and redundancy are quantitative variables that can be measured in constructs that facilitate data integration, namely DDL.

9.1 The Origins of Information Theory

Ralph Vinton Lyon Hartley (November 30, 1888 – May 1, 1970), an electronics researcher, contributed to the foundations of information theory in his 1928 paper entitled Transmission of Information. He formulated the problem of electrical communications as a process in which a sender has a set of symbols (e.g., an alphabet) from which he selects symbols in a sequence. The information of the message denoted as H was defined by Hartley as the logarithm of the number of possible sequences of symbols which might have been selected. He showed that $H = n \log_2 s$ where n stands for the number of symbols selected and s is the number of distinct symbols in the set. For example, the Hebrew Alphabet has 22 letters. If a sender chooses to create a three letter word, then the information size the sender can generate is 3 $\log_2$ 22, which equals to 13.37; if the sender uses the 26 letters of the English alphabet, then H equals to 3 * $\log_2$ 26, yielding 14.10; Had the sender opted for sequences of 7 characters, H would be 7 $\log_2$ 26 = 32.90.

World War Two (WW-II) gave rise to radar controlled anti-aircraft gunnery. Radar receivers picked up radar signals and noise together – an ensemble of possible signals indicating the location and trajectory of an airplane. How does one estimate what the signal is despite the noise is what Wiener solved in the US. It was solved independently by the mathematician Kolmogoroff in the USSR, at about the same time. Claude Shannon viewed the noisy communications in a different way: if a sender produces a sequence of symbols and sends them over a noisy channel, what is the best way to encode the sequences such that it will produce the largest number of messages per time unit while keeping the errors to minimum? In attacking the problem Shannon confined the source to be *ergodic* That is, a source that produces an infinite sequence of ensembles, where the source's statistical characteristics do not change over time. "All writers of the English Language together constitute an approximately ergodic source" (Pierce 1961). This observation is consistent with Zipf's observation on the English language. However, "all men writing French and all men writing English could not constitute an ergodic source" (Pierce, 1961 p. 60). This observation might be of interest when examining data structures created by multi-national teams, such as the Harmonise Travel Ontology discussed in Chapter 12.

The message producing source chooses what sequences to send. Although the choice could be somewhat constrained (e.g., a "TH" in the middle of an English word will not be followed by a "Q"), the receiving side is uncertain what the message is, until examined, as this resolution of uncertainty is *the goal of communication.*

9.2 Entropy

When a message source has no other choice but to produces a single symbol, such as a string of zeroes, the recipient need not receive and examine the message, as it is predictable with utmost certainty. The recipient *knows* at any time what the message was, is, and what it will be. If the message source can produce a message of one bit, it can generate either zeroes or ones, creating a need to examine the incoming message. The traffic light with three lamps mentioned by Ashby (see chapter 6) can produce 2^3 different outcomes, increasing the uncertainty of its messages until examined by the recipient. The constraints engineers put on the traffic light result in fewer possibilities it can generate, therefore reducing the uncertainty. But how does one go about measuring the size of information in a message? That's what Shannon did, and it turns out his mathematical formula is very similar to that of entropy in physics.

Information Theory (Shannon 1948) defines *information* as only those symbols that are uncertain to the receiver. The term information is defined abstractly as uncertainty (Bailey 1994). Information theory has nothing to do with any inherent meaning in a message. Information theory measures the degree of order, or non-randomness and treats it mathematically much as mass or energy or as other physical quantities are treated. The units of measure are not joules or amperes; they are measured in binary digits, or *bits*. Suppose there is an ergodic source that produces only two symbols, at a known probability p_0 and p_1 for each symbol respectively. The amount of information, or entropy for this source is $H = -\left(p_0 * \log\left(p_0\right) + p_1 * \log\left(p1\right)\right)$ bits per symbol. If the ergodic source produces n symbols, then its entropy would be

$H = -(p_0 * \log(p_0) + p_1 * \log(p_1) ... + p_n * \log(p_n))$ or $H = -\sum_{i=1}^{n} p_i \log_2 p_i$ bits per symbol.

9.2.1 Relative Entropy

An ergodic source that generates *n* symbols whose probabilities of occurrence is identical (e.g., $p_1 = p_2 = \ldots = p_n = 1/n$) it is said to have *Maximum Entropy*, which can be expressed as $\log_2 n$. Rolling a fair dice or flipping a fair coin are examples of such ergodic sources.

However, many ergodic sources generate sequences of symbols whose probabilities of occurrence are not identical. That is, $p_1 \neq p_2 \neq \ldots \neq p_n$ Therefore, its *Actual Entropy* needs to account for the different probabilities, which is what $H = -\sum_{i=1}^{n} p_i \log_2 p_i$ does.

The ratio between the actual entropy and the maximum entropy is said to be the *Relative Entropy*, and defined as $H_{relative} = \frac{H - Actual}{H - Maximum}$

9.2.2 Redundancy: Constrained Entropy

According to Shannon's Information Theory, *redundancy* is a measure for the constraint amount (or size) imposed on a text in a given language. Redundancy is due to syntactical rules, where for every syntactic rule there is a constraint that must introduce some redundancy. The statistical structure of any given language itself exhibits a certain amount of redundancy as well, according to Shannon and as shown by Zipf.

Information Theory defines redundancy as the difference between maximum entropy and actual entropy, expressed as a ratio as follows:

$$\text{Redundancy} = \frac{MaxEntropy - ActualEntropy}{MaxEntropy}$$

An alternate definition of redundancy is: Redundancy = 1 – Relative Entropy. The equation suggests that maximizing the actual entropy of a set of symbols minimizes the redundancy associated with that set, which is a desirable outcome for efficient communications.

9.2.3 Entropy of English and Hebrew

This research examines various data structures formed by various DDLs using two natural languages: English and Hebrew. Therefore, it is of interest to know what is the entropy of each language, and perhaps compare it to the entropy calculated for DDL.

According to Pierce, if one ignores the relative frequencies of letters in English, it would require 4.76 bits to encode each letter, which is the maximum entropy. If the relative frequencies of letters are taken into account, then it would require 4.03 bits per letter, which is the actual entropy. Further, "if we encode word by word, taking into account relative frequencies of words, we require 2.14 bits per character" (Pierce 1961 page 104). Shannon estimated the entropy of characters in the English language to be between 0.6 and 1.3; subsequent experiments indicate it is about 1.1; as for English words, when taking into account their relative frequencies, the entropy of English is 2.14 bits per character (Pierce 1961).

Boyarsky and Gora calculated the maximum entropy of the Hebrew language to be about 12.2 bits per character, the actual entropy to be about 3.7 per character and the relative entropy to be about 0.30 per character (Boyarsky and Gora 2000).

9.3 Entropy and CAS Properties

The physicist and Nobel laureate Erwin Schrödinger uses entropy in the context of thermodynamics to assert that living organisms maintain their organization (structure) as follows:

> *It feeds upon negative entropy, attracting, as it were, a stream of negative entropy upon itself, to compensate the entropy increase it produces by living and thus to maintain itself on a stationary and fairly low entropy level. If D is a measure of disorder, its reciprocal,* ***1/D, can be regarded as a direct measure of order.*** *Since the logarithm of 1/D is just minus the logarithm of D, we can write Boltzmann's equation thus: -(entropy) = k log (1/D). ...thus the device by which an organism maintains itself stationary at a fairly high level of orderliness really consists continually sucking orderliness from its environment* (Schrodinger 1944).

Buckley quotes George A. Miller's words (Miller 1953) clarifying the relation between entropy and order, as follows: "A well organized system is predictable – you know almost what it is going to happen before it happens. A perfectly organized system is completely predictable and its behavior provides no information" that you did not already have about the system. Organization (e.g., arrangement) "can be visualized as a number of elements, where each element has its own set of alternative interactions with other elements. Each element has some freedom of choice of interaction, but also some constraints" (Buckley 1967, p. 87). A *complexion* is any specific set of choices out of all the possible sets, made by each element. The number of complexions in the organization (arrangement) is the number of possible alternatives one can choose from. This is equivalent to ensemble of variety in Shannon's information theory, and the entropy measure H thus can be used. If the elements were entirely independent and had no

interaction constraints then all combinations have equal probability, resulting in maximal entropy and zero organization. On the other hand, if the constraints are such that only one set of complexions is allowed, then there is zero entropy and maximum organization. However, in dynamic organizations such as Complex Adaptive Systems the set of complexions is not limited to one but is significantly less than the maximum entropy. Specifically, sociocultural systems are "a set of elements linked by way of the intercommunication of information" (Buckley 1967). While in lower level systems information is energy, in higher level systems information "is a relationship between sets or ensembles of structured variety" (Buckley 1967). Organization measures the amount of constraint introduced to a collective. It is possible to represent two ensembles of complexions (collectives, groups, societies, data structures) by *x* and *y*. The entropy H(x) represents the organization of *x* and similarly H(y) represents the organization of *y*. The amount of common or compatible organizational interaction is represented by "T", as shown in Figure 36 below. Sociocultural systems that differ substantially would be expected to have a narrow common area, whereas sociocultural systems with very similar organization would be expected to have a very large overlap, or a large "T".

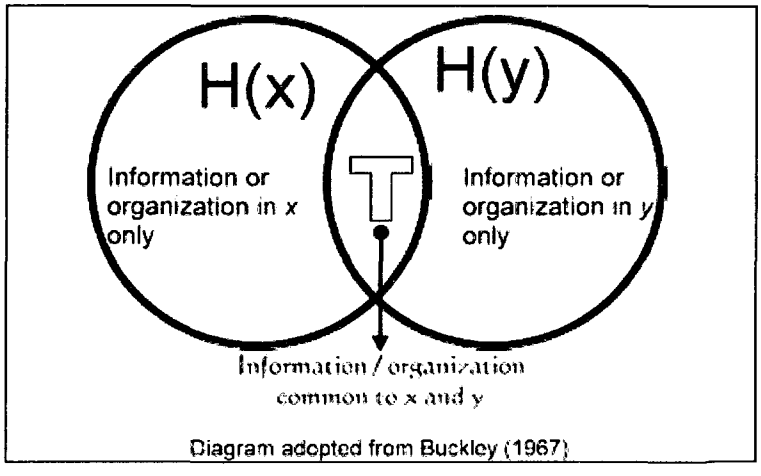


Figure 39: Information and Organization

9.4 Information Theory in the IS Literature

Minsky postulated in a series of articles the 1970's that a useful way to represent knowledge resulting from common sense reasoning, language, memory or perception is to break it into modular and structured chunks he termed "frames" (Minsky 1975; Minsky 1984). Bayle critically reviewed the Frames theory (Bayle 1989). He shows that the Frames approach is ill suited for representation of incomplete knowledge. Bayle states that "more in-depth work in the manner of Shannon's work in information theory is required if progress towards a unified theory of knowledge representation is to be achieved" (Bayle 1989).

The process of defining new measures for software complexity is ongoing. "It is still difficult to relate the measures to the phenomena we want to improve" writes Dospisil (Dospisil 2003) as she relates to an entropy-based complexity measure for object-oriented designs suggested by (Bansiya, Davis et al. 1999). Complexity is one of ten measures they suggest in addition to classical object oriented (OO) software metrics such as abstraction, encapsulation and coupling. Bansiya et al. suggest a new complexity measure of object-oriented classes: complexity is the degree of difficulty in understanding and comprehending the internal and external structure of classes and their relationships. Several large commercial object-oriented projects have been used to validate the approach.

Measurements of software complexity have been using entropy-based measure to assess functional complexity. (Abran, Ormandjieva et al. 2004) present "an exploratory study of related concepts across information theory-based measures and functional size

measures". The degree of complexity is measured in terms of an amount of information that's interacted among software components.

The efficacy of information retrieval systems can be measured from a system-oriented view point or a user-oriented view point. There are a number of performance measures, where each measure evaluates different aspects of retrieval performance. (Aslam, Emine et al. 2005) analyze the quality of various retrieval performance measures and propose a quality evaluation model based on maximum entropy. They assert that their demonstrated maximum entropy model corresponds to other known measures of overall performance and that the maximum entropy model can be used to accurately predict the value of other measures of performance.

Natural language processing applications that rely on statistical translation use maximum entropy. Statistical machine translation systems are mainly based on statistical word lexicons (Garcia-Varea and Casacuberta 2005). Such lexicons are typically context-independent. This means that such systems rely on other probabilistic distributions for meaning disambiguation. The authors postulate adding contextual information to statistical lexicons by using maximum entropy modeling. They build a stochastic model that takes into account a larger context than straight probabilistic distributions. They choose the distribution preserving as much uncertainty as possible in terms of maximizing the entropy (Garcia-Varea and Casacuberta 2005). The distribution is required to satisfy constraints, which represent facts known from the data. For instance translating a word W into f(w) when a word W' appears in within a given distance from W. The constraints are expressed as a function Hmax(s,t). where (s,t) is a pair of source and target word (Berger, Della Pietra et al. 1996). Through experimentation Garcia-Varea

et al. show that in fact usage of maximum entropy models improve the performance of the statistical translation systems by choosing the conditional distribution probability that maximizes the conditional entropy.

In summary – Information Theory has been used to develop measurements of software complexity as well as information retrieval efficacy.

9.5 Chapter 9 Summary and Implications to the Research

Shannon's measure of entropy is used in CAS. We show in the methodology chapter that measuring redundancy may be helpful in the assessment of the adequacy of new approaches to automatic data integration.

Implication of chapter to study	How Implication will be used
Entropy	The magnitude of order in a system
Redundancy	The difference between maximum entropy and actual entropy, expressed as a relation to the maximum disorder possible in a system
Complexion	Any specific set of choices out of all the possible sets

Table 10: Summary of Chapter 9 Implications to the Study

CHAPTER 10

METHODOLOGY

In the forgoing chapters we have presented a detailed orientation into different disciplines and the manner they are utilized in this research, as well as short summaries of how they have been used in information systems. This chapter describes the methodology used in this research. The research utilizes a synthesized, multi-disciplinary methodology for defining and measuring a set of CAS properties in DDL: variety, tension and entropy. This methodology is then used to test the existence or absence of particular constraining phenomena caused by or influencing these CAS properties. These phenomena are then constructed as indicators common to all DDLs including some of the most recent technologically deterministic solutions. The results are then used to answer the research questions.

The chapter begins with a description of qualitative and quantitative analysis methods that are adequate and necessary to satisfy the research objectives. Naturally, these principles and methods all relate to the various disciplines and their associated measures discussed earlier: words and meanings distribution, meaning preservation, and entropy. It then describes the data collection methods, followed by the resulting database of study, and concludes with a discussion on data validity.

10.1 Approach

DDLs are abstract constructs used to describe the organization of data in a system. The abstract constructs become tangible by means of implementation. That is, by the creation of data structures obeying the DDL rules (constraints). Therefore, the data collection

requires the gathering of DDLs implementations. This research uses publicly available data structures expressed in different DDLs, and tests for the existence, and magnitude where applicable, of CAS properties necessary for automatic data integration. The results of each DDL are analyzed and contrasted with results obtained for other DDLs. Findings are analyzed using qualitative methods and quantitative calculations. The research systemizes the findings by proposing a framework for the evaluation of DDL suitability to automatic data integration.

Chapter 6 has a detailed discussion of CAS properties identified as significant to data integration. Those properties cannot be directly measured in non-physical environments, such as social systems or socio-technical systems (Buckley 1967; Burrell and Morgan 1979; Bailey 1994; Buckley 1998). Hence, measures by proxy are necessary. The CAS properties in focus and the methods of measurement selected for them are: *variety*, measured using qualitative analysis of ambiguity and quantitative analysis of Zipf distributions; *tension*, measured using meaning preservation as a nominal variable; and, *order*, measured using entropy as a ratio variable. Each of the measures and its relation to CAS has been discussed in length in earlier chapters.

10.1.1 Measuring Variety

Variety in DDL is measured using distribution of signifiers (words in natural language, in most cases), the distribution of signifiers' meanings, and signifiers' examination for five types of ambiguity. Measuring *Signifiers Distribution* is done by tallying the frequency of occurrence of each word that participates in a representative schema chosen for this research. The second measure for variety, the *Number of Meanings* is achieved by using WordNet (Miller 1995; WORDNET 2005), a lexical database for the English language,

to retrieve the number of meanings of each participating word in English; for Hebrew words, the Babylon computerized dictionary is used to determine the number of meanings per word. For EDI and SWIFT, the number of meanings is derived from their formal documentations, respectively. *Word Ambiguity* is assessed qualitatively against five ambiguity types. Table 11 illustrates how the different variables indicative of variety are used, followed by several sections providing explanations for each measurement.

	1980's				1990's				
Parameter	**Protocol**		**Data Dictionary**		**Markup Language**				
DDL	**Proprietary**		**Natural Language**		**DTD**	**XML**	**XML**	**XSD**	**XML**
Model Name	EDI	SWIFT	ADABAS	NCREIF	RETS	REPML	MFDX	MISMO	REXML
# of Data Elements					454	262	138	138	1862
# of Unique Words					212	212	105	105	323
Total Words					452	454	231	231	3832
Internal Ambiguity									
Word sense					Y	Y	Y	Y	Y
Structural					Y	Y	Y	Y	Y
Projection					Y	Y	Y	Y	Y
Referential					Y	Y	Y	Y	Y
Resolution					Y	Y	Y	Y	Y
Cross Standard Ambiguity					Y	Y	Y	Y	Y

Table 11: Measures of Variety

Word distribution. The number of words in each "alphabet" (a collection of signifiers) are counted and then ranked according to their usage frequency. Frequencies are plotted, and a regression coefficient is calculated.

Number of meanings distribution. The number of meanings each word has is counted for each alphabet, and ranked according to the number of meanings. Frequencies are plotted, and a regression coefficient is calculated.

Word ambiguity. Natural language exhibits numerous types of ambiguities (Gardent and Webber 2001). If the ambiguity can be resolved by the recipient using "common sense" it is an attribute of an open system, otherwise it is an attribute of a closed system. Existence of ambiguity is tested and classified using the following five classes of ambiguity:

- **Word sense ambiguity** occurs when a word has several meanings. For example, "bark" could be a noun (a tree's bark) or a verb (make barking sounds like a dog). Word sense ambiguity ties back to the discussion on Zipf's distribution of meanings.
- **Structural ambiguity** occurs when analyses of a string yield a multiplicity of syntaxes. For example, the statement "old shoes and hats" is ambiguous because is not clear whether the adjective "old" applies to the hats, to the shoes or to both. This type of ambiguity does not depend on an interaction with an observer, therefore it is a closed system concept.
- **Projection ambiguity** refers to the various ways in which a presupposition (e.g., subjective knowledge about the world,) can be integrated into the overall meaning of a text. For example: "Tom inherited $2,000 from its deceased owner". Tom could be a cat's name, but it could also be a name of a slave when slavery was a common practice in the US. Even the value of $2,000 is an example of projection ambiguity, as its current value is far less than what it was, say, in 1795. This type of ambiguity depends on an observer's interpretation tied to the observer's surroundings; therefore it is an open system concept.
- **Referential ambiguity** occurs when it is uncertain what a particular natural language expression is referred to. For example, "he is beautiful" can have a different meaning to an observer, depending upon whether it was said by a man or women. This type of ambiguity depends on the message generator as well as the recipient's socio-cultural interpretation; therefore it is an open system concept.
- **Resolution ambiguity** covers the many possible ways in which semantically underspecified elements (e.g., anaphor[5], ellipsis[6],) can be interpreted in a given context. "It happened yesterday" is an anaphor – "it" is an expression referring to another. This type of ambiguity depends on an observer's understanding of what "it" refers to, therefore it is an open system concept. "John took the order and

---

5 A word or phrase that takes its reference from another word or phrase and especially from a preceding word or phrase

6 Omission of one or more words that are obviously understood but that must be supplied to make a construction grammatically complete

shipped the package" is ellipsis – omitting the process of picking and packing that might have taken place in between the two actions. This type of ambiguity depends on an observer's ability to connect the dots, so to speak. Therefore it is an open system concept.

- **Cross-Structure ambiguity.** For a mapping to occur between a data structure and another one in its environment, the mapping should preserve meaning. A barrier to that end would be cross structure ambiguity. For example, a system's data structure expressed in any DDL may have an element expressed using a term, such as "ListDate". The external data structure, expressed in any DDL, may have a concept expressed as "OriginalListingDate", hinting that there could be more than one listing date (e.g., UpdatedListingDate). This causes resolution ambiguity, which may prohibit the possibility of a meaning preserving mapping between the two terms.

10.1.2 Measuring Tension

Successful mapping between a data structure, expressed in any DDL, to some autonomous and heterogeneous data structure needs to "make sense", in other words - preserve the meaning of the external variety vis-à-vis the internal structure of the system whose goal is the integration. Per CAS such a mapping produces tension. The process of mapping one schema to another data structure is a form of translation (Shu, Housel et al. 1975; Shu, Housel et al. 1977). A formal treatment of *meaning-preserving translation* from a language L_1 to a language L_2 has been discussed in chapter 6 in and chapter 7 at length. It demonstrates that a difference in the number of nodes (e.g., data elements) between two data structures precludes invertibility and therefore precludes vocabulary preservation and precludes structure preservation. Suffice to re-emphasize at this point that meaning preservation must satisfy the *invertibility* requirement (a translation function f must have an inverse function g that maps sentences from L_2 back to L_1). If such an

inverse function is not supported, then meaning cannot be preserved, and no tension will be created. The results of a pilot study and additional related work (Rohn and Klashner 2001; Rohn and Klashner 2004; Rohn 2006; Rohn 2007) indicate no DDL exists that was designed to supports meaning preservation (Sowa 1999; Sowa 2001). Therefore, for the purpose of this research suffice to measure tension as a nominal variable that assumes "true" or "false" values. If this research will come across a DDL designed to support tension, future work will be necessary to establish a methodology for gauging the "strength" of the tension.

The existence of an $f(x)$ and its inverse $g(f(x))$ are assessed qualitatively by attempting to map each data structure to at least one other data structure in the sample. Comparison of the number of nodes and their depth in each pair of data structures is a first step. If similar nodes exist, an attempt to map them in a manner that preserves their meaning is made. The result of the attempted mapping noted. Every such mapping is assessed for feasibility without human intervention using the mechanism provided by each DDL. If more than one data structure exists for the same vertical market (e.g., Real Estate), then a mapping is attempted from the data structure under investigation to each one of the other data structures in that domain. Such mapping attempt is illustrated in Figure 37.

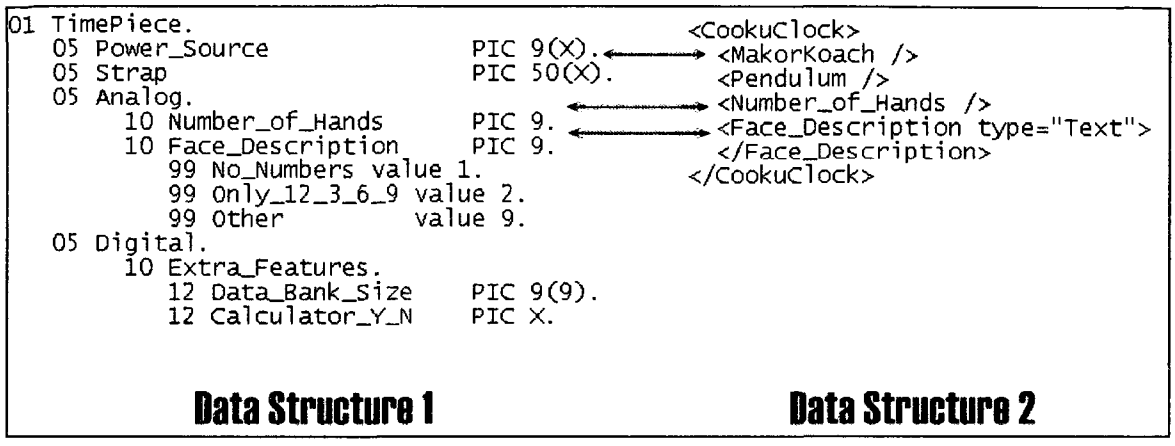


Figure 40: Exploring Support for Meaning Preservation

10.1.3 Measuring Entropy

Morphogenic systems reduce their local entropy and increase order. This research utilizes entropy (Shannon 1948) as a direct measure of the level of order achieved by utilizing a given DDL. The calculation is done using the following formula (discussed in chapter 9):

$H = -\sum_{i=1}^{a} p_i \log_2 P_i$ a = # of words in the alphabet and P*i* = the probability of word *i*

H and P*i* are calculated for every DDL individually. The frequency of each word used in a DDL is tallied, and the results are used to calculate the probability for each word. The steps and formulas used for the process are illustrated in Table 12.

# of Words	49
# of Occurences	339
H-Maximum	5.6147 $\log_2$ of 49
H-Actual	4.8425 - Sum (P_i * Log_2 P_i)
H-Relative	0.8625 (H-Actual) / (H-Maximum)
Redundancy	0.1375 1 - Relative Entropy
Entropy of Alphabet (words)	

Table 12: An Example of Entropy Calculation

Von Bertalanffy demonstrated the physical equivalence of thermodynamics entropy and information theory entropy (Raymond 1950). Buckley quotes George A. Miller's words (Miller 1953) to clarify the relation between entropy and order, as follows: "*A well organized system is predictable – you know almost what is going to happen before it happens. A perfectly organized system is completely predictable and its behavior provides no information*" that you did not already have about the system (see organized simplicity in section 7.3.1). Organization (e.g., arrangement) "*can be visualized as a number of elements, where each element has its own set of alternative interactions with other elements. Each element has some freedom of choice of interaction, but also some constraints*" (Buckley 1967, p. 87).

This research hypothesizes and empirically demonstrates that the level of entropy in a given data structure is an indication of its fitness for automatic integration. The entropy of a DDL is measured to assess its probability to successfully facilitate the integration of data structures. "The important distinction between open and closed systems has often been expressed in terms of entropy" (Buckley 1967).

10.2 Data Gathering Method

The study uses DDL samples representing technologies from several computing generations – from the 1960's to date. Standards are represented by SWIFT and EDI. Structured DDLs are represented by COBOL FD sections, and by ADABAS (Batory 1985) data structures. Semi-structured DDLs are represented by numerous XML, DTD and XSD structures. Ontologies are represented by RDF and OWL structures. Gathering DDLs from different vertical markets and written in two unrelated languages minimizes the risk of bias due to a specific market, a specific natural language, or the cultural

background of the schema creators. Vertical markets represented in the data sample are Banking, Real Estate and the Travel industry. Data structures were created in either English or Hebrew transliterated into Latin-based alphabet.

Measuring variety using Zipf's method requires counting symbols, or signifiers. The signifiers can be words in a natural language, or numeric identifiers, or any other form of signifiers. For example, in EDI the atomic symbols, or "alphabet" of signifiers, are message types (e.g., message 518, message 584). For COBOL signifiers are field names in the program's FD section. For SQL these are tables and column names. For XML these are tags (e.g., <*BrithYear*>). For RDF and OWL these are triplets. Composite symbols engineered in natural language (complex words made out of more than a single word or acronym) are broken down into their building blocks – single words. For example, from the XML tag `<xs:attribute name="CloseDate">` the attribute's name: `CloseDate` is extracted, and then broken down into its two atomic components: `Close Date`. The same process is repeated for each composite symbol.

This process creates a list of symbols or words that are counted, ranked and analyzed in terms of the number of meanings per word, their ambiguity classification, and their meaning preservation. See Appendix A: Data Gathering Illustration for a pictorial explanation.

10.3 Database of Study

A collection of publicly available data structures using a wide variety of DDLs has been identified and secured for further analysis. It consists of EDI, SWIFT, Cobol, Adabas, XML, DTD, CSD, RDF and OWL. This section describes the manner in which each DDL sample was allocated and chosen to be part of the sample used in this study.

The EDI standard is represented by a group of 165 different messages identified by the EDI governing body as EDIFACT Version D.99B Messages. See Appendix B: Edifact Version D.99b Messages for a list of those messages.

The SWIFT standard is represented by a group of messages identified by the SWIFT governing body as Financial Institution Transfers Messages MT200 through MT293 (SWIFT 2005) and Securities Markets Messages MT568 through MT599 (SWIFT 2005).

Structured DDLs are represented by two COBOL structures, and an ADABAS structure. COBOL has one English based schema and one Hebrew based schema. Adabas has a Hebrew based schema. See Table 14 for more details.

XML was designed to facilitate data exchange among computerized systems. Consequently, thousands of schemas became publicly available in a very short time. Many of the schemas compete with each other, resulting in substantial conceptual overlap. A survey by the author of this work in 2004 of XML business data exchange vocabularies identified over 2000 different XML business vocabularies publicly available for immediate use.

Leading IS scholar and MISQ co-editor, Lynne Markus, chose to examine a certain aspect of the Real Estate Market to illustrate her discussion of the standardization of XML-based e-business frameworks (Markus, Steinfield, & Wigand, 2003). Markus uses only MISMO in her analysis of standards, and refers to the Uniform Electronic Transactions Act (UETA) in 1999 as an "eMortgage standard". This research follows Markus' footsteps and focused on real estate vocabularies expressed in semi-structured DDLs.

Stakeholders in the real estate market include assessors, county recorders, realtors, appraisers, mortgage bankers, mortgage brokers, insurers, buyers, sellers, and renters. One would expect to find a vertical market standard that serviced the needs of its different stakeholders. However, there is none. Rather, there exists a variety to choose from. Table 13 lists some of the efforts and initiatives taken to standardize XML based Real Estate related data exchange. This research uses the entire set of these XML sources.

Organization / Source	Abbreviated Source Name
The Real Estate Transaction Standard	RETS
Real Estate Listing Markup Language	REPML
The Multifamily Data Exchange	MFDX
National Council of Real Estate Investment Fiduciaries	NCREIF
Realm Software and Financial Services	REXML
Mortgage Industry Standards Maintenance Organization	MISMO

Table 13: Real Estate Schemas Used in this Work

RDF is represented by a travel "standard" proposed by the Travel Agent Game in Agentcities or TAGA in short (Zou and Finn 2003; Youyong 2005). It is a framework designed by the University of Maryland, Baltimore County (UMBC) to support agent-based market simulations and games. It was first used in a simulation of travel agents competing to provide travel packages to customers traveling from City A to City B. A travel package includes a round-trip flight ticket, hotel accommodations and ticket to entertainment events.

OWL is represented by an ontology that claims proposed standard status for the Travel industry, namely HARMONISE Travel Ontology (Höpken 2005) from the Tourism Harmonisation Network (THN), set up by the Harmonise project partners consortium with funding from the European Commission's IST program. Twelve

European travel websites currently use HARMONISE to create an electronic market place for tourism.

Table 14 summarizes the DDLs represented in this research. Each DDL has at least one implemented schema used for analysis purposes.

DDL Type	DDL Syntax	Schema Name
Ontology	**RDF**	TAGA
	OWL	HARMONIZE
Markup Language	**DTD**	RETS
	XML	REPML
	XSD	MISMO
	XML	MFDX
	XML	REXML
Prog. Lang.	**Cobol FD**	Cobol FD
	COBOL FD	Israeli Bank (Hebrew Structure)
Data Dictionary	**Natural Language**	NCREIF
		Israeli Bank (Hebrew Structure)
Protocol	**Proprietary**	EDI
		SWIFT

Table 14: Summary of examined DDLs

10.4 Validity of Data

No questionnaires or similar data collection instruments are used in this study. The data used are collected directly from the field without intermediaries or inference. As such, face validity and content validity are integral to the data itself.

10.5 Originality and Limitations of Data

Each data structure collected for this study is originally published by its owners. We have no control over the data structures, their length, composition, and complexity. Tens of thousands (if not more) data structures exist in the world. The study is limited to a handful of data structures faithfully representing different approaches and DDL generations.

10.6 Chapter 10 Summary and Implications to the Research

DDLs enable the creation of data structures that use relations (e.g., hierarchy) and signifiers. Signifiers are words in natural language or other combinations of alphanumeric symbols. This multi-disciplinary methodology establishes a way to measuring a set of CAS properties in DDL: variety, constraint, tension and entropy, as shown in Table 15.

CAS Attribute	Measure Used	Method of Measurement	Type of Variable
Variety	Zipf Distribution of Words	Counting, plotting, Correlation Coefficient	Ratio
	Zipf Distribution of Meanings	Counting, plotting, Correlation Coefficient	Ratio
	Ambiguity	Assessment	Nominal
Tension	Meaning Preservation	Evaluate if g(f(x)) exists	Nominal
Entropy	Info. Theory	Calculation	Ratio

Table 15: Summary of CAS Attributes and Their Measurement Methods

The existence of variety is required for a CAS based analysis of relations and mappings with some of the environment. Variety is measured primarily using Zipf distribution of words and meanings, yielding a ratio variable. Ambiguity assessment is used as secondary measure of variety, validating the primary findings. Ambiguity is measured as a nominal variable assuming the values "Y" and "N". The quantitative measure of variety is also used to verify or refute that the data sources are ergodic, which is necessary for the application of Information Theory based measures, such as entropy.

Tension is necessary to create and maintain mappings between an IS and some of the external variety surrounding it. Correct mapping between (at least part) of a data structure and (at least part of) a relevant external data structure must occur for automatic data integration to take place. Correct mapping requires that what is signified by the external variety is mapped to the same concept represented differently (or even identically) in the information system DDL that initiates the data integration. In other words, that the

signifier preserved its meaning regardless of its representation. The type of tension created by such mapping is non-physical, and therefore it is measured in non-physical terms. A relaxed version of Sowa's meaning preservation rules is used to determine if tension does or can exist. The first rule assessed for each data structure and DDLs in the sample is invertibility. A mapping function *f* from the system's DDL (L_1) to elements in an external DDL (L_2) must have an inverse function *g* that maps from L_2 back to L_1. For any mapped elements *e* in L_1, f(e) is an element in L_2, and g(f(e)) is an element in L_1.

Entropy is used to measure the degree of order in a data source. The higher the entropy the less organized a source is, and the less knowledge one can obtain from it. Entropy is used to assess if newer DDL are significantly different from older ones in the manner in which they export their inner entropy and thus maintain order. Entropy is calculated by using the formula $H = -\sum_{i=1}^{n} p_i \log_2 p_i$ for actual entropy, $\log_2 n$ for maximum entropy and $H_{relative} = \frac{H - Actual}{H - Maximum}$ for relative entropy.

Redundancy is a measure for the constraint amount (or size) imposed on a text in a given language by some syntactic rule. DDLs also have syntactic rules, which constrain them. The difference between maximum entropy and actual entropy is redundancy, and is measured as follows: $\frac{MaxEntropy - ActualEntropy}{MaxEntropy}$.

In summary, each CAS property is assessed or measured using an established approach, taken from a scientific discipline relevant to the property of interest. The chapter rationalized the selection of methods and their application to DDLs. Table 16 summarizes the attributes and their measures, pointing to how they impact this study.

Implication of chapter to study	How Implication will be used
Variety	A CAS property indicating heterogeneity in the environment surrounding any given system
Ambiguity	Qualitative measure to determine existence and nature of variety
Words Distribution	Quantitative measure to determine existence and magnitude of variety external to a given schema
Meanings Distribution	Quantitative measure to determine existence and magnitude of variety internal to a given schema
Tension	A CAS property indicating successful mapping between at least part of the CAS and the variety surrounding it
Meaning Preservation	A nominal measure to determine if a given DDL supports the creation and maintenance of tension
Order	A CAS property related to the degree of available energy sustaining the internal arrangement of a given system. High-level CAS, such as socio-technical systems, do not use physical energy to maintain their inner order. Physical thermodynamics entropy and information theory entropy are equivalent (See chapter 6 section 5.3 for details)
Entropy	A measurable ratio variable indicating the magnitude of order imposed by constraints of any given DDL, indicative of its fitness to automatic integration.

Table 16: Summary of Chapter 10 Implications to the Study

CHAPTER 11

THE DATA COLLECTION

The purpose of this chapter is to qualitatively and quantitatively describe the thirteen schemas gathered from publicly available resources. Per the methodology established in chapter 10, every sampled schema undergoes several steps, such as extraction of field names from the data structure, discovery of atomic symbols ("alphabet"), counting, and calculations of focal attributes to arrive at their CAS underlying constructs. Each schema is accompanied by a summary page of tables and figures pertaining to the data descriptions preceding it. Schemas are presented in this order: COBOL (English), COBOL (Hebrew), ADABAS (Hebrew), NCREIF, RETS, REPML, MFDX, REXML, MISMO, HARMONISE, TAGA, EDI, and SWIFT. Detailed descriptions have been provided in Chapter 10 section 3 and is summarized in Table 17.

DDL Type	DDL Syntax	Schema Name	Domain
Ontology	**RDF**	TAGA	Tourism
	OWL	HARMONIZE	Tourism
Markup Language	**DTD**	RETS	Real Estate
	XML	REPML	Real Estate
	XSD	MISMO	Real Estate
	XML	MFDX	Real Estate
	XML	REXML	Real Estate
Prog. Lang.	**Cobol FD**	Cobol FD (English)	Neutral
	COBOL FD	Cobol FD (Hebrew)	Banking
Data Dictionary	**Natural Language**	NCREIF	Real Estate
		Hebrew Structure	Banking
Protocol	**Proprietary**	EDI	Finance
		SWIFT	Banking

Table 17: Database of Study

11.1 COBOL (English)

COBOL (common business-oriented language) is a 3rd generation procedural programming language, developed in 1959 (Sammet 1978). It has undergone several modifications and standardization efforts. COBOL allows connotative variables and constants naming – it permits both long names (up to 30 characters in early versions) and dashes to connect word.

A medium size COBOL program was selected as a case study item. The program contains Input, Output, calculations, and all elements of program control, such as branching, conditional operations etc. Procedure names, Variable names and Constants names have been extracted from the program and isolated for analysis, removing all COBOL pre-defined syntax. For example, from the two following COBOL lines

```
move VKSD0080-STATUS to IO-STATUS
perform Z-DISPLAY-IO-STATUS
```

the variable names VKSD0080-STATUS, IO-STATUS, Z-DISPLAY-IO-STATUS were used, and the rest ignored, as the rest is COBOL's DDL prescribed syntax constraints.

11.1.1 COBOL (EN) Words Distribution

The COBOL data structure has 57 elements comprised of 333 words after breaking holophrases into single words. It consists of 49 unique "words".

The frequency usage of words is summarized in Table 20 on page 162. Here are two examples to help read the table: There are 24 words that appear 1 times in the COBOL schema. There are two words appearing 12 times. These two words are MESSAGE and STATUS. Their usage is shown in Table 21 and Table 22 respectively. It is of interest to note that the expression VKSD0080 appears 12 times as well.

We graphed the distribution of words and word frequencies for COBOL shown in Figure 41 and Figure 42 respectively. We calculated a power regression having a correlation coefficient R^2=0.92 for word distribution and a correlation coefficient R^2=0.64 for words frequency distribution.

11.1.2 COBOL (EN) Words - Number of Meanings

Word	# of Meanings
GET	36
RETURN	29
POST	23
CLOSE	20
KEY	20
LEFT	19
END	18
OPEN	15
RECORD	13
RIGHT	12
TITLE	12
PROGRAM	10
DISPLAY	9
FILE	9
BUFFER	8
HEADER	7
RESULT	6
CODE	5
MESSAGE	5
ZERO	5

Table 18: COBOL (EN) Meanings

COBOL (EN) has 49 unique "words" with a total of 308 meanings. Table 18 has the words with the highest number of meanings. Leading the list is "GET" with 36 different meanings according to WordNet, followed by "RETURN" with 29 meanings. We graph the meanings distribution and calculated a power regression coefficient giving R^2=0.98 as shown in Figure 51. We noted that the most used words in COBOL (EN) ("Message" and "Status") are not the ones that have the highest number of meanings. As per WordNet "Message" has 5 meanings and "Status" has 2 meanings.

11.1.3 COBOL (EN) Entropy Calculations

# of Words	49	
# of Occurences	339	
H-Maximum	5.6147	$\log_2$ of 49
H-Actual	4.8425	- Sum (P_i * Log_2 P_i)
H-Relative	0.8625	(H-Actual) / (H-Maximum)
Redundancy	0.1375	1 - Relative Entropy

Table 19: COBOL (EN) Entropy

Entropy was calculated according to the formulas presented in the literature review section of Shannon's Shannon's Information Theory on page 134. COBOL (EN) relative entropy is 0.8625

Frequency	Frequency Count
1	24
2	14
3	6
4	5
5	2
6	1
7	2
9	1
12	2

Table 20: COBOL (EN) Word requency

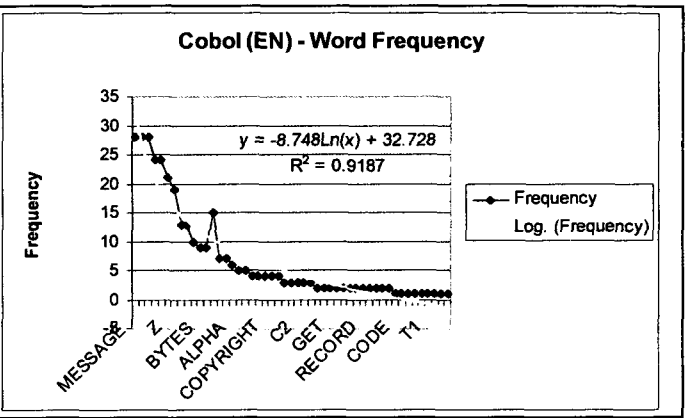


Figure 41: COBOL (EN) Word Frequency

MESSAGE-TEXT (12)
Z-DISPLAY-MESSAGE-TEXT (9)
MESSAGE-BUFFER (3)
MESSAGE-TEXT-2 (2)
MESSAGE-HEADER (1)
MESSAGE-TEXT-1 (1)

Table 21: COBOL (EN) Usage of "MESSAGE"

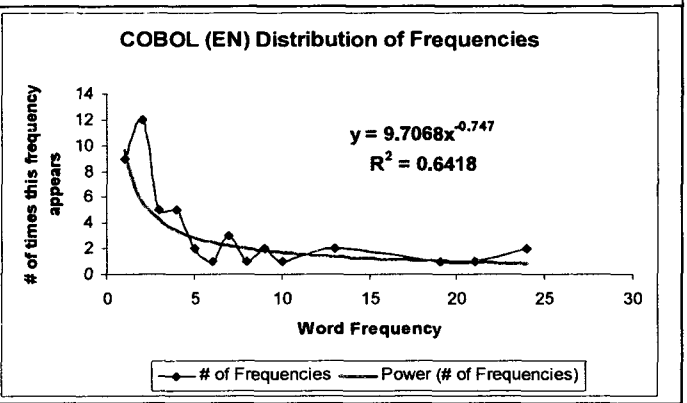


Figure 42: COBOL (EN) Distribution Frequency

VKSD0080-STATUS (9)
IO-STATUS (6)
IO-STATUS-04 (6)
Z-DISPLAY-IO-STATUS (4)
IO-STATUS-0403 (2)
IO-STATUS-0401 (1)

Table 22: COBOL (EN)Usage of "STATUS"

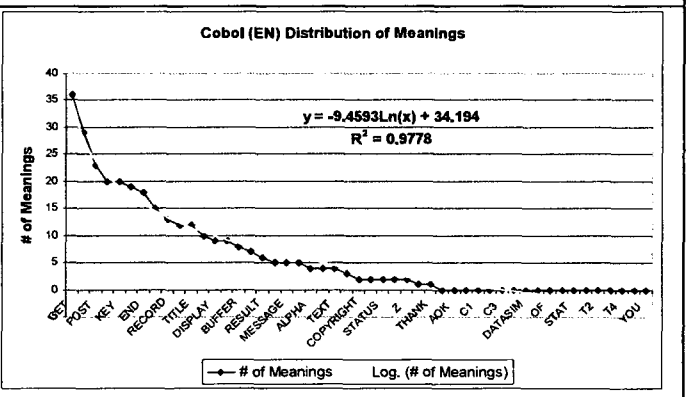


Figure 43: COBOL (EN) Meaning Distribution

11.2 COBOL (Hebrew)

This COBOL Hebrew data structure (Appendix C) is taken from an Israeli banking information system. It has 73 elements comprised of 167 words after breaking holophrases into single words. It consists of 77 unique "words".

11.2.1 COBOL (Heb) Words Distribution

Word	Freq.
KOD	10
ISKA	8
TAR	7
SNIF	6
CHN	5
M	5
MISPAR	5
NEGDI	5
SCHUM	5
SUG	5
BANK	4
MEKORY	4
PEULA	4

Table 23: COBOL (Heb) Words Distribution

The most used words in this data structure are summarized in Table 23. The distribution of words for the Hebrew COBOL data structure are summarized in Table 23. Here are two examples to help read the table: There are 42 words that appear 1 times in the COBOL (Heb) schema. There is one word appearing 10 times. This word is KOD. The next most frequent word is "ISKA" (transaction / deal). Their usage is shown in Table 27 and Table 28 respectively.

We graphed the distribution of words and word frequencies for COBOL shown in Figure 44 and Figure 45 respectively. We calculated a power regression having a correlation coefficient R^2=0.91 for word distribution and a correlation coefficient R^2=0.90 for words frequency distribution.

11.2.2 COBOL (Heb) Words - Number of Meanings

COBOL (Heb) has 77 unique "words" with a total of 491 meanings. Table 24 has the words with the highest number of meanings. Leading the list is "MAARECHET" (application / information system) with 25 different meanings, followed by "ERECH" (value) with 16 meanings. We graph the meanings distribution and calculated a power regression coefficient giving R2=0.90 as shown in Figure 46. We noted that the most

11.3 ADABAS (Hebrew)

This ADABAS Hebrew data dictionary structure (Appendix D) is taken from an Israeli banking information system written in the mid 1980's.

11.3.1 ADABAS (Heb) Words Frequency

Word	Freq.
Shem	16
Yeled	16
Kod	11
Ben	8
Loazit	8
Tar	8
Zihui	8
Zug	8
M	7
Pirtey	6
Prati	5
Sug	5
Telefon	5
Shem	16
Yeled	16
Kod	11
Ben	8

Table 29: ADABAS Words

This data structure has 82 elements comprised of 193 words after breaking holophrases into single words. It consists of 68 unique "words". The words used most frequently in this data structure are summarized in Table 29. The distribution of words for the Hebrew ADABAS data structure are summarized in Table 32. Here are two examples to help read the table: There are 39 words that appear 1 times in the COBOL (Heb) schema. There are two words appearing 16 times. They are SHEM (name) and YELED (child). Their usage is shown in Table 33 and Table 34 respectively.

We graphed the distribution of words and word frequencies for ADABAS shown in Figure 47 and Figure 48 respectively. We calculated a power regression having a correlation coefficient R^2=0.92 for word distribution and a correlation coefficient R^2=0.65 for words frequency distribution.

11.3.2 ADABAS (Heb) Words - Number of Meanings

ADABAS (Heb) has 68 unique "words" with a total of 601 meanings. Table 30 has the words with the highest number of meanings. Leading the list is "AVODA" (work / occupation) with 27 different meanings, followed by "ANAF" (branch) with 25 meanings. We graph the meanings distribution and calculated a power regression

used words in COBOL (Heb) (""KOD" and "ISKA") are not the ones that have the highest number of meanings. "KOD" has 4 meanings and "ISKA" has 6 meanings.

Word	Hebrew	# of meanings	Possible English Meaning
MAARECHET	מערכת	25	Application / System
ERECH	ערך	16	Value
CHN	חשבון	15	Account
TNUA	תנועה	14	Transaction
MOSHEC	מושך	13	Withdrawer
CROSS	קרוס	13	Counter Account
CHOVA	חובה	13	Debit
TASH	תש	11	Payment
PEULA	פעולה	11	Action
PAKID	פקיד	10	Clerk
NOSAFIM	נוספים	10	Additional
MISPAR	מספר	10	Number (of)
MEFUTZELET	מפוצלת	10	Split (transaction)
MAAVAR	מעבר	10	Intermediate (account)
KVUTZA	קבוצה	10	Group
HAMARA	המרה	10	Exchange
ACHUZ	אחוז	10	Percent
ZCHUT	זכות	9	Credit
TKUFA	תקופה	9	Period

Table 24: COBOL (Heb) Words with the most meanings

11.2.3 COBOL (Heb) Entropy Calculations

# of Words	77	
# of Occurrences	166	
H-Maximum	6.2668	$\log_2$ of 49
H-Actual	5.8731	- Sum (P_i * Log_2 P_i)
H-Relative	0.9372	(H-Actual) / (H-Maximum)
Redundancy	0.0628	1 - Relative Entropy

Table 25: COBOL (Heb) Entropy

Entropy was calculated according to the formulas presented in the literature review section of Shannon's Information Theory on page 134. COBOL (Heb) relative entropy is 0.9372.

Freq.	Total
1	42
2	15
3	7
4	3
5	6
6	1
7	1
8	1
10	1

Table 26: COBOL (Heb) Words Frequency

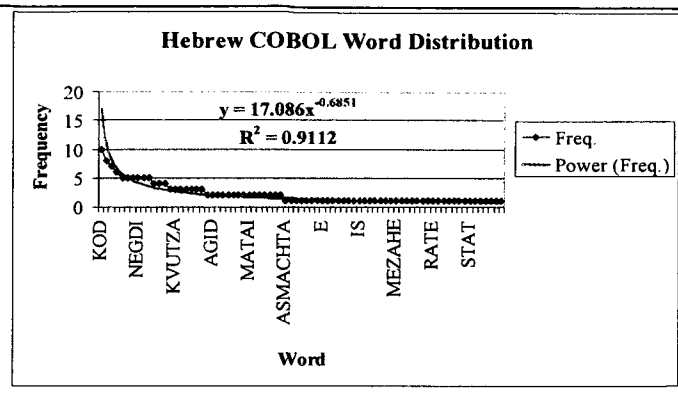


Figure 44: COBOL (Heb) Word Meaning Distribution

TMF-KOD-CHOVA-ZCHUT
TMF-KOD-STORNO
TMF-KOD-TASH
TMF-KOD-TEUR
TMF-KOD-ZIHUI-TOFES
TMF-KOD-PAKID
TMF-KOD-MAARECHET
TMF-KOD-ISKA-ATIDIT
TMF-KOD-SHAAR
TMF-KOD-TASH-MEKORY

Table 27: COBOL (Heb) Usage of "KOD"

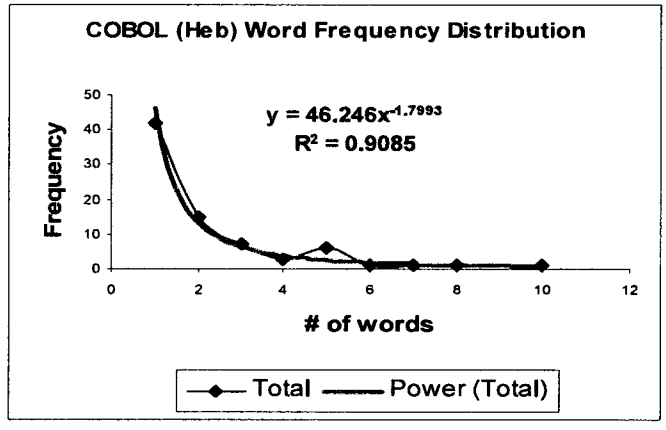


Figure 45: COBOL (Heb) Word Usage Distribution

TMF-SUG-ISKA
TMF-MISPAR-ISKA
TMF-TAR-ERECH-ISKA
TMF-KOD-ISKA-ATIDIT
TMF-M-RATZ-LE-ISKA
TMF-M-TNUA-LE-ISKA
TMF-HGDARAT-ISKA
TMF-SUG-ISKA-MEKORY

Table 28: COBOL (Heb) Usage of "ISKA"

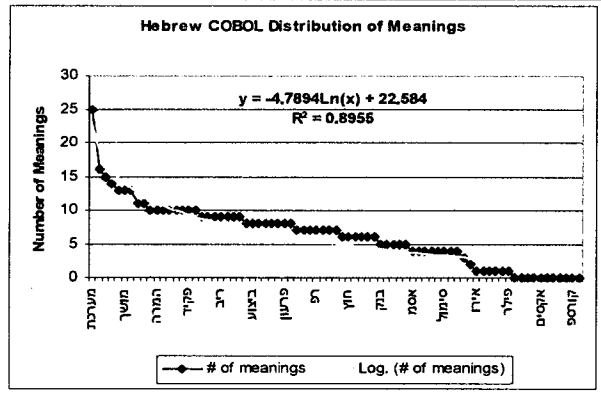


Figure 46: COBOL (Heb) Word Meaning Distribution

11.3.3 ADABAS (Heb) Entropy

# of Words	69	
# of Occurences	193	
H-Maximum	6.1085	$\log_2$ of 49
H-Actual	5.4380	- Sum (P_i * Log_2 P_i)
H-Relative	0.8902	(H-Actual) / (H-Maximum)
Redundancy	0.1098	1 - Relative Entropy

Table 31: Adabas (HEB) entropy

Entropy was calculated according to the formulas presented in the literature review section of Shannon's Information Theory on page 134. ADABAS (Heb) relative entropy is 0.8902.

Frequency	# of Words with this Freq.
1	39
2	10
3	5
4	2
5	3
6	1
7	1
8	5
11	1
16	2

Table 32: ADABAS (Heb) Words Frequency

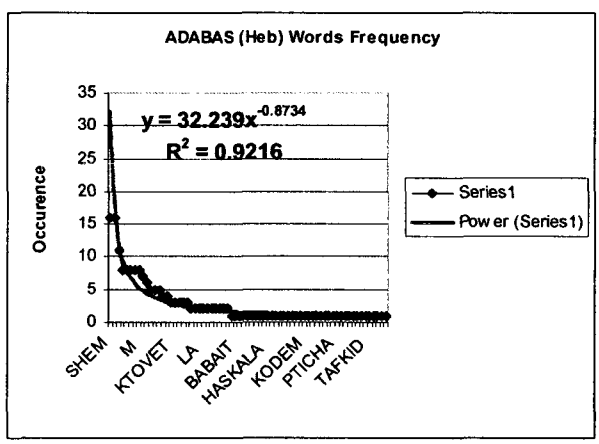


Figure 47: ADABAS (Heb) Word Meaning Distribution

coefficient giving R^2=0.94 as shown in Figure 49. We noted that the most used words in ADABAS (Heb) ("SHEM" and "YELED") are not the ones that have the highest number of meanings. "SHEM" has 3 meanings and "YELED" has 10 meanings.

Word	Hebrew	# of meanings	Possible English Meaning
AVODA	עבודה	27	work
ANAF	ענף	25	branch
KESHER	קשר	24	connection
KODEM	קודם	21	previous
YACHAS	יחס	20	relation
YESUD	יסוד	17	foundation
TOKEF	תוקף	16	effective (date)
CHESHBON	חשבון	15	account
CHN	חשבון	15	account
ISUK	עיסוק	14	occupation
KLALIIM	כללים	14	regulation
MIN	מין	14	Sex
KASHUR	קשור	13	link
MISHPACHA	משפחה	13	Family
TA	תא	13	cell
BAIT	בית	12	home
HASKALA	השכלה	11	education
PEULA	פעולה	11	Action
PRATI	פרטי	11	private

Table 30: ADABAS (Heb) Words with the most meanings

SHEM
SHEM-PRATI
SHEM-MISHPACHA
SHEM-MISH-KODEM
SHEM-BEN-ZUG
SHEM-PRATI-BEN-ZUG
SHEM-MISHP-BEN-ZUG
SHEM-YELED-1
SHEM-PRATI-YELED-1
SHEM-MISHP-YELED-1
SHEM-YELED-2
SHEM-PRATI-YELED-2
SHEM-MISHP-YELED-2
SHEM-LOAZIT
SHEM-PRATI-LOAZIT
SHEM-MISHP-LOAZIT

Table 33: ADABAS (Heb) Usage of SHEM

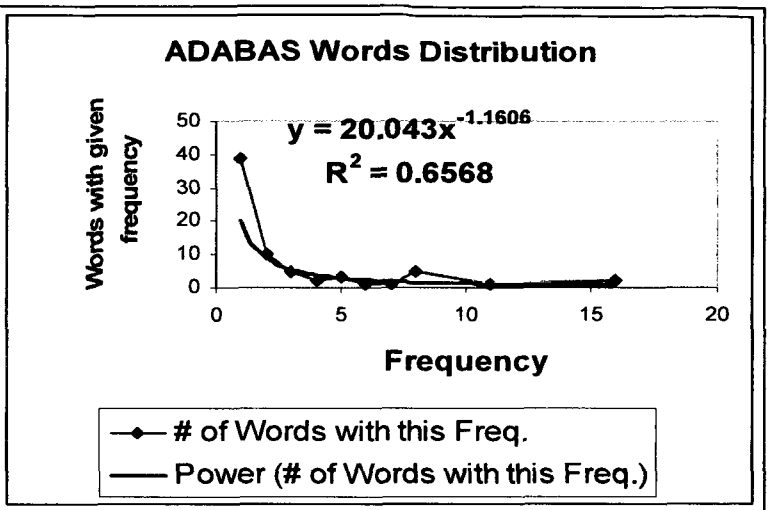


Figure 48: ADABAS (Heb) Word Usage Distribution

PIRTEY-YELED-1
SHEM-YELED-1
SHEM-PRATI-YELED-1
SHEM-MISHP-YELED-1
KOD-ZIHUI-YELED-1
M-ZIHUI-YELED-1
MIN-YELED-1
TAR-LEDA-8-YELED-1
PIRTEY-YELED-2
SHEM-YELED-2
SHEM-PRATI-YELED-2
SHEM-MISHP-YELED-2
KOD-ZIHUI-YELED-2
M-ZIHUI-YELED-2
MIN-YELED-2
TAR-LEDA-8-YELED-2

Table 34: ADABAS (Heb) Usage of YELED

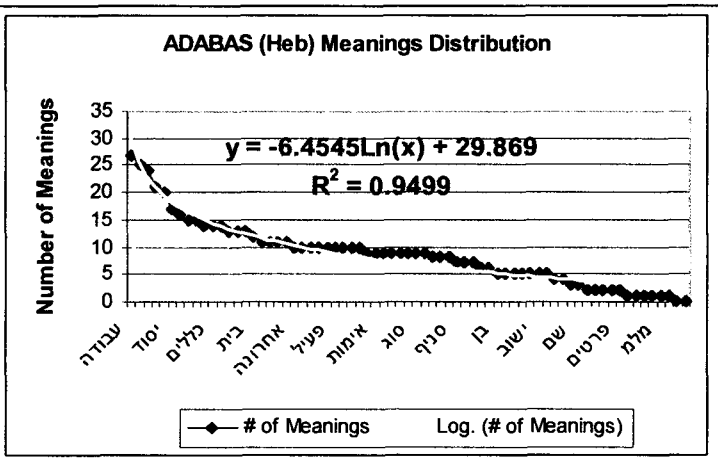


Figure 49: ADABAS (Heb) Word Meaning Distribution

11.4 NCREIF

NCREIF has 112 elements and attributes defined. These are comprised of 271 words after breaking holophrases into single words. It consists of 156 unique words.

11.4.1 NCREIF Word Distribution

The frequency usage of words is summarized in Table 37 on page 171. Here are two examples to help read the table: There are 10 words that appear 10 times in the NCREIF schema. There is one word that is the most frequent, and it appears 105 times. The word is "VALUE" and its usage is shown in Table 38. The next most frequent word is "RATE" and its usage is shown in Table 35.

We graphed the distribution of words and word frequencies for NCREIF shown in Figure 58 and Figure 59 respectively. We calculated a power regression having a correlation coefficient R^2=0.93 for word distribution and a correlation coefficient R^2=0.91 for words frequency distribution.

11.4.2 NCREIF words - Number of Meanings

No.	Word	Meanings
1	Rights	35
2	Base	29
3	Return	29
4	Center	23
5	Free	20
6	Land	20
7	Balance	16
8	Forward	15
9	Name	15
10	Approach	14
11	Life	14
12	Raw	14
13	Real	14

Table 35: NCREIF Meanings

NCREIF has 156 words with a total of 874 meanings. Table 35 has the words with the highest number of meanings. Leading the list is "RIGHTS" with 35 different meanings according to WordNet, followed by "BASE" with 29 meanings. We graph the meanings distribution and calculated a power regression coefficient giving R^2=0.98 as shown in Figure 51. We noted that the most used words in

NCREIF ("Value" and "Rate") are not the ones that have the highest number of meanings. As per WordNet "Value" has 11 meanings and "Rate" has 6 meanings.

11.4.3 NCREIF Entropy Calculations

# of Words	157	
# of Occurrences	275	
H-Maximum	7.2946	$\log_2$ of 157
H-Actual	6.9248	- Sum (P_i * Log_2 P_i)
H-Relative	0.9493	(H-Actual) / (H-Maximum)
Redundancy	0.0507	1 - Relative Entropy

Table 36: NCREIF Entropy

Entropy was calculated according to the formulas presented in the literature review section of Shannon's Information Theory on page 134. NCREIF relative entropy is 0.9493

Frequency	Frequency Count
1	105
2	52
3	33
4	28
5	10
6	6
7	14
8	8
9	9
10	10

Table 37: NCREIF Word Frequency Distribution

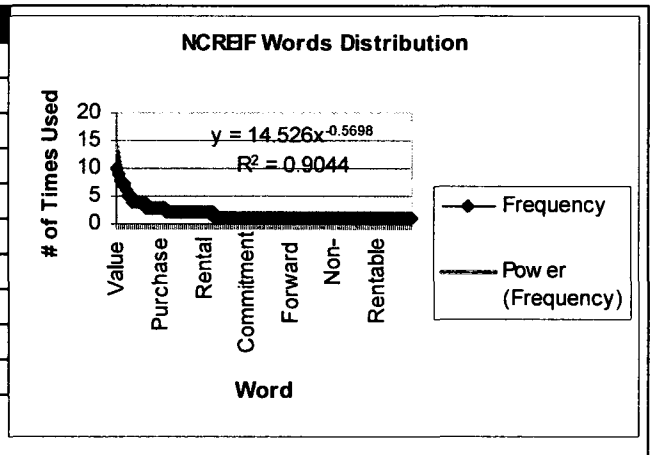


Figure 50: NCREIF Word Distribution

Value (10 times)

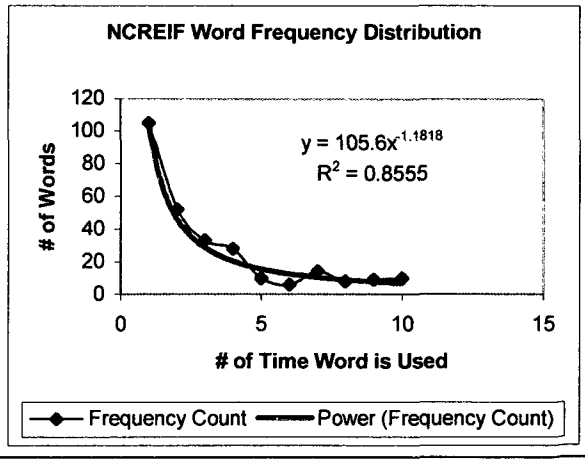
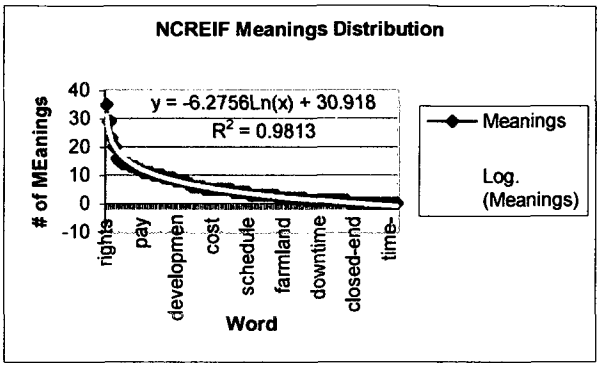


Table 38: NCREIF Usage of the word VALUE

Figure 33: NCREIF Word Frequency Distribution

Rate (9 time)	
Table 39: NCREIF usage of the word RATE	**Figure 51: NCREIF Word Meaning Distribution**

11.5 RETS

RETS has 262 XML elements, comprised of 454 words after breaking holophrases into single words. It consists of 212 unique words.

11.5.1 RETS Words Frequency

The frequency usage of words is summarized in Table 42. Here are two examples to help read the table: There are 141 different words that appear only once in the XML schema. There is only one single word that appears 26 times in the XML schema. This word is "TYPE". See Table 43 for details on how it is being used. The next most frequent word is "re" (most likely an abbreviation for Real Estate) appears 15 times in the text. See Table 44 for details on how it is being used.

We graphed the distribution of words and word frequencies for RETS shown in Figure 52 and Figure 56 respectively. We calculated a power regression having a correlation coefficient R^2=0.91 for the Words Distribution and a correlation coefficient R^2=0.84 for the Words Frequency Distribution.

11.5.2 RETS Words - Number of Meaning

No.	Word	# of Meanings
1	Open	43
2	Close	37
3	Place	32
4	places	32
5	Last	22
6	Start	21
7	change	20
8	Land	20
9	address	18
10	End	18
11	High	18
12	present	18
13	Service	18
14	Fire	17

Table 40: RETS Meanings

RETS has 212 words that yield a total of 329 meanings. Table 40 has the words with the highest number of meanings. "Open" leads the list with 43 different meanings according to WordNet.

We graph the meanings distribution and calculated a power regression and a logarithmic regression coefficient. The power regression line has a very good match with word usage distribution data, having R2=0.92. See Figure 54 for details. The logarithmic regression line has an very good match with the meaning distribution data, having R2=0.97.

We noted that the most used words in RETS are not the ones that have the highest number of meanings. The word used most frequently in the RETS data schema is "TYPE". The number of meanings for "TYPE" is 6 according to WordNet. The second most frequent word in RETS is "RE". It has three meanings according to WordNet. It is of anecdotal interest to name these meanings: (1) rhenium, Re, atomic number 75; (2) Ra, Re -- ancient hawk-headed Egyptian sun god; (3) re, ray -- the syllable naming the second note of the musical scale. None of the three meanings is within the Real Estate domain.

11.5.3 RETS Entropy Calculations

# of Words	212	
# of Occurences	453	
H-Maximum	7.7279	Log2 of 212
H-Actual	7.0612	- Sum (Pi * Log2 Pi)
H-Relative	0.9137	(H-Actual) / (H-Maximum)
Redundancy	0.0863	1 - Relative Entropy

Table 41: RETS Entropy

Entropy for RETS was calculated according to the formulas presented in the literature review section of Shannon's Information Theory on page 134. RETS relative entropy is 0.9137

RETS Frequency Frequency	Count
1	141
2	32
3	15
4	3
5	5
6	2
7	3
8	2
9	1
10	2
11	1
12	3
15	1
26	1

Table 42: RETS Word Frequency Distribution

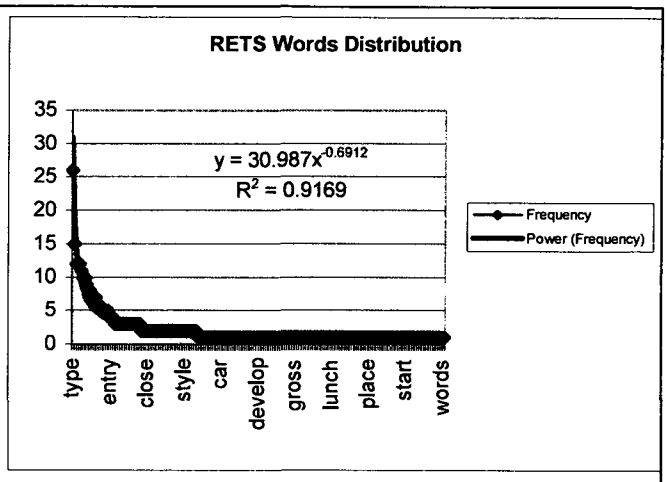


Figure 52: RETS Word \Distribution

Building type
Change type
Document type
Listing type
Ownership type
Transaction type
Type (19 times)
Vestment type

Table 43: RETS usage of the word TYPE

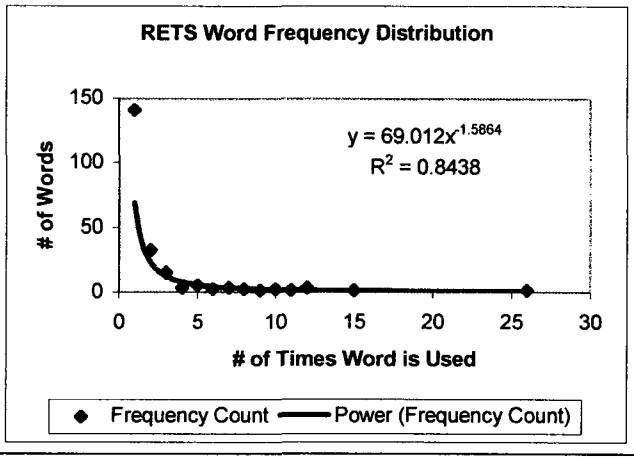


Figure 53: RETS Word Frequency Distribution

REActivities	REAgent
REActivity	REAgents
REData	REOffice
REHistories	REOfficeRosters
REOfficeRosters	REPropEntry
REOffices	REProperties
REPropHistory	REProspects
REProspect	REPublicRecords
RETax	

Table 44: RETS usage of the word RE

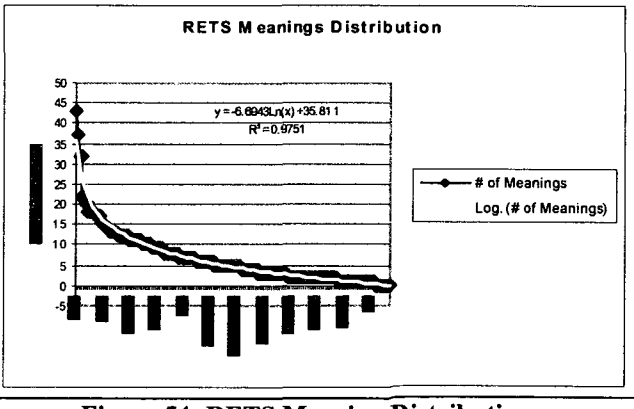


Figure 54: RETS Meaning Distribution

11.6 REPML

REPML has 262 XML elements, comprised of 454 words after breaking holophrases into single words. It consists of 212 unique words.

11.6.1 REPML Words Frequency

The frequency usage of words is summarized in Table 47 on page 178. Here are two examples to help read the table: there are 58 different words that appear twice in the REPML schema. There is one word that is the most frequent, and it appears 13 times. The word is "TYPE" and its usage is shown in Table 48. The next most frequent word is "DATE". It appears 10 times, and its usage is shown in Table 49.

We graphed the distribution of words and word frequencies for REPML, as seen in Figure 55 and Figure 56 respectively. We calculated a power regression for word distribution having R^2=0.80 and a regression coefficient of R^2=0.97 for word frequency distribution.

11.6.2 REPML Words - Number of Meanings

No.	Word	Meanings
1	block	28
2	beds	26
3	balance	16
4	built	15
5	name	15
6	date	13
7	media	13
8	note	13
9	notes	13
10	contract	12
11	title	12
12	value	11
13	price	9

Table 45: REPML Meanings

REPML has 258 words with a total of 390 meanings. Table 45 has the words with the highest number of meanings. Leading the list is "block" with 28 different meanings according to WordNet, followed by "beds" with 26 meanings. We graph the meanings distribution and calculated a power regression coefficient giving R^2=0.97 as shown in Figure 57. We noted that the most used words in

REPML ("type" and "date") are not the ones that have the highest number of meanings. As per WordNet "TYPE" has 8 meanings zand "DATE" has 13 meanings.

11.6.3 REPML Entropy Calculations

Entropy was calculated according to the formulas presented in the literature review section of Shannon's Information Theory on page 134. REPML relative entropy is 0.9567

# of Words	116	
# of Occurences	258	
H-Maximum	6.8580	log2 of 116
H-Actual	6.5611	- Sum (Pi * Log2 Pi)
H-Relative	0.9567	(H-Actual) / (H-Maximum)
Redundancy	0.0433	1 - Relative Entropy

Table 46: REPML Entropy

Frequency	Frequency count
13	1
10	1
7	1
6	3
5	1
4	10
3	4
2	58

Table 47: REPML Word Frequency

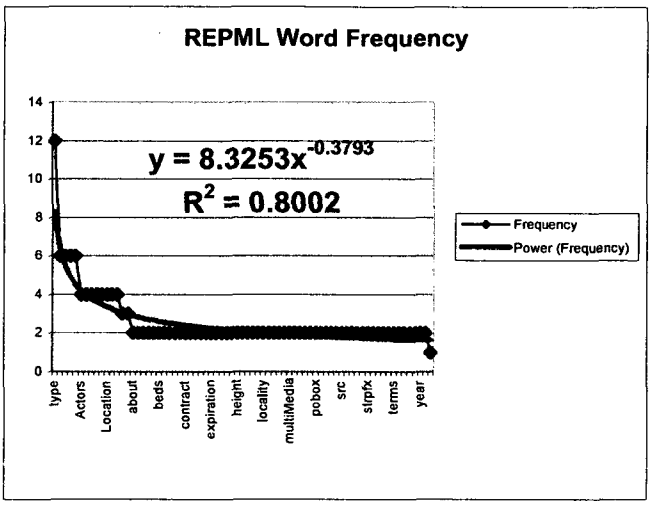


Figure 55: REPML Word Distribution

Type (11 times)
Subtype (2 times)

Table 48: REPML usage of "TYPE"

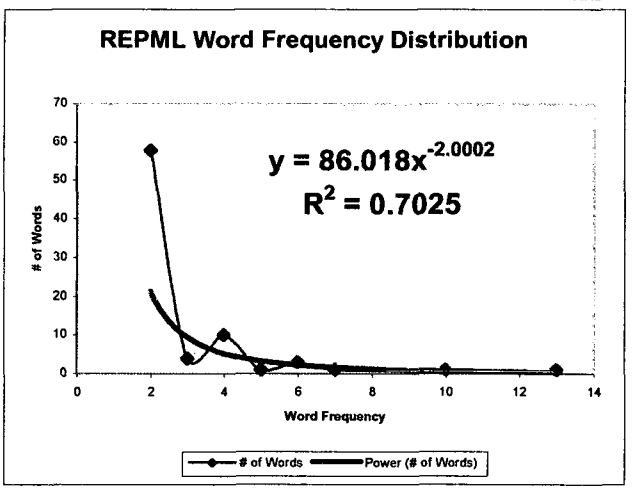


Figure 56: REPML Word Frequency Distribution

DateTime (4 times)
ContractDate (2 times)
ExpirationDate (2 times)
originalListingDate (2 times)

Table 49: REPML usage of the word DATE

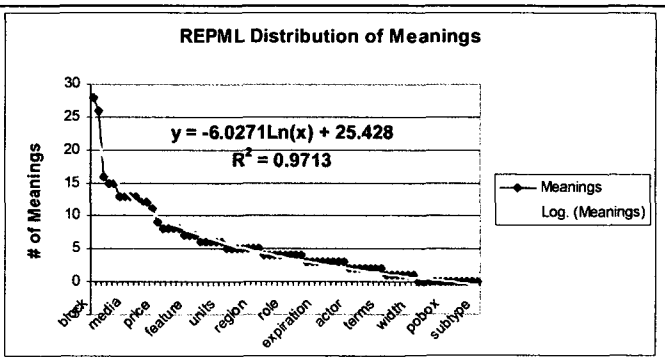


Figure 57: REPML Meaning Distribution

11.7 MFDX

MFDX has 138 XML elements, comprised of 231 words after breaking holophrases into single words. It consists of 105 unique words.

11.7.1 MFDX Words Frequency

The frequency usage of words is summarized in Table 52. Here are two examples to help read the table: there are 63 different words that appear only once in the MFDX schema. There is one word that appears 20 times in the MFDX schema. This word is "ID". It appears 23 times. Table 53 details how it is being used. The next most frequent word is "ADDRESS" is used 10 times in the MFDX data schema, as shown in Table 54.

We graphed the distribution of words and word frequencies for MFDX shown in Figure 58 and Figure 59 respectively. We calculated a power regression having a correlation coefficient R^2=0.93 for the Words Distribution and a correlation coefficient R^2=0.91 for the Words Frequency Distribution.

11.7.2 MFDX Words - Number of Meanings

MFDX has 105 unique words with a total of 882 meanings. Table 50 has the words with the highest number of meanings. Leading the list is "open" with 43 different meanings, followed by "call" with 41 different meanings. We graph the meanings distribution and calculated a power regression coefficient giving R^2=0.97 as shown in Figure 60. We noted that the most used words in MFDX ("ID" and "Address) are not the ones that have the highest number of meanings. As per WordNet "ID" has 3 meanings and "ADDRESS" has 18 distinct meanings.

No.	Word	Meanings
1	Open	43
2	Call	41
3	charge	40
4	close	37
5	Line	35
6	Short	28
7	Square	25
8	Center	23
9	Last	22
10	Active	19
11	Address	18
12	First	17
13	Home	17

Table 50: MFDX Meanings

11.7.3 MFDX Entropy Calculations

Entropy for MFDX was calculated according to the formulas presented in the literature review section of Shannon's Information Theory on page 134. MFDX relative entropy is 0.9163

# of Words	105	
# of Occurences	231	
H-Maximum	6.7142	$\log_2$ of 105
H-Actual	6.1524	- Sum (P_i * Log_2 P_i)
H-Relative	0.9163	(H-Actual) / (H-Maximum)
Redundancy	0.0837	1 - Relative Entropy

Table 51: MFDX Entropy

MFDX Frequency	Frequency Count
1	63
2	17
3	11
4	3
5	3
6	3
8	2
10	2
20	1

Table 52: : MDFX Word Distribution

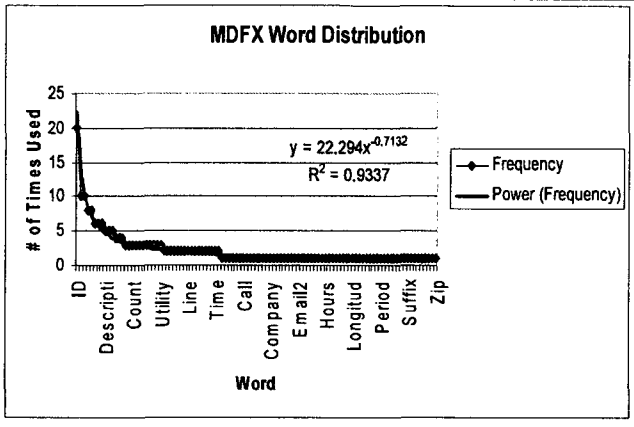


Figure 58: MDFX Word Distribution

AdTypeID
AmenityID
DayID
FileTypeID
ID (3 times)
LeasePeriodID
ParkingTypeID
ParkingTypeID
PetTypeID
PropertyTypeID
RDTID (2 times)
RefID (4 times)
SchoolTypeID
SpecialAccessID
UtilityID
VacancyClassID

Table 53: MDFX usage of "ID"

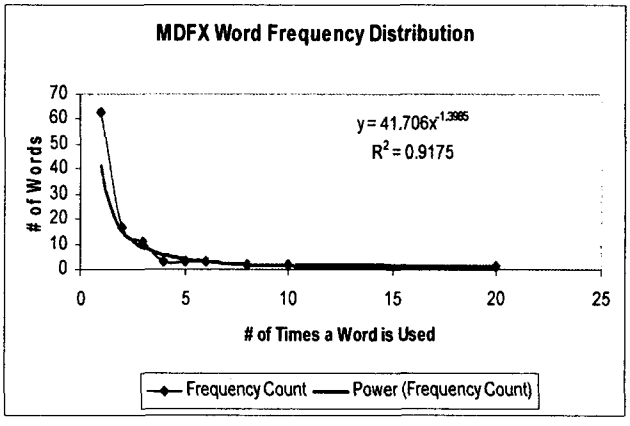


Figure 59: MFDX Word Frequency Distribution

Address (4 times)
AddressLine
AddressLine1
AddressLine2
AddressLine3
AddressLine4
AddressType

Table 54: : MDFX usage of the word "ADDRESS"

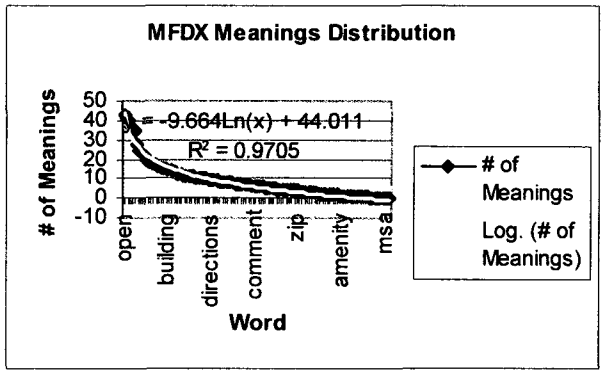


Figure 60: MFDX Meaning Distribution

11.8 REXML

REXML has 1862 XML elements and attributes defined comprised of 3832 words after breaking holophrases into single words. It consists of 323 unique words

11.8.1 REXML Words Frequency

The frequency usage of words is summarized in Figure 61. Here are several examples to help read the table: there are 91 different words that appear only once in the REXML schema. There are 54 words that appear 2 times in the REXML schema. There is one word that is the most frequent, and it appears 163 times. The word is "DATE" and its usage is shown in Table 56. The next most frequent word is "REFERENCE". It is used 160 times and its usage is shown in Table 57.

We graphed the distribution of words and the distribution word frequencies for REXML shown in Figure 61 and Figure 62 respectively. We calculated a power regression having a correlation coefficient R^2=0.93 for the Words Distribution and a correlation coefficient R^2=0.69 for the Words Frequency Distribution.

Frequency	Frequency Count
1	91
2	54
3	30
4	23
5	8
6	14
7	8
8	6
9	13
10	5
11	1
12	5
13	3
14	1
15	1
16	3
18	3
19	4
20	1

Frequency	Frequency Count
21	5
22	1
23	1
24	1
25	2
26	2
28	2
29	1
30	1
31	1
34	1
37	1
38	1
39	1
40	4
41	1
42	1
43	1
44	1

Frequency	Frequency Count
46	1
47	2
49	1
50	1
58	1
66	1
69	1
75	1
77	1
91	1
92	1
105	1
115	1
120	2
121	1
148	1
160	1
163	1

Table 55: REPML Word Frequency Distribution

EffectiveDate (108 times)	AmortizeStartDate (1 times)
EndDate (12 times)	BalloonDate (1 times)
ReviewDate (12 times)	BudgetBeginDate (1 times)
BaseRentStepsEffectiveDate (6 times)	DateOfSale (1 times)
StartDate (6 times)	HistoricalFinancialsBeginDate (1 times)
BeginDate (5 times)	HistoricalTenantSalesBeginDate (1 times)
AvailableDate (2 times)	MaturityDate (1 times)
BeginLeasingDate (2 times)	ProjectionBeginDate (1 times)
OriginalLeaseStartDate (2 times)	

Table 56: REXML usage of the word "Date"

InflationReference (14 times)	RecoveryReferences (4 times)
LeaseReference (14 times)	TenantImprovementReference (4 times)
PartnerReference (14 times)	CashFlowDistributionReference (3 times)
RecoveryReference (12 times)	MonthsVacantReference (3 times)
BudgetAccountReference (9 times)	PriorCashFlowDistributionReference (3 times)
MarketRentReference (8 times)	RecoveryPoolReference (3 times)
SalesVolumeReference (7 times)	WhenPaidReference (3 times)
AreaReference (6 times)	FreeRentReference (2 times)
BreakpointReference (6 times)	NonContiguousLeaseReference (2 times)
SalesSeasonalityReference (6 times)	ParentLeaseReference (2 times)
PreferenceLevel (5 times)	ReferenceAccount (2 times)
RenewalProbabilityReference (5 times)	ResaleDistributionReference (2 times)
CreditLossReference (4 times)	ChartReference (1 times)
GeneralVacancyReference (4 times)	DebtNoteReference (1 times)
LeasingCommissionReference (4 times)	FreeRentReferenceType (1 times)
MarketLeasingAssumptionReference (4 times)	PriorBalloonDebtNoteReference (1 times)
ParentAccountReference (4 times)	PropertyReference (1 times)
RecoveryChargeReference (4 times)	SeasonalityReference (1 times)

Table 57: REXML usage of the word "Reference"

11.8.2 REXML Words - Number of Meanings

REXML has 323 words with a total of 2043 meanings. Table 58 has the words with the highest number of meanings. Leading the list is "HOLD" with 45 different meanings according to WordNet, followed by "CHARGE" with 40 meanings. We graph the meanings distribution and calculated a power regression coefficient giving R^2=0.98 as shown in Figure 63. We noted that the most used words in REXML ("Date" and "Reference") are not the ones that have the highest number of meanings. As per WordNet "Date" has 13 meanings and "Reference" has 10 meanings.

No.	Word	Meanings
1	Hold	45
2	Charge	40
3	Line	35
4	Based	33
5	Ground	32
6	Following	30
7	Base	29
8	Direct	25
9	Spread	23
10	Steps	23
11	Stop	22
12	Hard	21
13	Start	21

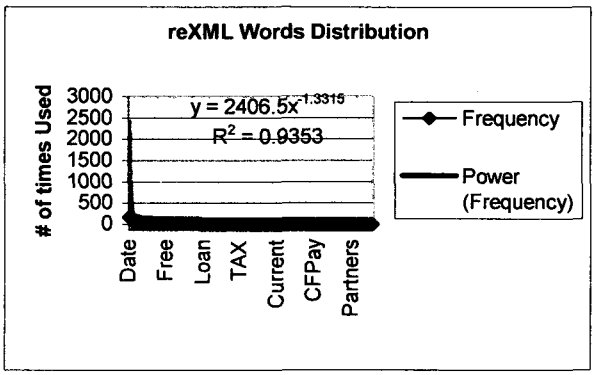
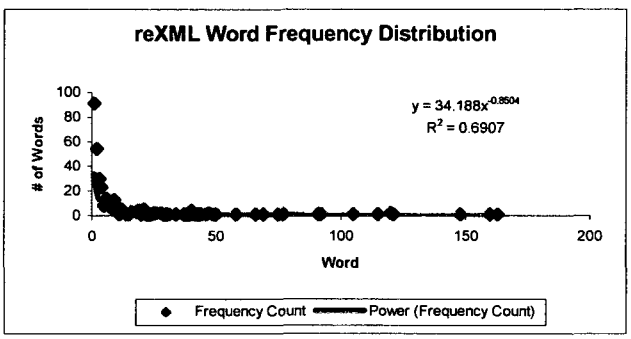
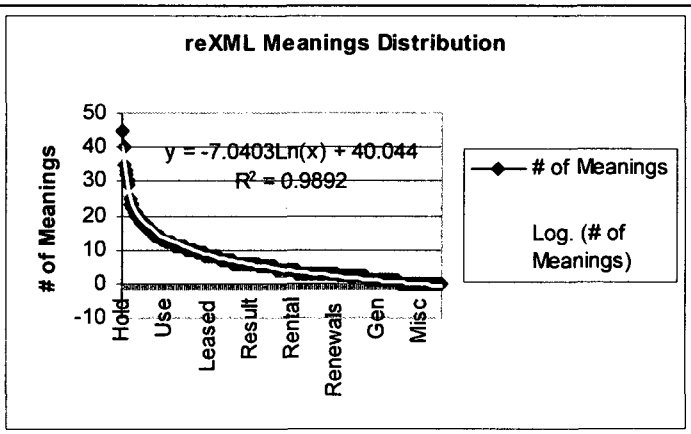
Table 58: REXML Meanings

11.8.3 REXML Entropy

Entropy for REXML was calculated according to the formulas presented in the literature review section of Shannon's Information Theory on page 134. Its relative entropy is 0.8182

# of Words	323	
# of Occurrences	3832	
H-Maximum	8.3354	$\log_2$ of 323
H-Actual	6.8201	Sum (P_i * Log_2 P_i)
H-Relative	0.8182	(H-Actual) / (H-Maximum)
Redundancy	0.1818	1 - Relative Entropy

Table 59: REXML Entropy

Please refer to Table 56 on page 184 for details	
Table 60: ADABAS (Heb) Words Frequency	**Figure 61: reXML Word Distribution**
Please refer to Table 56 on 184 for details	
Table 61: ADABAS (Heb) Usage of "KOD"	**Figure 62: reXML Word Frequency Distribution**
Please refer to Table 57 on page 184 for details	
Table 62: ADABAS (Heb) Usage of "ISKA"	**Figure 63: reXML Meanings Distribution**

11.9 MISMO

MISMO Mortgage Application expressed in XSD has 138 elements and attributes defined. These are comprised of 232 words after breaking holophrases into single words. It consists of 105 unique words.

11.9.1 MISMO Word Distribution

The frequency usage of words in MISMO is summarized in Table 65 on page 189. Here are two examples to help read the table: there are 63 different words that appear only once in the MISMO schema. There are 20 words that appear 20 times in the MISMO schema. There is one word that is the most frequent, and it appears 20 times. The word is "ID" and its usage is shown in Table 66. The next most frequent word is "ADDRESS" and its usage is shown in Table 67.

We graphed the distribution of words and word frequencies for MISMO shown in Figure 64 and Figure 65 respectively. We calculated a power regression for word frequency that resulted in a correlation coefficient R^2=0.93 for the Words Distribution and a correlation coefficient R^2=0.91 for the Words Frequency Distribution.

11.9.2 MISMO Words - Number of Meanings

MA Word	# of Meanings
Open	43
Call	41
Charge	40
Close	37
Line	35
Short	28
Square	25
Center	23
Last	22
Active	19
Address	18
First	17
Home	17

Table 63: MISMO Meanings

MISMO has 105 words with a total of 882 meanings. Table 63 has the words with the highest number of meanings. Leading the list is "OPEN" with 43 different meanings according to WordNet, followed by "CALL" with 41 meanings. It is of interest to note that MISMO has 9 words with no meaning in WordNet. Two of them are "MSA" and "RDTID". We graph the meanings distribution and calculated a power regression coefficient giving R^2=0.97 as shown in Figure 66. We noted that the most used words in MISMO ("ID" and "Address") are not the ones that have the highest number of meanings. As per WordNet "ID" has 3 meanings and "Address" has 18 meanings.

11.9.3 MISMO Entropy Calculation

# of Words	105	
# of Occurences	231	
H-Maximum	6.7142	$\log_2$ of 105
H-Actual	6.1524	- Sum (P_i * Log_2 P_i)
H-Relative	0.9163	(H-Actual) / (H-Maximum)
Redundancy	0.0837	1 - Relative Entropy

Table 64: MISMO Entropy

Entropy for MISMO was calculated according to the formulas presented in the literature review section of Information Theory on page 134. The relative entropy is 0.9163

Frequency	Frequency Count
1	63
2	17
3	11
4	3
5	3
6	3
8	2
10	2
20	1

Table 65: MISMO Word Frequency Distribution

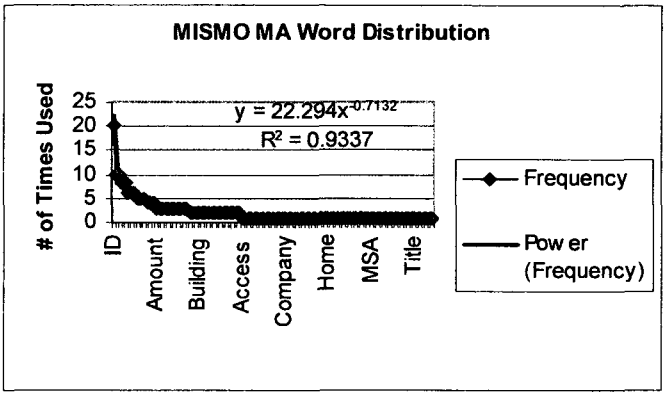


Figure 64: MISMO Word Distribution

RefID (4 times)
AmenityID
PropertyTypeID
ID (3 times)
DayID
SchoolTypeID
ParkingTypeID (2 times)
FileTypeID
SpecialAccessID
RDTID (2 times)
LeasePeriodID
UtilityID
AdTypeID
PetTypeID
VacancyClassID

Table 66: MISMO usage of the word "ID"

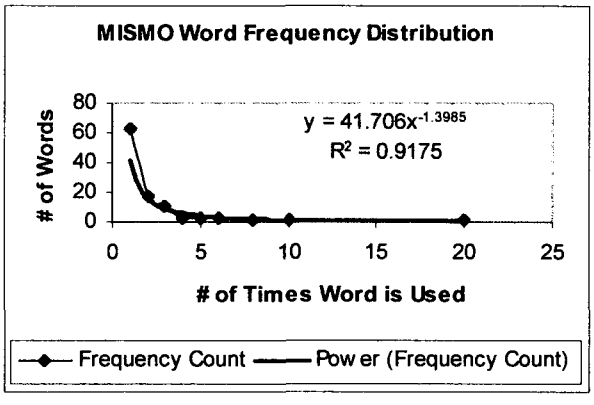


Figure 65: MISMO Word Frequency Distribution

Address (4 times)
AddressLine
AddressLine1
AddressLine2
AddressLine3
AddressLine4
AddressType

Table 67: MISMO usage of the word "ADDRESS"

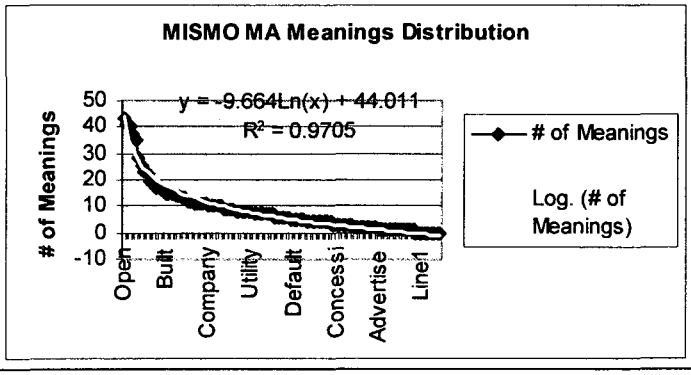


Figure 66: MISMO MA Word Meanings Distribution

11.10 HARMONIZE

HARMONIZE Travel Ontology expressed in OWL has 912 triplets. These are comprised of 900 words after breaking holophrases into single words. It consists of 175 unique words.

11.10.1 HARMONIZE Word Distribution

The frequency usage of words in HARMONIZE is summarized in Table 70 on page 192. Here are two examples to help read the table: there are 72 different words that appear four times each in the HARMONIZE schema. There is one word that is the most frequent, and it appears 48 times. The word is "TO" and its usage is shown in Table 71. The next most frequent word is "DATE" and its usage is shown in Table 72.

We graphed the distribution of words and word frequencies for HARMONIZE shown in Figure 67 and Figure 68 respectively. We calculated a power regression for word frequency that resulted in a correlation coefficient R^2=0.90 for the Words Distribution and a correlation coefficient R^2=0.30 for the Words Frequency Distribution.

11.10.2 HARMONIZE Words - Number of Meanings

Harmonize Word	# of Meanings
open	43
point	37
line	35
post	23
support	22
field	21
start	21
free	20
level	19
address	18
end	18
position	18
service	18

Table 68: HARMONIZE Meanings

HARMONIZE has 176 words with a total of 1104 meanings. Table 68 has the words with the highest number of meanings. Leading the list is "OPEN" with 43 different meanings according to WordNet, followed by "POINT" with 37 meanings. It is of interest to note that HARMONIZE has 21 words with no meaning in WordNet. We graph the meanings distribution and calculated a power regression coefficient giving R2=0.97 as shown in Figure 69. We noted that the most used words in HARMONIZE ("TO" and "DATE") are not the ones that have the highest number of meanings. As per WordNet "TO" has 0 meanings and "DATE" has 13 meanings.

11.10.3 HARMONIZE Entropy Calculation

Entropy for HARMONIZE was calculated according to the formulas presented in the literature review section of Information Theory on page 134. The relative entropy of HARMONIZE is 0.2795

# of Words	175	
# of Occurrences	900	
H-Maximum	7.4512	$\log_2$ of 105
H-Actual	2.0827	- Sum (P_i * Log_2 P_i)
H-Relative	0.2795	(H-Actual) / (H-Maximum)
Redundancy	0.7205	1 - Relative Entropy

Table 69: Harmonise Entropy

FREQ	Frequency Count	FREQ	Frequency Count
3	10	15	2
4	72	16	3
5	21	17	1
6	10	18	1
7	1	20	2
8	16	21	2
9	11	22	5
10	2	24	2
11	2	31	1
12	4	42	1
13	4	47	1
14	1	48	1

Table 70: HARMONIZE Word Frequency Distribution

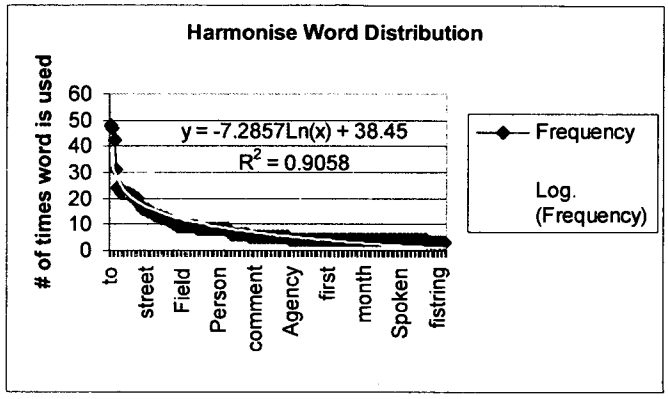


Figure 67: HARMONIZE Word Distribution

belongsTo(23)
linksTo (5)
relatedTo (7)
toPoint (4)
toTime (4)

Table 71: HARMONIZE usage of the word "TO"

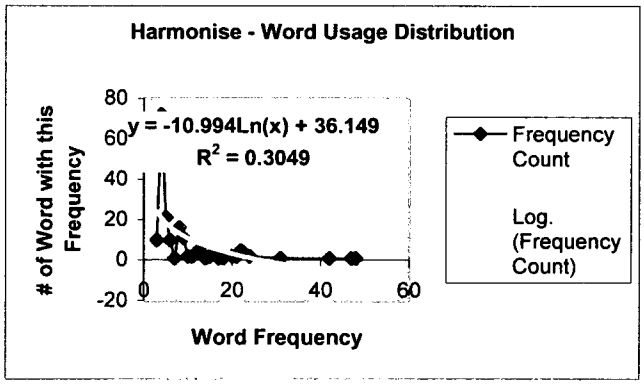


Figure 68: HARMONIZE Word Frequency Distribution

date (8)
creationDate (5)
datesOpenList (5)
dateText (5)
validityDateRange (5)
dateAwardAchieved (4)
DateList (4)
DateRange (4)
datesClosed (4)
datesOpen (4)
endDate (4)
memberDate (4)
startDate (4)

Table 72: HARMONIZE usage of the word "DATE"

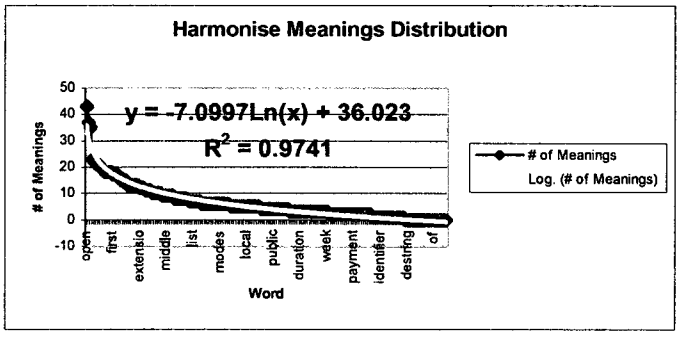


Figure 69: HARMONIZE Word Meanings Distribution

11.11 TAGA

TAGA has 83 proposition elements, comprised of 126 words after breaking holophrases into single words. It consists of 33 unique words.

11.11.1 TAGA Words Frequency

The frequency usage of words is summarized in Table 75. Here are two examples to help read the table: There are 18 different words that appear only once in the OWL schema. There is only one single word that appears 18 times in the OWL schema. This word is "RESERVATION". See Table 76 for details on how it is being used. The next most frequent word is "PREFERENCE" and it appears 16 times in the text. See Table 77 for details on how it is being used.

We graphed the distribution of words and word frequencies for TAGA shown in Figure 70 and Figure 71 respectively. We calculated a power regression having a correlation coefficient R^2=0.889 for the Words Distribution and a correlation coefficient R^2=0.542 for the Words Frequency Distribution.

11.11.2 TAGA Words - Number of Meanings

TAGA has 33 unique words that yield a total of 225 meanings as seen on Table 73. "RETURN" leads the list with 29 different meanings according to WordNet.

Word	# of Meanings
return	29
has	20
Service	18
Number	17
offer	16
Name	15
reserve	11
Value	11
travel	9
date	8
Type	8
Plan	7
price	7
Reservation	7

Agent	6
penalty	4
prefer	4
Preference	4
departure	3
Id	3
Itinerary	3
Ready	3
Resource	3
Airline	2
by	2
Provider	2
Customer	1
Entertainment	1
Hotel	1

Table 73: TAGA Distribution of Meanings

We graph the meanings distribution and calculated an exponential regression that yields a correlation coefficient R^2=0.98. See Figure 72 for details.

We noted that the most used words in TAGA are not the ones that have the highest number of meanings. The word used most frequently in the TAGA data schema is "RSERVATION". The number of meanings for "RSERVATION" is 7 according to WordNet. The second most frequent word in TAGA is "PREFERENCE". It has 4 meanings according to WordNet.

11.11.3 TAGA Entropy Calculations

# of Words	33	
# of Occurences	126	
H-Maximum	5.0444	log2 of 116
H-Actual	4.2793	- Sum (Pi * Log2 Pi)
H-Relative	0.8483	(H-Actual) / (H-Maximum)
Redundancy	0.1517	1 - Relative Entropy

Table 74: TAGA Entropy

Entropy for TAGA was calculated according to the formulas presented in the literature review section of Information Theory on page 134. TAGA relative entropy is 0.848

Frequency	Frequency Count
1	18
2	3
3	1
5	1
6	5
7	1
10	1
13	1
16	1
18	1

Table 75: TAGA Word Frequency

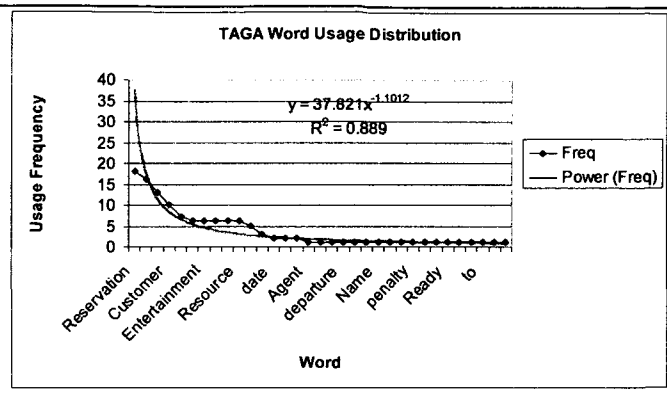


Figure 70: TAGA Word Distribution

Reservation (8)
AirlineReservation (3)
EntertainmentReservation (3)
HotelReservation (3)
HasReservation (1)

Table 76: TAGA Usage of the word Reservation

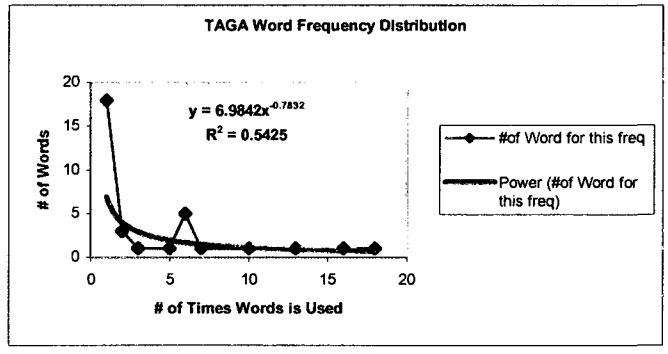


Figure 71: TAGA Frequency Distribution

Preference (6)
EntertainmentPreference (3)
HotelPreference (3)
AirlinePreference (3)
HasPreference (1)

Table 77: TAGA Usage of the word Preference

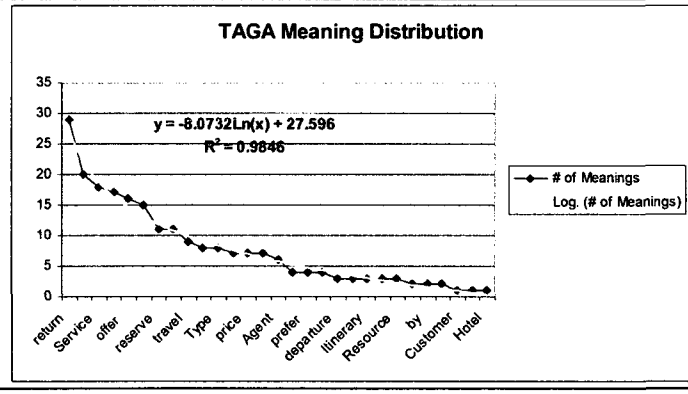


Figure 72: TAGA Meaning Distribution

11.12 EDI

EDI is expressed in proprietary flat file message types. We evaluated the EDIFACT Version D.99B messages. There are 165 messages in this EDI group. Each message has its own identifier, an arbitrary 5 letters sequence. None of the identifiers is a word in English.

11.12.1 EDI Word Distribution

Every "word" in the EDI schema appears only once. Refer to Figure 73 for graphical representation of the distribution.

11.12.2 EDI Words - Number of Meanings

Meaning for EDI "words" are available from the EDI standard, not from WordNet. Each "word" corresponds to one and only one message type. Ambiguity is not a factor.

11.12.3 EDI Entropy Calculation

Entropy for EDI was calculated according to the formulas presented in the literature review section of Information Theory on page 134. The relative entropy for EDI is 1.00

# of Words	166
# of Occurences	166
H-Maximum	7.3750Log2 of 166
H-Actual	7.3750 - Sum (Pi * Log2 Pi)
H-Relative	1.0000(H-Actual) / (H-Maximum)
Redundancy	0.00001 - Relative Entropy

Table 78: EDI Entropy

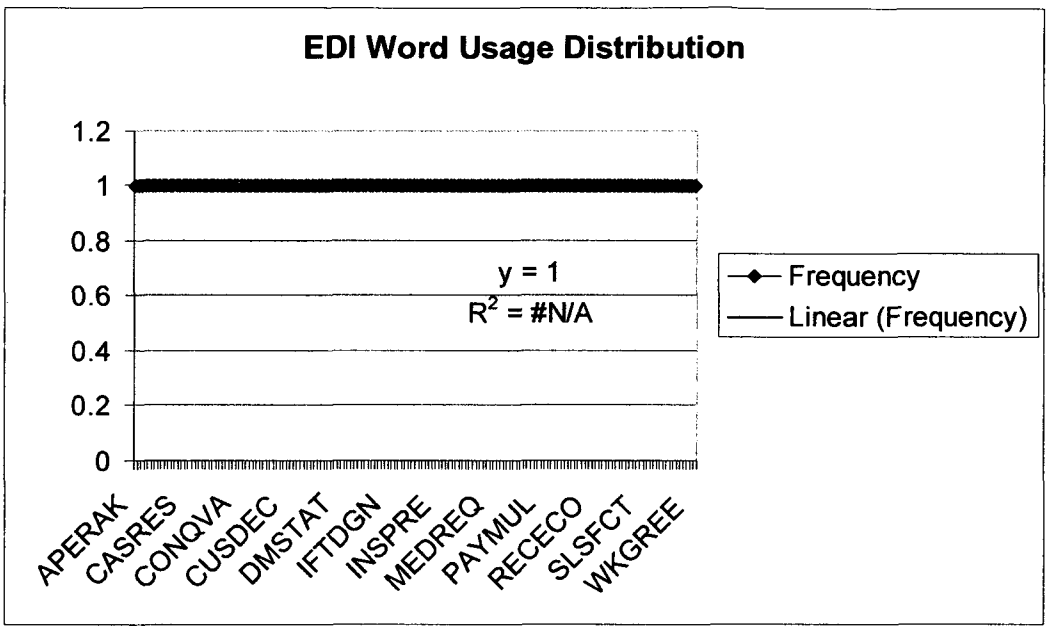


Figure 73: EDI Word Distribution

11.13 SWIFT

SWIFT is expressed in proprietary flat file message types. We evaluated the SWIFT Category 5 Messages (Securities Markets MT568 - MT599) and Financial Institution Transfers Messages (MT200 – MT293). There are 38 messages altogether in these SWIFT group. Each message has its own identifier, a 5 alphanumeric sequence. None of the identifiers is a word in English.

11.13.1 SWIFT Word Distribution

Every "word" in the SWIFT schema appears only once. Refer to Figure 74 for graphical representation of the distribution.

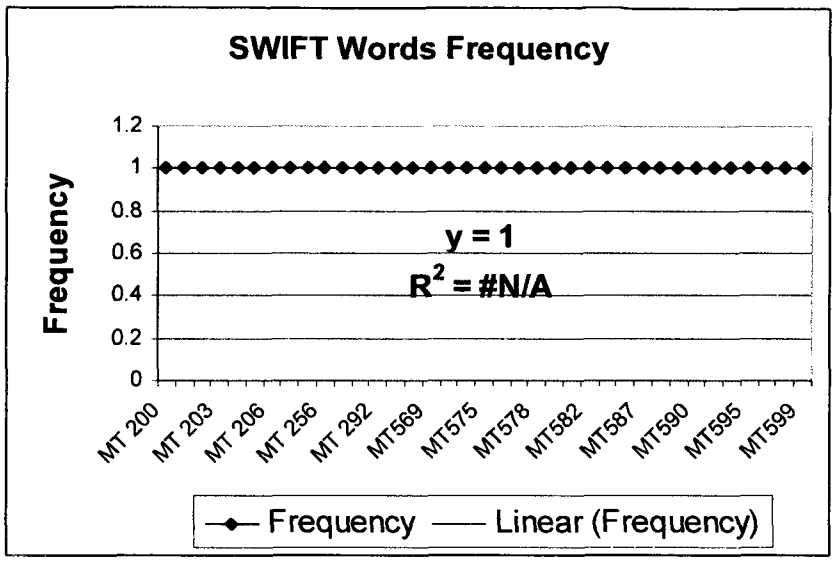


Figure 74: SWIFT Word Distribution

11.13.2 SWIFT Words - Number of Meanings

Meaning for SWIFT "words" are available from the SWIFT standard, not from WordNet. Each "word" corresponds to one and only one message type. Ambiguity is not a factor. See Figure 75 for graphical representation of this data.

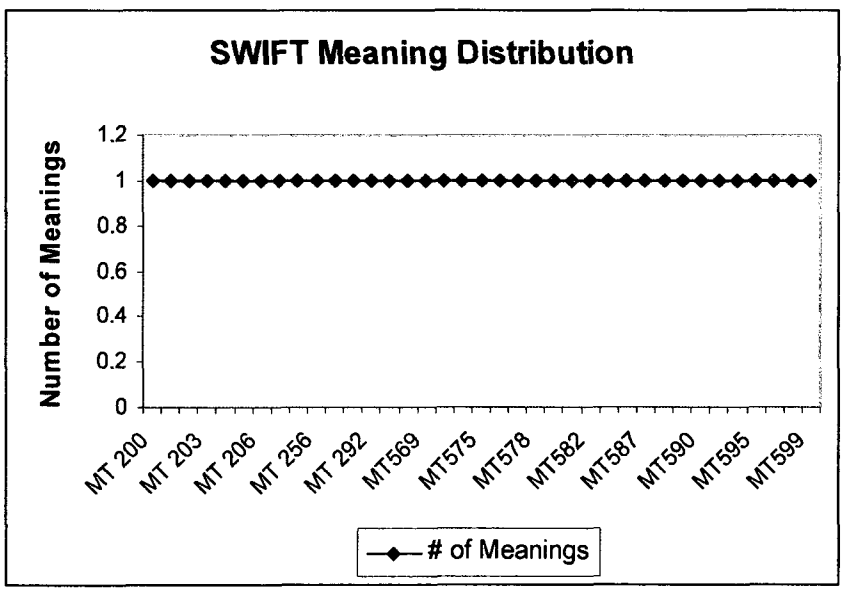


Figure 75: SWIFT Meaning Distribution

11.13.3 SWIFT Entropy Calculation

Entropy for SWIFT was calculated according to the formulas presented in the literature review section of Information Theory on page 134. The relative entropy for SWIFT is 1.00

# of Words	37	
# of Occurences	37	
H-Maximum	5.2095	log2 of 37
H-Actual	5.2095	- Sum (Pi * Log2 Pi)
H-Relative	1.0000	(H-Actual) / (H-Maximum)
Redundancy	0.0000	1 - Relative Entropy

Table 79: SWIFT Entropy

11.14 Chapter 11 Summary and Implications to the Research

Using the multi-disciplinary methodology described earlier, this chapter gives a detailed report of measured CAS properties in multiple DDLs: variety, tension and entropy. For each schema in the sample, every CAS property is described and measured using an established approach, taken from a scientific discipline relevant to the property of interest. The data generated here is used in subsequent chapters for analysis and for drawing conclusions.

Implication of chapter to study	How implication will be used to draw conclusions
Alphabet creation	Extraction of signifiers out from the DDL syntax (constraints) for the purpose of measuring CAS related attributes of interest. Most signifiers are expressed in natural language, therefore the "alphabet" is usually a list of words in English or Hebrew.
Signifiers distribution calculation	Quantify the distribution of signifiers, for later comparison with other DDLs, perform analysis and draw conclusions.
Signifiers meaning distributions	Signifiers may be ambiguous; the study quantifies the distribution of meanings in each DDL, for later comparison with other DDLs, perform analysis and draw conclusions.
Meaning preservation determination	Determines if a given DDL has the necessary constraints to support, as a minimum, bijective mapping, for later comparison with other DDLs, perform analysis and draw conclusions.
Entropy calculation	Determine the degree of "order" in each DDL for later comparison with other DDLs, perform analysis and draw conclusions.

Table 80: Summary of Chapter 11 Implications to the Study

CHAPTER 12

DATA ANALYSIS

This chapter builds on the individual measures obtained from the thirteen data schemas as described in chapter 11. Analysis of individual data schemas at a more abstract level leads to the discovery of patterns and their in-depth examination. The chapter starts with a juxtaposed simple summary of all data schemas, describing the number of elements, words, unique words and meanings in each. Then the methodology established in chapter 10 is used on that summary, leading to the discovery of patterns not seen in the vast literature reviewed for this research. Those patterns indicate that not only CAS attributes exist among data structures, but they have some hidden internal organization which resembles many naturally occurring phenomena that have the Zipf distribution (e.g., earthquake magnitudes, city sizes, income). Attention is then shifted to aspected directly related to the first two research questions posed at the outset. *Variety,* a key CAS construct, is addressed by evaluation of internal variety and external variety through the analysis on Zipf distributions of words used and number of meanings. Increased support (or lack thereof) for *Tension* is analyzed by longitudinal review of meaning preservation characteristics in the various DDLs. Finally, *Entropy* in DDLs is compared between all data structures and analyzed for variation in time. The chapter ends a summary of findings and sets the stage for the next two and final steps– drawing conclusions and making recommendations.

12.1 Additional Observations of Power Distributions

Table 81 summarizes the *Variety* found in the environment examined for this research: number of data sources, number of data elements, number of words, number of unique words and number of meanings present in the sample data used for this research.

Abbreviated Source Name	DDL	# of Data Elements	# of Words	# of Unique Words	# of Meanings
NCREIF	NL	112	271	156	874
ADABAS (Heb)	NL	82	193	68	601
COBOL (Eng)	COBOL	57	333	49	308
COBOL (Heb)	COBOL	73	167	77	491
EDI	Proprietary	166	165	165	165
SWIFT	Proprietary	37	37	37	37
REPML	XML	262	454	212	390
MFDX	XML	138	231	105	882
RETS	DTD	454	452	212	1385
MISMO	DTD	138	232	105	882
REXML	XSD	1862	3832	323	2043
HARMONIZE	RDF	912	900	175	1104
TAGA	OWL	83	126	33	225

Table 81: Summary of Raw Data

Table 82 summarizes the correlations coefficients of each data structure relating to Word Frequency distribution, Word Usage distribution and Word Meanings distributions. These distributions are based on Zipf's approach to counting words and graphing the results. This aspect is important for addressing the second research question: do new DDLs progressively meet the theoretical requirements spelled out in the proposed data integration theory?

Abbreviated Source Name	DDL	Word Frequency Distribution Power Correlation	Word Distribution Power Correlation	Word Meaning Distribution Logarithmic Correlation
NCREIF	NL	R^2=0.8555	R^2=0.9044	R^2=0.9813
ADABAS (Heb)	NL	R^2=0.6568	R^2=0.9216	R^2=0.9499
COBOL (Eng)	COBOL	R^2=0.6418	R^2=0.9187	R^2=0.9778
COBOL (Heb)	COBOL	R^2=0.9085	R^2=0.9112	R^2=0.8955
EDI	Proprietary	Not applicable	Not applicable	Not applicable
SWIFT	Proprietary	Not applicable	Not applicable	Not applicable
REPML	XML	R^2=0.6016	R^2=0.8002	R^2=0.9713
MFDX	XML	R^2=0.9175	R^2=0.9337	R^2=0.9705
RETS	DTD	R^2=0.8438	R^2=0.9169	R^2=0.9751
MISMO	DTD	R^2=0.9175	R^2=0.9937	R^2=0.9705
REXML	XSD	R^2=0.6907	R^2=0.9353	R^2=0.9892
HARMONIZE	OWL	R^2=0.3049	R^2=0.9058	R^2=0.9741
TAGA	RDF	R^2=0.5425	R^2=0.9230	R^2=0.9846

Table 82: Correlations Coefficients Summary

Graphing some of the data in Table 81 and in Table 82 reveal concealed power distributions. Data for each distribution is described and graphed in Table 83 through Table 86.

Table 83 lists all DDL data sources sorted by the number of data elements present in each such data source. The data was graphed and a correlation coefficient was calculated. The correlation is surprisingly gets very close perfect "1".

Source Name	# Data Elements
REXML	1862
HARMONIZE	912
RETS	454
REPML	262
EDI	165
MISMO	138
MFDX	138
NCREIF	112
TAGA	83
ADABAS (Heb)	82
COBOL (Heb)	73
COBOL (Eng)	57
SWIFT	37

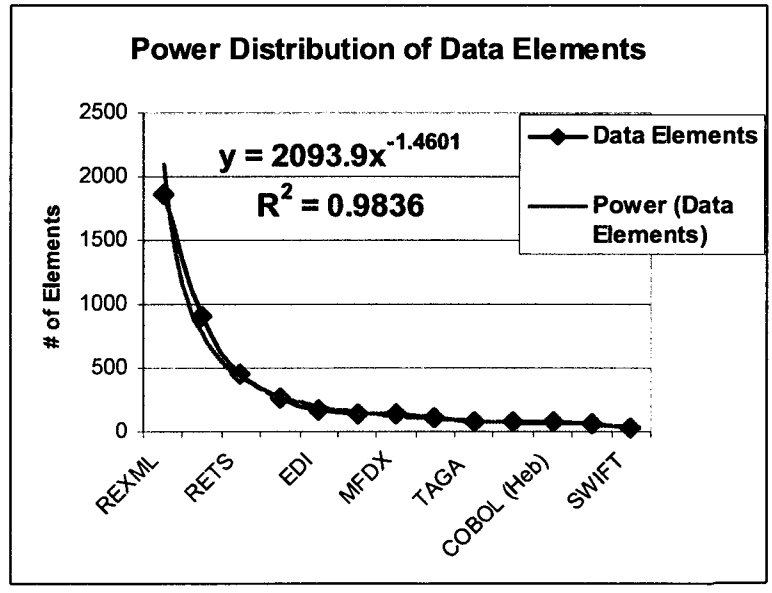


Table 83: Number of Data Elements Distribution

Table 84 lists all DDL data sources sorted by the ratio of Words to Data elements in each such data source. That is, we count the total number of words used in every given DDL representative. Then we count the number of data elements in each DDL. The ratio is the quotient resulting from the division of the total number of words by the number of data elements, for each DDL representative. We graphed the data and a calculated correlation coefficient. The correlation is 0.91.

Source Name	No. of Words / No. of Elements
COBOL (Eng)	5.842
NCREIF	2.420
ADABAS (Heb)	2.354
COBOL (Heb)	2.288
REXML	2.058
REPML	1.733
MISMO	1.681
MFDX	1.674
TAGA	1.518
EDI	1.000
SWIFT	1.000
RETS	0.996
HARMONIZE	0.987

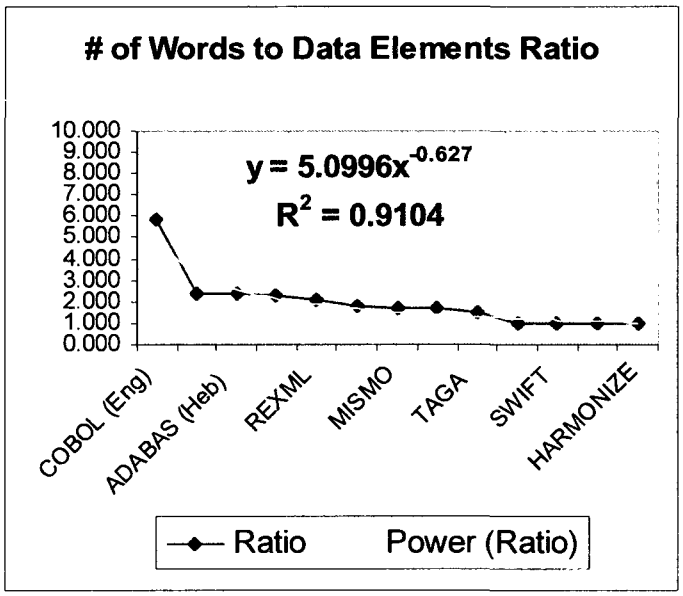


Table 84: Number of Words to Data Elements Ratio

Table 85 lists all DDL data sources sorted by the ratio of Words to Unique Words in each such data source. That is, we count the total number of words used in every given DDL representative. Then we create a list of all the words without repletion – that's the list of unique words, where each word appears only once. The ratio is the quotient resulting from the division of the total number of words by the number of unique words. The data is graphed and a correlation coefficient was calculated. The correlation coefficient for this ratio is 0.91

Source Name	# of Words / # Unique Words
REXML	11.864
COBOL (Eng)	6.796
HARMONIZE	5.143
TAGA	3.818
ADABAS (Heb)	2.838
MISMO	2.210
MFDX	2.200
COBOL (Heb)	2.169
REPML	2.142
RETS	2.132
NCREIF	1.737
EDI	1.000
SWIFT	1.000

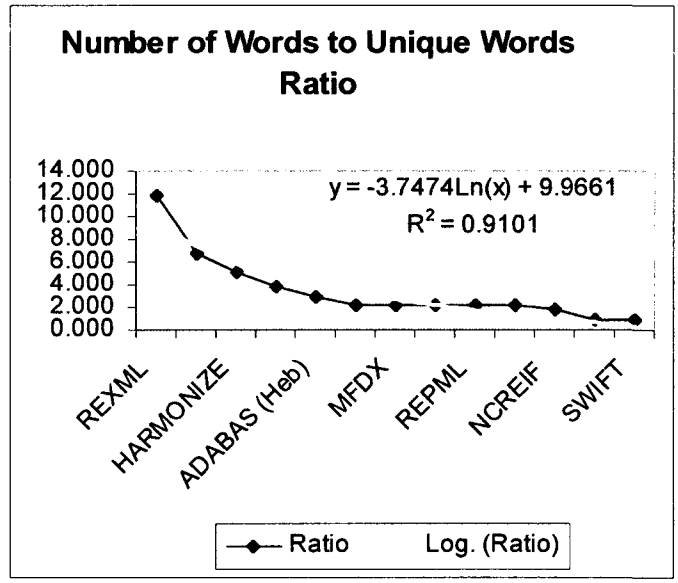


Table 85: Number of Words to Unique Words Ratio

Table 86 lists all DDL data sources sorted by the ratio of Words to Meanings in each such data source. That is, we count the total number of words used in every given DDL representative. Then we counted the number of meanings for each unique word. The ratio is the quotient resulting from the division of the total number of words by the number of meanings. The data is graphed and a correlation coefficient was calculated. The correlation coefficient for this ratio is 0.92

Abbreviated Source Name	# of Words / # Meanings
SWIFT	1.000
EDI	1.000
REXML	0.911
HARMONIZE	0.826
REPML	0.672
TAGA	0.369
RETS	0.328
COBOL (Eng)	0.185
MFDX	0.156
MISMO	0.156
COBOL (Heb)	0.149
ADABAS (Heb)	0.136
NCREIF	0.128

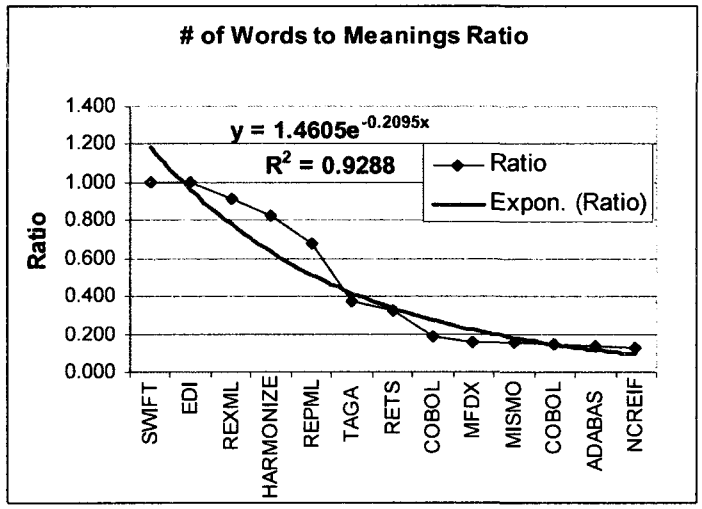


Table 86: Number of Words to Meanings Ratio

Table 87 lists all DDL data sources sorted by the ratio of meanings divided by # of unique words in each such data source. That is, we count the total number of meanings for every word in every given DDL representative. Then we counted the number of unique word in each DDL. The ratio is the quotient resulting from the division of the total number of meanings by the number of unique words. The data is graphed and a correlation coefficient was calculated. The correlation coefficient for this ratio is 0.83

Abbreviated Source Name	Meanings / unique words
MFDX	3.818
MISMO	3.802
NCREIF	3.225
ADABAS (Heb)	3.114
RETS	3.064
COBOL (Heb)	2.940
TAGA	1.786
HARMONIZE	1.227
EDI	1.000
SWIFT	1.000
COBOL (Eng)	0.925
REPML	0.859
REXML	0.533

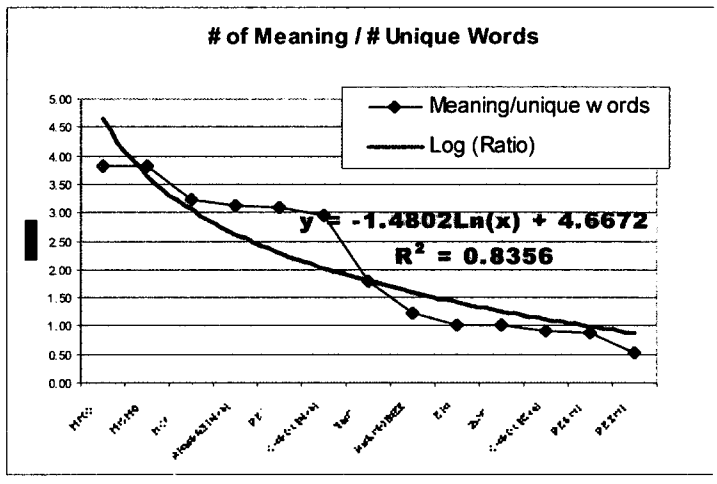


Table 87: Number of Meanings to Unique WordsRatio

We set to calculate the ratio between the number of words that have more than one meaning and the total number of their meanings. This is very similar to the data presented in Table 87 but with two noticeable differences. First - EDI and SWIFT are omitted, as each signifier there has exactly one meaning, in accordance with their design. Second – we are interested only in words that have more than one meaning. Table 88 lists all DDL data sources (except EDI and SWIFT) sorted by the ratio of number of meanings divided by number of words meeting the criteria of having more than one meaning. That is, we find all words with more than one meaning, and ignore words with one or zero meanings. Then we tally the number of meanings for every such word in our set, in every relevant DDL. The ratio is the quotient resulting from the division of the total number of meanings by the number of participating words. The data is graphed and a correlation coefficient was calculated. The correlation coefficient for this ratio is 0.95

Source Name	Ratio
EDI (Omitted)	N/A
SWIFT (Omitted)	N/A
COBOL (Eng)	12.000
MFDX	10.357
MISMO	10.357
ADABAS (Heb)	9.917
TAGA	8.538
REXML	8.142
NCREIF	8.028
COBOL (Heb)	7.951
HARMONIZE	7.884
RETS	7.722
REPML	7.411

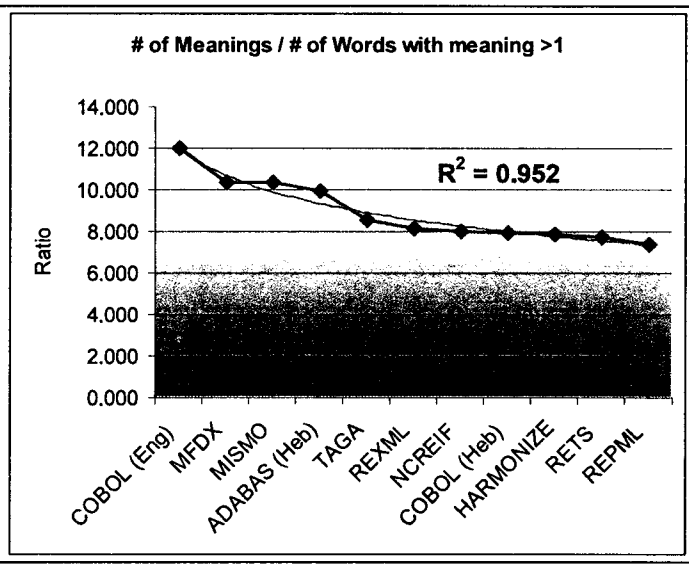


Table 88: Ratio of meanings to words that have more than one meaning

12.2 Variety Measured Through Ambiguity

Chapter 10 outlined the methodology devised for analyzing different types of ambiguities as a method to assess internal and external CAS variety. The analysis of ambiguities internal to DDL is done first, followed by analysis of cross-structures ambiguities.

This analysis is limited to seven out of the thirteen representative data structures. The balance has been accounted for in a peer reviewed article (Rohn & Klashner, 2004). A repetition of the same analysis has no added-value for drawing conclusions based on facts.

Cross-structures ambiguity analysis is limited to the Real Estate vertical market for two reasons: first, it has the largest number of DDLs representatives in one domain; they are all contemporary DDLs; second, it has a large overlap of objects among data structures. In contrast, this research collected two data structures for the banking industry, two for the travel industry two data structures based on Hebrew. Cross structure ambiguity using seven data structures makes a much stronger case than two structures only.

12.2.1 Internal Ambiguity Analysis

Each one of the standards in Table 90 exhibits word sense ambiguity, ranging from 2 to 6 meanings for its most used word. In addition, these standards exhibit other types of ambiguities, as shown hereafter.

RETS exhibits the following types of ambiguity:

- **Structural ambiguity**; for example, "Change Type" can either mean a type that is being modified or the nature of a change that occurred somewhere, but not both.

- **Projection ambiguity**; for example "CarrierRoute" can refer to a local common carrier (say a bus company), a telecommunication operator (also known as carrier in the telecom industry), or perhaps the last four digits of a location's nine number zip code. The later requires prior knowledge about the United States Postal Service zip code system.
- **Referential ambiguity**; for example, "Close Date" can be a date in close proximity (the date is close) or, as the original intention is, the date on which a sale was recorded. This event is normally referred to as the "closing date" in both the mortgage and real estate industries as seen on Fannie-Mae Uniform Residential Loan Application Form 1003.
- **Resolution ambiguity**; for example, the same term "CloseDate" also qualifies as an ellipsis because it is semantically underspecified.

REPML exhibits the following types of ambiguity:

- **Structural ambiguity**. Most of it is comprised of single words or acronyms (e.g., "pl4"). These elements of natural language do not exhibit structural ambiguity.
- **Projection ambiguity**. For example, it makes use of the acronyms "ID" four times. It also uses the word "identifier" in six different places in the proposed standard. One needs to presuppose that "ID" is a holophrase for "Identification" or perhaps "Identity". The distinction that ID and Identifier are probably not the same is inferred. Similarly, resolution of the term

"pobox" requires knowledge about Post Office Boxes. This term also serves as a good example of meaning preserved by replication.

- **Referential ambiguity**. For example, it makes use of the term "identifier" in six different nodes of XML schema. One needs to have knowledge of the entire path (e.g., context) to attempt a resolution. Additionally, it is not guaranteed that such context is always available for automatic integration processes.
- **Resolution ambiguity.** For example, "property" is a term used in the schema. It is a semantically underspecified element since there are many possible ways to interpret the term in the given context. It can be interpreted as an area (a place), or as "belonging" or "ownership", both valid in the context of real estate.

MFDX exhibits the following types of ambiguity:

- **Structural ambiguity**; for example, "Directions" can either mean instructions to get to the property, or the directions the property faces, but not both.
- **Projection ambiguity**; for example "DayID" can refer to the day of the week when the broker's office is open, or perhaps a mechanism that identifies days. The later requires prior knowledge about such a mechanism.
- **Referential ambiguity**; for example, "Close Time" can be a time when the property is closed or, as the original intention is, the time in which the broker's office is closed.

- **Resolution ambiguity**; for example, the same term "CloseTime" also qualifies as an ellipsis because it is semantically underspecified.

NCREIF exhibits the following types of ambiguity:

- **Structural ambiguity**; for example, it uses the term "Permanent or Development Crops". It is not clear whether the noun "crops" applies to the "permanent" or just to the "development". That is [(permanent crop) or (development crop)] versus [(permanent) or (development crop)].
- **Projection ambiguity**; for example, it uses the terms "Cost Approach Improvement Value" and "Cost Approach Land Value". A presupposition (e.g., knowledge about the world) can be integrated into the overall meaning of each term, as proposing a different strategy to appraise a reality's value.
- **Referential ambiguity**; for example, the term "Highest and Best Use" can be read as [(highest use) or (best use)] or as [(highest and best) use].
- **Resolution ambiguity**; for example, the term "Agribusiness" is an ellipsis and thus a semantically underspecified element.

REXML exhibits the following types of ambiguity:

- **Structural ambiguity**; for example, "Budget Account Reference" can refer to a budget, or to an account, or to an account within a given budget, or to a budget within a given account.
- **Projection ambiguity**; for example "DateOfSale" can refer to the date when the sale of a real estate took place, or the date the sale was recorded at the

county's clerk office. The later requires prior knowledge about the manner in which transfer of real estate ownership occurs in most (not necessarily all) counties in the United States. In the US, it is the later that actually transfers the ownership from the seller to the buyer.

- **Referential ambiguity**; for example, "Breakpoint Reference" can be a physical point of discontinuity, change, or cessation; or, as the original intention probably is, a reference to a point in time or an event that will make the financial investment profitable.
- **Resolution ambiguity**; for example, the same term "Breakpoint Reference" also qualifies as an ellipsis because it is semantically underspecified.

MISMO exhibits the following types of ambiguity:

- **Structural ambiguity**; for example, "Directions" can either mean instructions to get to the property, or the directions the property faces, but not both.
- **Projection ambiguity**; for example "DayID" can refer to the day of the week when the broker's office is open, or perhaps a mechanism that identifies days. The later requires prior knowledge about such a mechanism.
- **Referential ambiguity**; for example, "Close Time" can be a time when the property is closed or, as the original intention is, the time in which the broker's office is closed.
- **Resolution ambiguity**; for example, the same term "CloseTime" also qualifies as an ellipsis because it is semantically underspecified.

The balance of the structures is analyzed in a similar manner. Table 89 contains a summary of ambiguities found.

Source	Ambiguity Type				
	Word Sense	Structural	Projection	Referential	Resolution
RETS	Yes	Yes	Yes	Yes	Yes
REPML	Yes	No	Yes	Yes	Yes
NCREIF	Yes	Yes	Yes	Yes	Yes
MISMO	Yes	Yes	Yes	Yes	Yes
MFDX	Yes	Yes	Yes	Yes	Yes
REXML	Yes	Yes	Yes	Yes	Yes
HARMONIZE	Yes	Yes	Yes	Yes	Yes
EDI	No	No	No	No	No
SWIFT	No	No	No	No	No

Table 89: Internal Ambiguities Summary

12.2.2 Cross Standard Ambiguity

A material construct in the Real Estate vertical market is *location*. From a legal perspective it is expressed in block and lot. The public commonly uses street address to locate a place. In many countries a street address is complemented by a postal code. A less common means of location description is longitude and latitude coordinates. The following table summarizes what each data structure in the sample use for location. Variety among data structures is evident.

Source Name	Location Data	Comments
RETS	(a) Parcel number assigned by the taxation authority with jurisdiction over the property. (b) Street address, either parsed or unparsed.	
REPML	(a) Longitude & Latitude (b) Street address	
MFDX	(a) Street address broken down to fields. (b) Longitude & Latitude (c) Metropolitan Statistical Area Code for Subject Address (d) Property or Management Company's Internal Address Reference	
NCREIF	(a) Street address broken down to fields.	Asset level operating information
REXML	(a) Street address broken down to fields.	
EDI	(a) Street address broken down to fields.	Application for mortgage Source: AUS logical data dictionary, version 2.3.1 As of 22-Jan-2004
HARMONIZE	(a) Street address broken down to fields.	

Table 90: Location Expressed in Different Standards

This section analyzes ambiguity between standards. For example, identical XML tags in Real Estate do not carry the same meaning. In contrast, there are identical concepts expressed differently. Table 91 has representative samples of cross-reference ambiguities, followed by an explanation for each line in the table.

#	Source 1			Source 2		
	Source	Term Used	Meaning	Source	Term Used	Meaning
1	RETS	ListDate	The date on which the listing contract became effective.	REPML	OriginalListingDate	The date on which the listing contract became effective.
2	NCREIF	Cancellation	The lessee may terminate the lease prior to its expiration date on stated dates or upon the occurrence of certain events.	REPML RETS	N/A	
3	REPML	MapLocation	Location on a map	HARMONIZE	Location	Building or region
4	REXML	Total Floor Area	Total area	MISMO	Gross Building Area	Total area

Table 91: Cross Standards Analysis

Item 1: there are two distinct terms that express an identical concept, thus exhibiting redundancy, which is a characteristic of natural languages. This demonstrates a challenge integration systems need to overcome: the ability to determine that two distinct XML expressions are in fact an identical concept. In addition, the RETS schema and the REPML schema differ in their scope and granularity. Hence, full integration of the two is impossible without violating meaning-preservation constraints. The constraints require a mapping function that is invertible, proof preserving, structure preserving and vocabulary preserving. Vocabulary preservation refers to same content words or symbols that represent categories, relations, and individuals in an ontology; these must appear in both mapping from source one to source two and vice versa (Sowa 2001).

Item 2: NCREIF has a reference to tenant in property that is rented out. No such concept exists in neither REPML nor RETS. Consequently it is impossible to accurately map NCREIF into REPML or into RETS without loss of information. This gap is not a natural language characteristic; rather it is a matter of information scope and

completeness. Having such gaps preclude satisfying the meaning-preservation constraints.

Item 3: REPML references a location on a map, while HARMONIZE makes references to either a building or a region. This exhibits projection ambiguity, as presupposition (knowledge about the world) is integrated into the overall meaning this XML tag. This vocabulary gap precludes satisfying meaning-preservation constraints.

Item 4: there are two distinct terms that express an almost identical concept, thus exhibiting redundancy, which is a characteristic of natural languages. REXML refers to residential location while MISMO refers to commercial real estate. Each term measures a slightly different area. This granularity gap rules out satisfying all meaning-preservation constraints.

Ambiguity is a source of *Variety*. Resolution of ambiguity requires a *regulator* that satisfies the LRV. Unconstrained variety is undesirable in CAS; it is detrimental to automatic data integration from autonomous heterogeneous data sources. Ambiguity introduces noise to a communications channel, which begets redundancy in an attempt to overcome the noise. The answer to the first research question suggests that redundancy is a hinders automatic data integration. Lack of measurable decrease in redundancy is relevant to answering the second research question of whether there been real advancement in DDL design towards such integration. New DDLs do not have a variety constraining mechanism, which is desirable for achieving automatic data integration.

12.3 Tension Measured Using Meaning Preservation

The research evaluates if a given DDL is designed to create tension as understood in CAS theory. It does so by evaluating if implementations of DDLs satisfy meanings

preservation requirements as defined by Sowa's work, expounded on earlier in this work, along with ambiguity explained qualitatively. Ambiguity is unclearness by virtue of having more than one meaning, and can be defined formally. For the purpose of this research ambiguity (A) is defined as a function f that maps Σ-symbols to symbol meanings over Σ. $s \in \Sigma$ is the origin symbol, and $m \in$ symbol meanings over Σ. Cardinality of the two groups is not required to be equal between $|\Sigma|$ and | symbol meanings over Σ |. However, it is required that $|\Sigma| \leq$ |symbol meanings over Σ|, and it is required that at least one $f(s) \rightarrow m$ exists otherwise we end up with symbols that are meaningless. The existence of $f(s)$ and its exact inverse $g(f(s))$ ensures inevitability and vocabulary preservation. Suffice to show that there are instances not obeying this requirement to conclude that Sowa's requirements for meaning preservation are not met. $f(s) \rightarrow m$ in a data structures requires treating any data element as "s" regardless of the number of words it is made of. It could also be a data element whose name is not engineered using natural language, as was the case with old Basic programs. A data element s needs to map to its counterpart m and back to satisfy two conditions of meaning preservation, namely inevitability and vocabulary preservation. Suffice to show that there are instances not obeying this requirement to conclude that Sowa's requirements for meaning preservation are not met.

Meaning preservation requires the creation of structure-preserving maps, or homomorphism. There may be more than one mathematically correct solution when mapping from the source to the target data structure (Kolaitis 2005). This is illustrated in Figure 76: it is possible too map from *Dog1.Head* to *Car.Hood* and so forth ending up with perfect homomorphism. It would be a wrong solution because semantics is lost.

Mapping from "Dog1" to "Dog2" provide an example of partial mapping, and there are 2 possible mappings. Mapping back from "Dog2" to "Dog1" has two different solutions, each yielding a local homomorphism, which in turn results in partial integration only. The single correct solution is the one that satisfies the requirement of mapping back to the source correctly (Sowa 2001), and there is no guarantee such mapping exists.

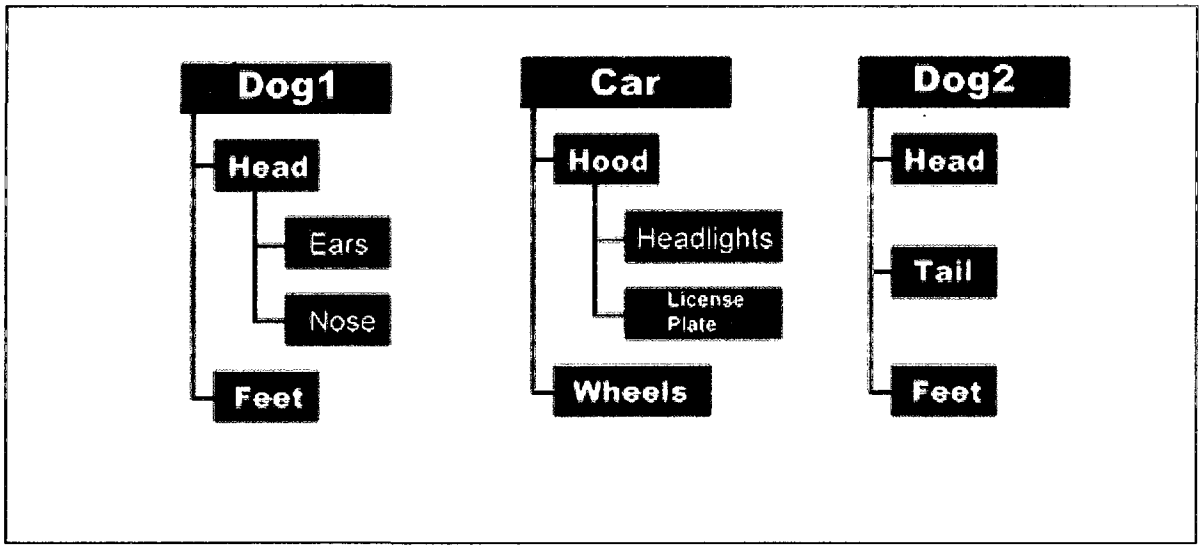


Figure 76: Three Data Structures

Sampled instances of data structures are used to qualitatively assess if homomorphism exists as a DDL design constraint. The number of nodes in each data structure are counted and compare to the number of nodes in a data structure that is a candidate for integration in the same domain. Different number of nodes precludes structure preservation, a Sowa requirement for meaning preservation.

Row Number	Abbreviated Source Name	DDL	# of Data Elements
1	REXML	XSD	1862
2	HARMONIZE	RDF	912
3	RETS	DTD	454
4	REPML	XML	262
5	EDI	Proprietary	165
6	MFDX	XML	138
7	MISMO	DTD	138
8	NCREIF	NL	112
9	TAGA	OWL	83
10	ADABAS (Heb)	NL	82
11	COBOL (Heb)	COBOL	73
12	COBOL (Eng)	COBOL	57
13	SWIFT	Proprietary	37

Table 92: Number of Data Elements

Table 92 lists the number of data elements in each DDL sample. MFDX and MISMO (rows 6 and 7 respectively) are both from the Real Estate domain and have the exact number of data elements. They seem to be present the only case where homomorphism is possible. In fact, not only that it is feasible, it can't be otherwise because MFDX is MISMO, only implemented in XML rather than in DTD.

The TAGA and ADABAS samples (rows 9 and 10 respectively) have 83 and 82 data elements respectively. This may give some hope for a homomorphism if one data element is dropped from TAGA or one is artificially added to ADABAS. However, the two have only superficial resemblance. Except for an almost identical number of data elements they have nothing else in common: they are of unrelated domains; their internal structures differ in the number their groups and complex elements. The number of their atomic elements is entirely different too. These findings preclude a valid isomorphic solution in this case.

All other DDLs samples in this research preclude isomorphism between any pair of data structures due to the differences in their number of data elements, clearly listed in Table 92.

12.4 Order Measured by Entropy

Per CAS theory, morphogenic systems reduce their local entropy and increase order. This research utilizes entropy (Shannon 1948) as a direct measure of the level of order achieved by a given DDL. The level of entropy in a given data structure is an indication of its fitness for automatic integration. Table 93 reports in ascending order the level of relative entropy calculated for every data structure. All DDLs exhibit similar entropy levels ranging from 0.82 to 0.96, with the exception of the two strict standards EDI and SWIFT. There is no correlation between entropy levels and computing generation of the DDL. That is, if one expects to see an improvement in the DDL's ability to export a system's entropy and increase order as the computing industry matures, this expectation is not met.

Data Structure	Entropy
REXML	0.82
TAGA	0.85
Cobol FD	0.86
PIDX DD	0.88
ADABAS	0.89
RETS	0.91
HARMONIZE	0.92
MDFX	0.92
MISMO	0.92
PIDX Invoice	0.92
NCREIF	0.95
REPML	0.96
EDI	1
SWIFT	1

Table 93: Entropy in DDL

12.5 Data Analysis Summary

Table 94 summarizes all qualitative and quantitative attributes of the DDL examined in this study. Each attribute measured is tied to the research questions stated at the outset, via measurable key attributes: Variety, Tension, and Entropy.

Parameter	Ontology		Markup Language					Prog. Lang.		Data Dictionary		Protocol	
DDL	RDF	OWL	DTD	XML	XML	XSD	XML	Cobol FD		Natural Language		Proprietary	
Model Name	TAGA	HARMONIZE	RETS	REPML	MFDX	MISMO	REXML	Cobol (EN)	Cobol (Heb)	ADABAS	NCREIF	EDI	SWIFT
# of Data Elements	83	912	454	262	138	138	1862	57	73	82	112	166	37
# of Unique Words	33	175	212	212	105	105	323	49	77	68	156	166	37
Total Words	126	900	452	454	231	231	3832	333	167	193	271	166	37
Entropy													
H-Maximum	5.0444	7.4512	7.7279	6.8580	6.7142	6.7142	8.3354	5.6147	6.2668	6.1085	7.2946	7.3750	5.2095
H-Actual	4.2793	2.0827	7.0612	6.5611	6.1524	6.1524	6.8201	4.8425	5.8731	5.4380	6.9248	7.3750	5.2095
H-Relative	0.8483	0.2795	0.9137	0.9567	0.9163	0.9163	0.8182	0.8625	0.9372	0.8902	0.9493	1.0000	1.0000
Redundancy	0.1517	0.7205	0.0863	0.0433	0.0837	0.0837	0.1818	0.1375	0.0628	0.1098	0.0507	0.0000	0.0000
Internal Ambiguity													
Word sense	Y	Y	Y	Y	Y	Y	Y	Y	Y	Not Applicable		NO	NO
Structural	Y	Y	Y	NO	Y	Y	Y	Y	Y	Not Applicable		NO	NO
Projection	Y	Y	Y	Y	Y	Y	Y	Y	Y	Not Applicable		NO	NO
Referential	Y	Y	Y	Y	Y	Y	Y	Y	Y	Not Applicable		NO	NO
Resolution	Y	Y	Y	Y	Y	Y	Y	Y	Y	Not Applicable		NO	NO
Meaning Preservation													
Invertibility	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO
Vocabulary Preservation	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO
Structure Preservation	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO
Cross Standard Ambiguity	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Not Applicable	
Zipf Regression Coefficient												Flat line	Flat line
Word Frequency Distribution	0.5425	0.9058	0.9169	0.8642	0.9337	0.9937	0.9353	0.9187	0.9112	0.9216	0.9044	#na	#na
Distribution of Meanings	0.9846	0.9741	0.9751	0.9713	0.9705	0.9705	0.9892	0.9778	0.8955	0.9499	0.9813	#na	#na
Distribution of Word Usage	0.0923	0.3049	0.8438	0.7025	0.9175	0.9175	0.6907	0.6418	0.9085	0.6568	0.8555	#na	#na

Table 94: Summary of Measurements

12.6 Chapter 12 Summary and Implications to the Research

This chapter summarized the findings and their analysis using tables and graphs of some distributions of elements, words, and meanings in the ergodic source of data chosen for this research. A cursory review shows there is no significant difference between DDLs except when compared to the EDI and SWIFT standards. All DDLs allow for a high degree of ambiguity. None supports meaning preservation as a design attribute. DDLs cannot be discriminated based on actual entropy, except for EDI and SWIFT. These observations are discussed in detail in chapter 13 and will be used in the implications discussion of the analysis.

Implication of chapter to study	How Implication will be used
Analysis of Variety	The breadth and depth of variety, quantified and qualified per the methodology, indicates DDLs are not designed to overcome this barrier to automatic integration. A proposed alternate approach to DDL design will needs to address this aspect of DDL.
Analysis of Tension	Lack of support for the creation of tension across all DDLs examined requires a radically different DDL design approach.
Analysis of Entropy	New design of DDL will require a mechanism to export the internal entropy of the DDL for its underlying system to sustain morphogenesis qualities.

Table 95: Summary of Chapter 12 Implications to the Study

CHAPTER 13

FINDINGS AND DISCUSSION

When facing any complex problem, one should try and understand it as a totality. How has it arisen and for what reasons? Where is it going and what route is it taking? Is it changing its nature or structure as it develops? Will it eventually solve itself or become extinct? Reverse engineering provides a prime example of an approach that investigates complex objects. The investigating engineer must know why the machine or device was built and understand the problem it is designed to address. Only then comprehension of the machine is possible. Its parts can only be understood in terms of the whole. This chapter is the next stepping stone to gaining an understanding and insight into DDL by reflecting on the preceding parts and tying them to the whole.

This chapter builds on the data gathering and analysis method presented earlier. Each section in this chapter is linked to one or more research question posed at the outset. The segments create arguments and substantive rational used as building blocks towards providing the definite answers articulated in the last chapter.

This chapter proposes a DDL evaluation framework that developers of DDL can use to test if their approach is significantly different (and hopefully better) from DDL proposed and implemented in the last four decades. This work asserts that "better" will have improved support of CAS characteristics that are required for automatic data integration of heterogeneous sources. The chapter then discusses the findings of chapter 12 (analysis) from a longitudinal view point and then moves to discuss some findings as they relate CAS attributes (variety, tension and entropy) to automatic data integration. Having identified common weaknesses in all DDLs, the chapter sets the stage for a DDL

design approach that is consistent with CAS requirements to support automatic data integration.

13.1 The Evaluation Framework

Chapters ten described a DDL evaluation methodology that focuses on three CAS attributes: Variety, Tension, and Entropy. Chapter eleven illustrated how the methodology is put to use. Combined, this creates a framework of measurements that is implemented consistently across data definition languages. This consistency facilitates the accurate quantification of fundamental attributes in DDL. The procedural aspects of the evaluation framework are as follows:

1. Identification of a DDL of interest
2. Finding at least one data structure built using a given DDL
3. Extraction of data elements
4. Breakdown of data elements to their atomic constituents
5. Count the number of signifiers (words)
6. Count the number of signifier meanings (words)
7. Evaluate internal ambiguities and cross-structures ambiguities
8. Measure the signifiers' Zipf distributions
9. Calculate their entropy
10. Evaluate how the DDL satisfy meaning preservation requirements

Once the attributes have been quantified and qualified, one needs to compare each of the measurements to all known patterns previously measured (e.g., Zipf Distributions for Variety, meaning preservation for Tension, and Entropy). The comparison will yield one of two possible outcomes:

a) No significant difference - trivially, a DDL under investigation does nor contribute anything new, thus it will not advance automatic integration of heterogeneous data sources beyond the current state of affairs.

b) New patterns show significant difference - this interpretation is the essence of several facets of the research and somewhat complex. Therefore, it is discussed at length and in detail within the CAS discourse in section 13.3 on page 238. For now, consider "significant" to be a noticeable change in any of the measured attributes – Variety, Tension and Entropy.

13.2 Consistent Patterns over Time

The first research question has been answered by proposing a data integration theory asserting that satisfying LRV and the existence of a noiseless communications channel are necessary requirements to fully support automatic data integration. The analysis is looking for evidence to help reject the hypothesis posed as a data integration theory. Lack of such evidence suggests that the proposed theory is logically sound and well grounded. The second research question asks if there has been real advancement in DDL design towards automatic data integration of autonomous heterogeneous sources. The rest of this chapter provides buiding blocks necessary to formulate a definite and complete answer to each research question.

This longitudinal study exposes consistent patterns in DDLs and data structures built using these DDLs. Studying Table 94: Summary of Measurements (page 222) reveals that there is no significant difference between DDLs on any of the fundamental measures used in the framework. The following sections discuss Variety, Tension and Entropy longitudinal patters. Each one of the three measures is related to the first research question – what the theoretical necessary requirements of a DDL built to fully support automatic data integration from autonomous heterogeneous data sources. The findings, that all DDLs look alike, answers the second research question: has there been real advancement in DDL design towards such integration? The answer is a resounding "no".

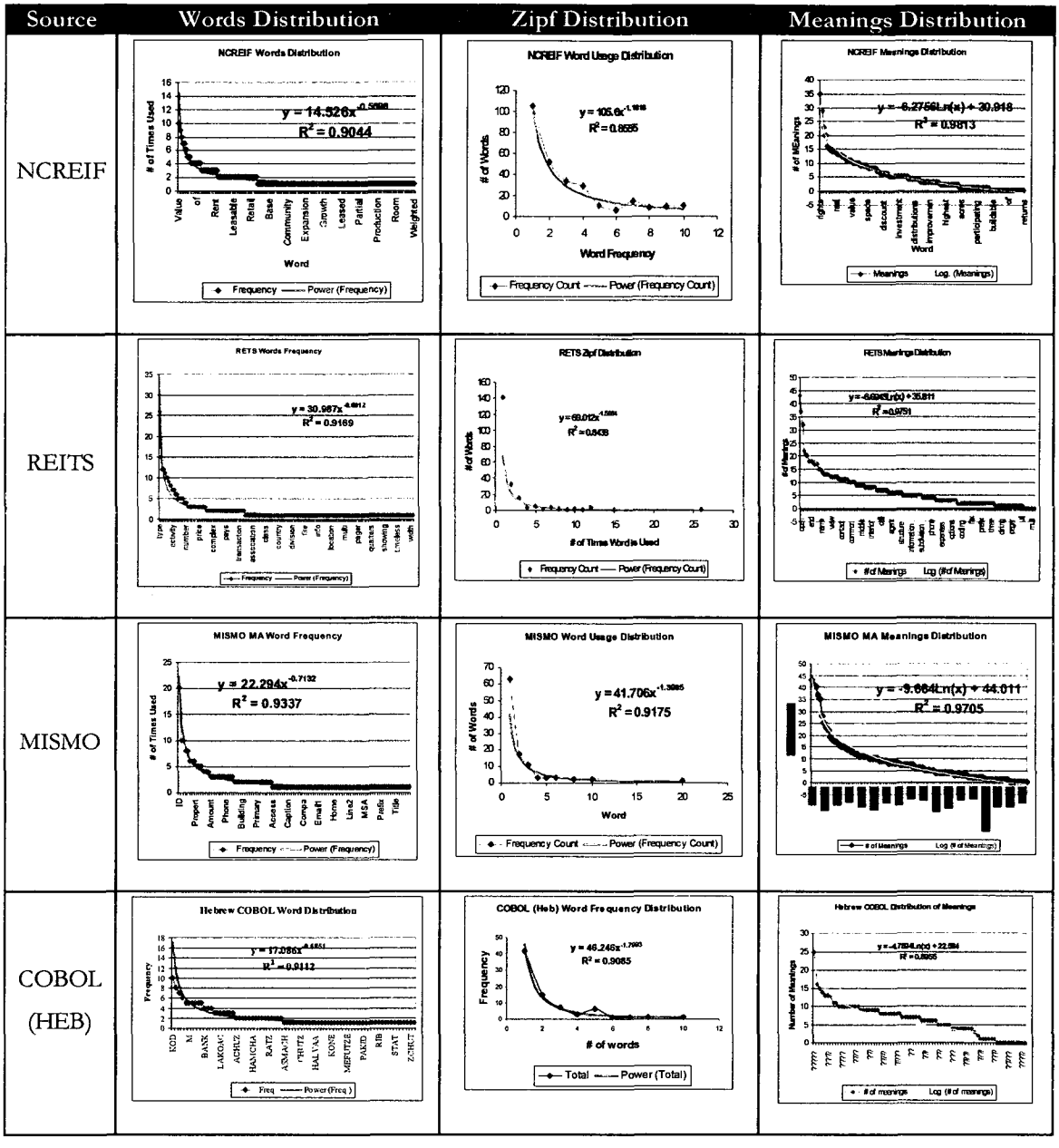


Figure 77: Similar Zipf Correletion Coeficients

13.2.1 Variety

Chapter six defined variety in relation to a set of distinguishable elements as meaning either the number of distinct elements, or the logarithm to the base 2 of the number of distinct elements, which Shannon defines as "information". The summary table in chapter

12 shows via simple tally that variety in the number of elements in the DDLs unmistakably exists. The data is extracted here in Table 96 for the reader's convenience. One needs to remember that the sampling limited SWIFT to 37 messages belonging to category 1 (financial institutions transfers) and category 5 (securities) only. SWIFT messages have an internal structure that consists of five blocks. Since they are distinguished by position only, they are not viewed as distinct data elements, as explained in chapter 2. Thus, the number of elements in SWIFT was multiplied by five for the purpose of calculating a realistic average of bits per data element for every computing era. The result is graphed in Figure 78 followed by a short explanation and discussion.

Data Structure	# of Data Elements
TAGA	83
HARMONIZE	912
RETS	454
REPML	262
MFDX	138
MISMO	138
REXML	1862
Cobol (EN)	57
Cobol (Heb)	73
ADABAS	82
NCREIF	112
EDI	166
SWIFT	37

Table 96: Data Elements

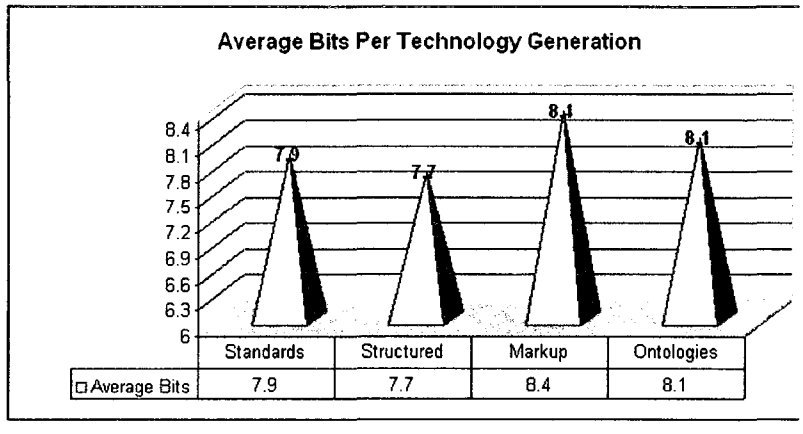


Figure 78: Average Variety (in Bits) per Computing Era

The difference among the four computing eras in the number of bits required to convey one piece of information is negligible, showing consistent pattern over time. Experienced COBOL programmers may even assert that they have worked with COBOL data structures having over two hundred elements rather than the 73 and 57 data elements in the sample. This would easily put the structured approach en par with the other approaches for information bits per data elements. Markup languages appear to require

the most bits per data element, but this is only due to the extraordinary number of elements in REXML. On average, the sample shows that data structures require 7.9 bits to convey one piece of information, with a standard deviation of 0.22 bits only.

Zipf Distribution of Words is yet another indication of variety. The words in each data structure are ranked according to their usage frequency and a regression coefficient is calculated for every data structure, as illustrated in Figure 79. The same procedure was applied to each DDL sample. The coefficients of all DDLs are plotted in Figure 80.

WORD	FREQUENCY
VKSD0080	28
STATUS	28
MESSAGE	28
TEXT	24
IO	24
Z	21
APPL	19
RESULT	13
HEADER	1
DATASIM	1
CODE	1
2	1
0401	1

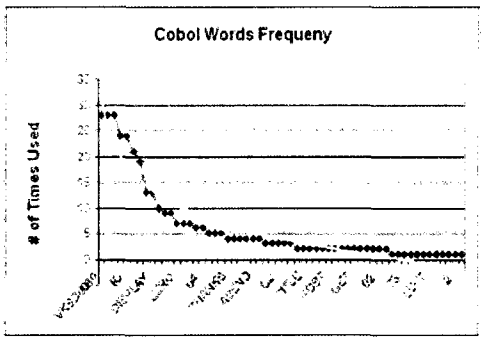


Figure 79: Word Frequency Illustration

All data definition languages, from the 1960's to date, use mostly words in natural language to engineer data structures. Although data structures are mostly constructed of nouns, the words used exhibit unmistakable Zipf distribution characteristics, as illustrated in Figure 79 for COBOL. The Word Frequency Distribution of all DDLs resemble a perfect power distribution having a correlation coefficient R^2 about the 0.90 value, per Figure 80 below. It is not possible to discriminate between DDLs using this measurement.

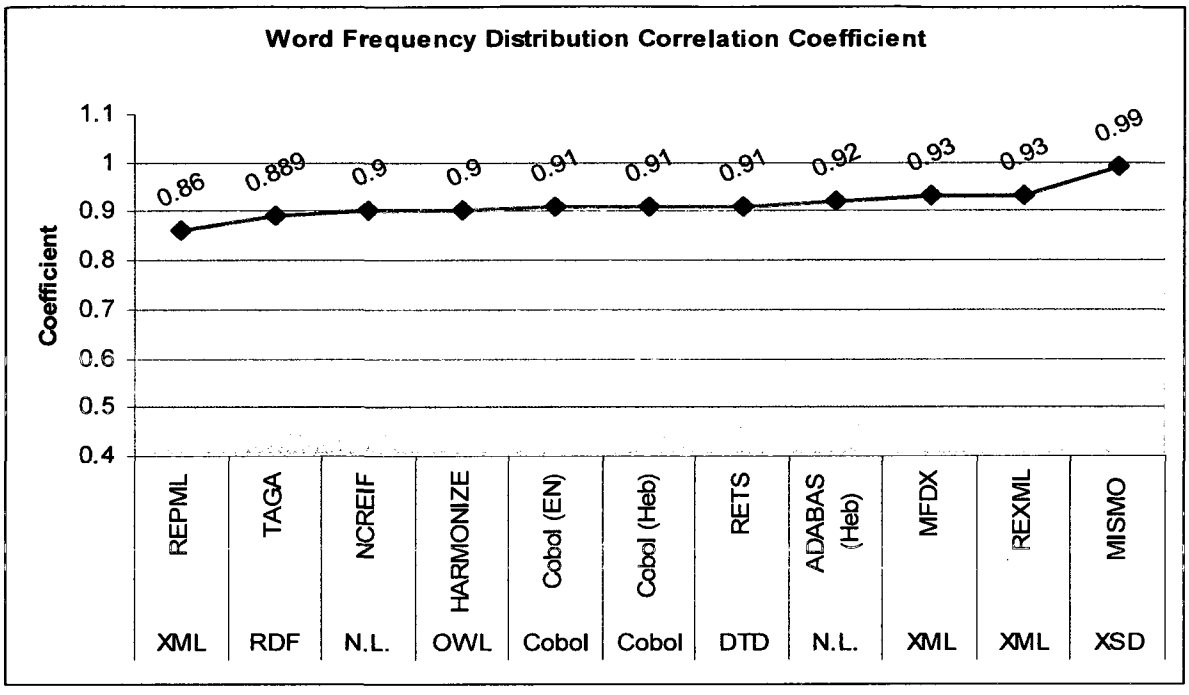


Figure 80: Word Frequency Distribution Correlation Coefficient

Zipf Distribution of Meanings is also a measure of variety. For each DDL, words were ranked according to their number of meanings. Frequencies of meanings were plotted for each DDL, and a correlation coefficient was calculated. The procedure is illustrated in Figure 81.

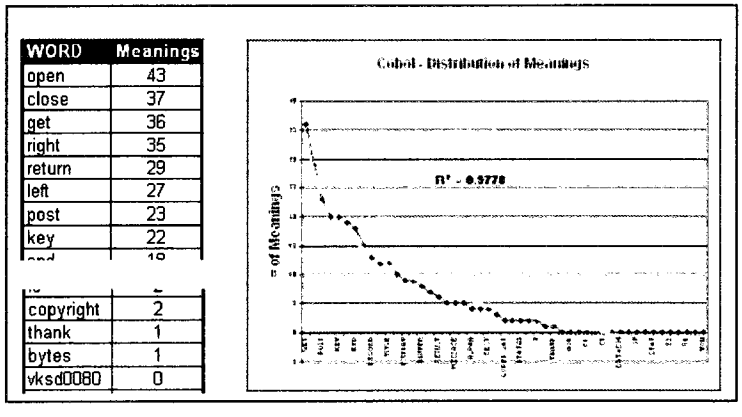


Figure 81: Word Meanings Illustration

The regression coefficients for all DDLs are plotted in Figure 82. Plots for each DDL were given in chapter 11.The meanings exhibit unmistakable Zipf distribution characteristics. The Meanings Frequency Distributions of all DDLs resemble a perfect power distribution

having a correlation coefficient R^2 at or above the 0.90 value, per Figure 82 below. It is not possible to discriminate between DDLs using this measurement.

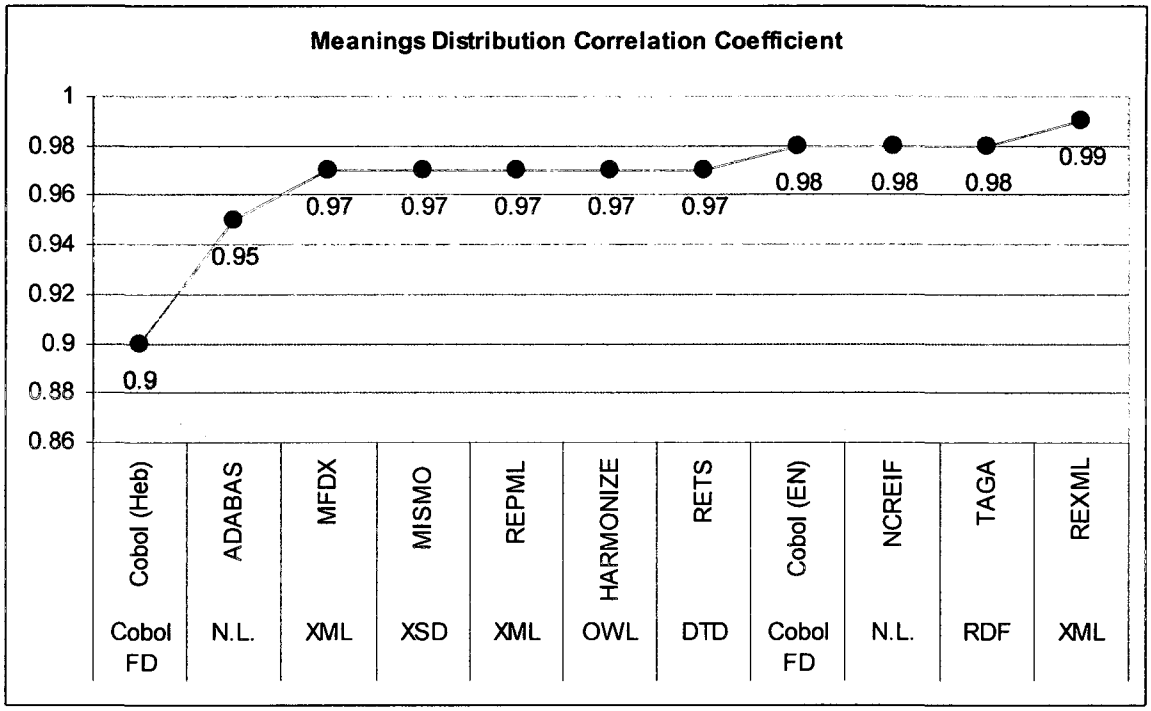


Figure 82: Meanings Distribution Correlation Coefficients

Zipf Distribution of Word Frequencies: For each DDL, the occurrences of words was counted and then plotted. Figure 83 illustrates the three steps process, from left to right. First, words are counted for the number of times they appear in the DDLs. For example, the word *Reservation* appears 18 times. The next step is to count how many words appear 18 times, 16 times, and so forth. There is one word only that appears 18 times, as shown on the last line of the middle part in Figure 83, depicted by two blue lines from left to center. Once the step is completed, the table is plotted and a correlation coefficient is calculated. This third step is illustrated on the right part of Figure 83. This data was furnished for each DDL in chapter 11.

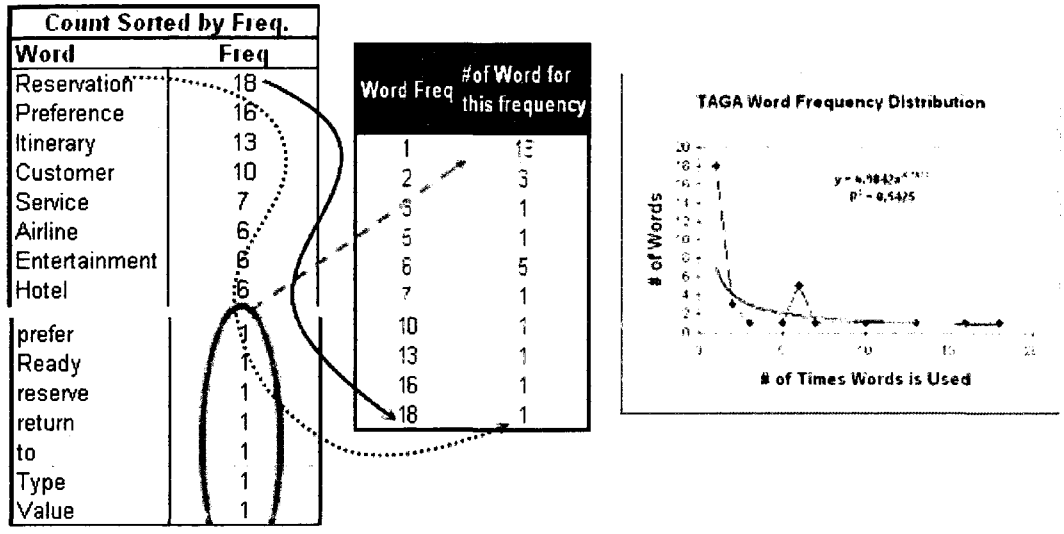


Figure 83: Getting the Distribution of Word Frequencies

The correlation coefficients of all DDLs (except for EDI and SWIFT of course) are plotted, as shown in Figure 84. The result has no visible pattern and looks quite stochastic. The two ontologies, Harmonise and TAGA, have a similar low coefficient, which indicates there is less order there than in the other DDLs.

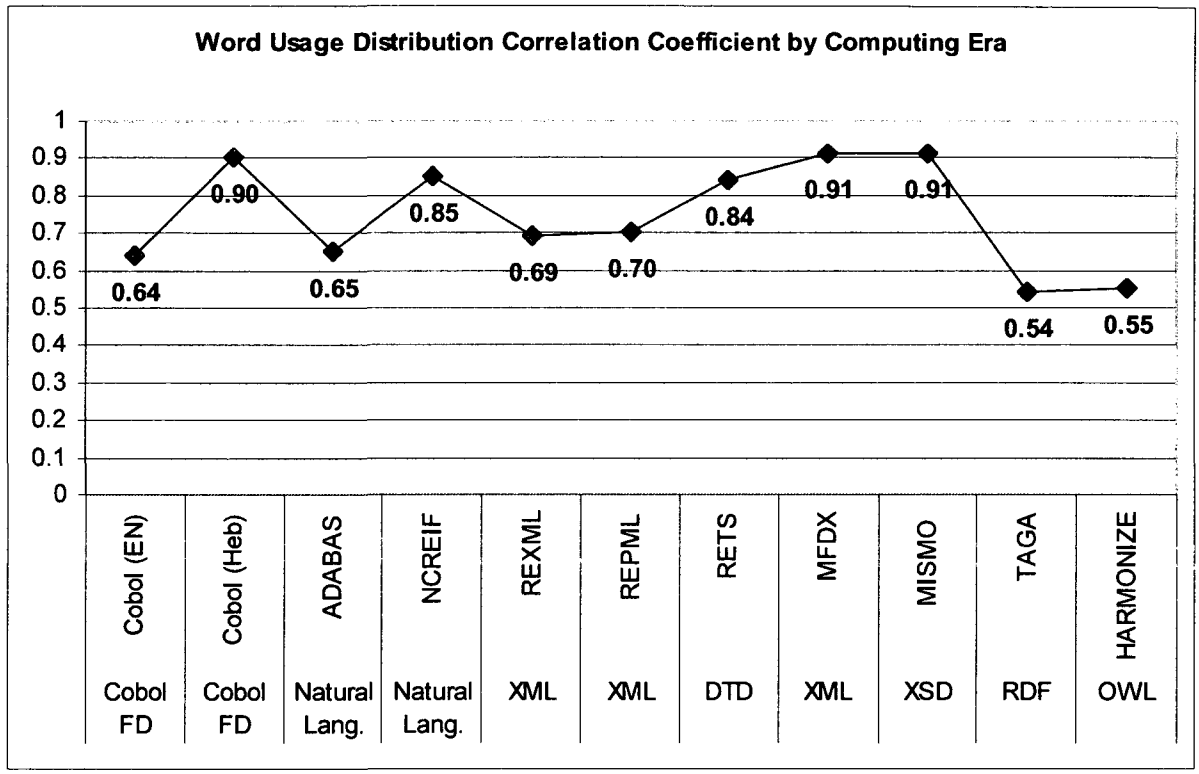


Figure 84: Word Usage Correlation Coefficients Sorted by Computing Era

The Harmonize raw data shows that the low correlation is mostly attributed to outliers. There are 72 words that appear 4 times each in the Harmonise ontology, seen as a spike on the graph in Figure 85. This quantity is an unusual high number for the mode of word usage. This anomaly could be partially caused by the fact that the Harmonise ontology is a product of several nonnative English speaking colleagues, who nevertheless collaborated to produce an English-based ontology. As such, it might not be part of ergodic source of the English language, as its appearance is illusive. The TAGA (Travel Agent Game in Agentcities) ontology is the only data structure in the sample created as an academic exercise to demonstrate "that agent and semantic web can fit

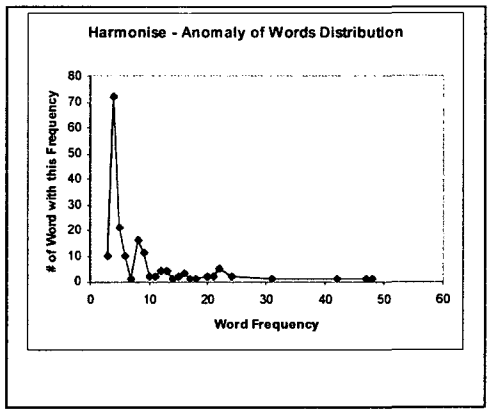


Figure 85: Harmonise Distribution

together" (Zou and Finn 2003). The goal for its development and the environment it was developed in may explain its low correlation coefficient.

Internal Ambiguity: All data definition languages, from the 1960's to date exhibit ambiguities that are a direct result from using natural language to engineer data structures. The only exceptions are predefined agreed upon standards, namely EDI and SWIFT. This should not come as a surprise, because disambiguation was a design feature of each standard.

Cross-Structures Ambiguity: All data definition languages, from the 1960's to date, exhibit cross-structure ambiguity that is a direct result from using natural language to engineer data structures. The only exceptions are predefined agreed upon standards, namely EDI and SWIFT. This too is self explanatory, as EDI and SWIFT were not designed to interact with anything except EDI and SWIFT respectively.

13.2.2 Tension

Building on the explanation of tension that was provided in chapter 6 section 3.2, it is understood that the formation of meaningful mappings between some parts of a system and its environment is sustained by non physical tension. The mapping needs to "make sense", in other words - preserve the meaning of the external variety vis-à-vis the internal structure of the system whose goal is the integration.

All data definition languages, from the 1960's to date, do not preserve meaning. This includes predefined agreed upon standards, such as EDI and SWIFT. However, predefined standards are "closed systems", hence, meaning is preserved by isolation from

the environment. Mappings between a standard and an information system's data structure is labor intensive and is achieved solely by domain experts with business and technical knowledge. They invest mental (psychic – per Buckley) energy in the mapping. That energy sustains the mapping until such time that either the system's internal data structure that is mapped to the standard changes, or the standard changes as it adapts to its environment. Case in point –changes made in mid 2007 to SWIFT (described in chapter 2) are an adaptation to changes in the legal environment mandated by the European Union.

In contrast, contemporary data structures devised using markup language (e.g., XML, XSD) or using ontologies (e.g., DAML, OIL, RDF, OWL) are supposed to be able to discover meaningful mappings and to sustain mappings whose one end may have changed but is still meaningful to the information system initiating the mapping. The new approach depends on annotation of data structures by means of ontologies. For example, a data element "thribfo", has a pointer to some ontology entry named "DateOfBirth". This mapping is of course part of the data structure design process, and is left to the data modeler. If lady luck is on his side, the data modeler will find a corresponding entry in the ontology he is supposedly committed to using. If the ontology's owner changed the entry to "DOB", the mapping is lost, as neither XML nor ontologies have a built-in mechanism that will sustain the mapping. This undesirable state can be currently rectified only if the data modeler invests additional mental energy to correct the mapping.

13.2.3 Entropy

Applied to IS, the entropy of a system expresses quantitatively an observer's uncertainty about the state a system may be in, and respectively measures the information that one

may have about the system. With the exception of EDI and SWIFT, all DDLs have high relative entropy, around 0.9. The Harmonise ontology is an anomaly, its relative entropy is only 0.27 for the same reasons hypothesized earlier, such as that it was created in English by non-native English speakers from European countries leading to an anomalous distribution of words.

The data collected from structured and semi-structured data sources show high levels of relative entropy, similar to levels found in natural languages, and specifically in English. The exceptions are protocols not based on natural language, namely EDI and SWIFT. There is no statistically significant different in the levels of entropy between any of the schemas that were engineered using natural language, regardless of their syntax – OWL, RDF, XML, DTD, XSD, Cobol or Data Dictionary expressed in either English or Hebrew.

Using Zipf's economy of words theory, the recipient's maximum economy is achieved when w different words have one meaning each, yielding exactly m meanings. That is, $w=m$. SWIFT and EDI, having zero redundancy, are the most economical means to relay a message to a listening (receiving) business partner, because they maximize w – every concept has exactly one "word", which is maximum entropy. Having exactly one word per concept satisfies all of Sowa's meaning preservation requirements, because the set of words maps to itself forming a 1 to 1 mapping with no exception.

13.2.4 Canonical and non-Canonical Data Sources

In a regular Canonical System every derivation rule (a method for generating objects) is of the form "Wv yields W'v" where W and W' are words over the alphabet of the calculus and V is a variable. Post's Canonical System is a way of defining arbitrary

enumerable sets of words. It gave rise to the computation of and operation with arbitrary enumerable sets of words not attached to their logical structure or their semantics. Data structures are arbitrary enumerable sets, therefore the notion of a canonical system is applicable description for DDLs. After all, DDLs mediate between a human initiated constructive process that uses W and transforms them to W' as data structures. Chapter six introduced Casti's clarification that a canonical control system is one that is completely reachable and completely observable. That is, every state Y_i can be traced back to its origin, some X_j where i,j $\in$[0, T] and each state Y is independent of all other states Y. Canonical systems are equivalent to meaning preserving systems: they have an initial state X_j that, with some input u(t) yields a state Y_i which is always traceable back to the origin, X_j.

Standards such as EDI and SWIFT are completely reachable and completely observable. Every data item in an IS that originates in an EDI (or SWIFT) message can be traced back to the specific EDI (or SWIFT) message. All such messages (and therefore their derived data) are independent of each other. EDI and SWIFT are enumerable sets, at least in a narrow sense. Hence, standards such as EDI and SWIFT are canonical systems that are completely reachable and completely observable. Therefore, they provide for maximum variety transfer. In contrast, all other DDLs, including ontologies (serving as pseudo-standards) are not canonical. Chapter three provide examples of circular and dead-end ontological annotations. Chapter seven provide examples of data structures that are not isomorphic, therefore precluding a mapping back to the origin and making DDLs non-canonical.

Strategies for resolving semantic ambiguity described in chapter 5 can be viewed in terms of canonical systems. There is a source alphabet with W words over it, one or more derivation rules, and a target alphabet with W' words over it. A state W' = 0 is said to be reachable from the origin if, given W(0) = 0, there exist a finite time interval [0, T] and an input (derivation rule) {u(t), t □[0, T]} such that W(t) = W'. Since the strategies do not resolve all states, it can be said that none of the strategies reviewed is completely reachable.

In summary, the values of the three CAS principle attributes are indistinguishable across computing generations of DDLs. Variety, Tension (lack thereof) and Entropy remain invariant over the years. Only standards provide a control mechanism ("regulator") whose output variety equals the variety in its input.

13.3 Additional Interpretation Using CAS Theory

Older DDLs, such as those used in the Adabas database, and programming languages such as COBOL were not designed to interact with external environments, therefore they are closed in nature. These types of DDLs are legitimately confined to the system of which they are part. In contrast, XML, XSD, DTD, RDF, and OWL are all contemporary DDLs designed to interact with their external environments. These relations are summarized in Table 97 below. They associate DDLs with the computing era to which they belong. Per their creators, the newer DDLs are designed to engage in interchanges with the environment across system boundaries. However, they are not open systems as defined by CAS, because the interchange is not an underlying driving force of these components' viability and continuity. In other words, this exposure to the environment does not facilitate their ability to create new relations, let alone autonomously modify the

system they are part of. Their inability to create new relations without human intervention means they are not designed to export their entropy, and they are not designed to respond to relevant changes in the environment. Their design does not include a mechanism to create and maintain tension. This is still deferred to human intervention; humans provide the high-level energy required to create relations and maintain tension. The energy isn't mechanical; it is mental energy as is the case in high level CAS. This concept was explored in depth in chapter 6. Therefore, contemporary DDLs fall short of being adaptive complex systems. This nuance is the reason for using the term "exposed" rather than "open" in a column heading on Table 97.

Era	Data Definition	Exposure	Nature of Structure	Nature of Relations
1970's	3rd generation data structure	Closed	Weak structures	Implied
1980's	SQL	Closed	Tight structure	Expressed
1990's	Semi Structured (XML, XSD)	Exposed	Weak structures	Implied
2000 to date	Ontology (RDF, OWL)	Exposed	No structure	Expressed

Table 97: Temporal Relations in DDL

The current state of DDLs does not facilitate internal changes to the representation of the world view of computerized information systems resulting from sensing an external world view. At best, data integration results in data aggregation, rather than foster evolutionary learning, an important property shared by complex adaptive systems. Data aggregation does not create new complexities that should emerge from interactions of simpler components, such as individual and independent data structures. CAS interact rather than just add together two or more identical pieces of data. Contemporary DDLs were designed to support automatic linkage among heterogeneous data structures across system borders, according to their creators. However, the data gathered and analyzed

hereto demonstrates empirically that XML, XSD, DTD, OWL, and RDF do not support such interaction. Regardless of the hyperbole generated mostly by economical motivation, no DDL examined hereto even comes close to being morphogenic. No DDL, either as a local schema or as a global schema is capable of autonomous recombination of existing and new data structures that result from interacting or integrating data structures. A global schema may increase the number of data elements it supports, but this is merely an aggregation of symbols rather than genuine evolution leading to growth in complexity.

A true system evolution is based upon recombination of existing and new data structures. In every interaction integration systems use existing patterns with some added variations. An evolutionary integration system should be able to recognize the patterns, experience the difference, and choose to reconstruct them or construct new patterns. Evolutionary integration systems should generate novelty without abandoning the best elements of their past. Evolutionary integration systems should be resilient; that is, flexible and open to "learning" (adaptation) in order to evolve while being durable and consistent with their schemas.

Social and technical processes are linked and interdependent. Technology is shaped by human engineers, economical forces and, consumer demand, to name a few factors. Social change is shaped, or at least influenced by, technology. Case in point – the incredible advancement in human connectivity due to progress in computing; the internet, cell-phones and other mobile communications devices have altered the social landscape in education and dating, to name only two well recognized phenomena. These socio-technical changes are morphogenic – exhibiting actual increase of complexity and decrease of their local entropy.

Chapter 9 explains how entropy is a measure of order. The very high levels of entropy present in different DDLs has not changed through the years. Relating to the second research question, this is additional strong evidence that no real progress towards automatic integration has been made in the design of DDLs, as real progress should have manifested itself in noticeable change in entropy. The fact that standards have perfect entropy H=1 while the other DDLs are slightly below that level indicate the possibility of chaotic behavior similar to that found in other dynamic systems. That is, some values produce tranquil and predictable behavior, while others, only slightly different in their magnitude, produce unpredictable behavior, as illustrated in Figure 86. This is left for future research, as it is way out of this work's scope.

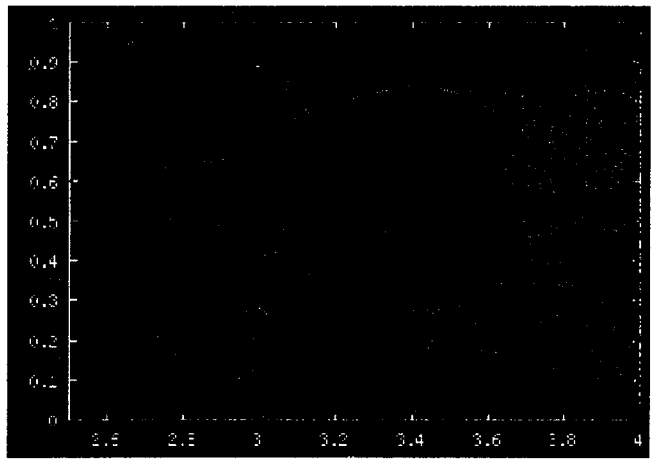


Figure 86: The Logistic Equation - predictable and unpredictable values

13.3.1 Variety and DDL

The paradigm underlying the evolution of adaptive systems increasing in their complexity begins with the fact of a potentially altering surroundings characterized by variety with constraints, and an existing adaptive system or organization whose perseverance and elaboration to higher levels depends upon having achieved mapping of some of the environmental variety and constraints into its own organization on at least a

semi lasting basis (Buckley, 1998). Therefore, adaptive systems – whether on the biological, psychological, or socio-cultural level – must manifest the following attributes (Buckley, 1998):

a) Some degree of plasticity vis-à-vis its environment such that it carries on a constant interchange with environmental events.
b) Some source of variety to act as a potential pool of adaptive variability to meet the problem of mapping new or more detailed variety and constraints in a changeable environment.
c) A set of selection criteria against which the variety pool may be sifted into those variations in the system that more closely map the environment and those that do not.
d) An arrangement for preserving and /or propagating these "successful" mappings.

Computer systems use DDLs to define characteristics of their environment. Data engineers use natural language to create data structures. The natural language, an ever evolving social instrument, provides the necessary external variety data engineers have at their disposal. The variety available has its own characteristics which Zipf discovered, in the form of distribution of words and distribution of meanings. This alone suffices for the creation of a variety of data structures. In addition, each organization, even within a tightly controlled vertical market, may have its own interpretation of its environment, thus forcing the creation of data structures perhaps similar but not identical to others. This is the plasticity vis-à-vis its environment – a required attribute of a CAS. Data structures are the source of potential pool of adaptive variability. An integration system provides (at least in theory) a set of selection criteria to map some of the environment (other data structures). However, integration systems do not assume that role. It is still left to humans to intervene in any data integration process for it to be "correct" and for the new structure to preserve the mappings. DDLs excel at carrying the variety of natural

language and of data structures, but they lack the necessary characteristics to facilitate integration, which is the evolvement of an adaptive system. Apparently, no new layer of technology, usually expressed as a new form of DDL (e.g. the move form XML to DTD to XSD to RDF to OWL) facilitates adaptation. The variety is shifted from one layer to the next, adding new variety and syntactic constraints, rather than creating new persistent mappings to achieve complexity autonomously. A direct measure of shifting the variety from one layer to the next is Zipf's distribution of words and distribution of meanings. This research has demonstrated there is no difference in the Zipf distributions across generations of DDLs, and specifically there is no change in Zipf distributions within the DDLs that were designed with data integration in mind. Added layers of DDLs that do not facilitate automatic mapping move the system from a relative organized simplicity towards chaotic complexity. For example, an environment that uses XML linked to a XSD that points to an OWL ontology can be either very simple, if all three components use the same terminology (thus – there is no variety and CAS cannot exist) or there is chaotic complexity, because the XML (or XSD) doesn't map directly to the OWL; it requires intermediary mappings that are not guaranteed to exist, as explained in the related work chapter.

13.3.2 The Law of Requisite Variety and DDL

Chapter 6 introduced Ashby's assertion that only variety can control variety. Applied to information systems, a complex IS that encounters a change in the environment and needs to respond to it such that the IS does not malfunction or halt, must have a regulation mechanism with enough variety to respond to the change correctly. Changes in the environment require some regulator that has sufficient variety in itself to respond to

the change. Currently, standards and other data structures all depend on humans to be their regulators. Ontologies may have been perceived as potential regulators, but their design and functionality appears to have ignored the law of requisite variety, which must be satisfied in order to create and maintain mapping tension. Ambiguity resolution strategies described in chapter 5 can be viewed as attempts to build cybernetic regulators that are external to DDLs. These strategies do not resolve ambiguities properly, indicating they do not satisfy the law of requisite variety.

Ashby pointed out that with most problems continued repetitive action will not lead to better results. It appears this is exactly what has happened to the evolution of DDLs. Designed for interaction with the environment, and specifically in support of the still illusive Semantic Web, even the latest DDLs do not have characteristics different from their predecessors. This is especially true in relation to the law of requisite variety. In principle, there are two methods to satisfy the law of requisite variety when it comes to automatic integration of autonomous and heterogeneous data sources. One is to lower goals and expectations. It might not be desirable, but it is an option that should not go unmentioned. Commitment to standards is a strategy that lowers expectations, reduces heterogeneity, and achieves valuable results for businesses. A different approach is to increase the power of the regulator until it is able to deal with the complex variety faced by integration. The next chapter will discuss that in more detail.

13.3.3 The Law of Requisite Variety and Approaches to Data Integration

Chapter four (pages 58 - 81) describes several different strategies to data integration. This section briefly analyzes the main approaches from a law of requisite variety perspective.

Standards Based Data Exchange. The common theme among the implementations of standards based data integration is the creation of a regulator that is external to the integration process itself. A regulator, such as EDI, SWIFT, and the Metabase project, does two things: first – it constraints the environment's infinite variety. Second, it is a regulator that is capable of transmitting all that variety, therefore satisfying Ashby's law of requisite variety.

Data Exchange Using Negotiated Agreements strategy differs somewhat from the standards based exchange in the manner in which agreements are reached and imposed. However, from a CAS perspective it is identical to the standards based exchange. Its end result is a regulator that that is capable of transmitting all the permissible variety, thus satisfying Ashby's law of requisite variety.

Wrapping data sources with XML was viewed as a semantics-reducing regulator. However, all it achieved was the reduction of syntax variety (number of DDLs) exposed to the environment for integration purposes. That is, the number of allowable syntaxes was reduced to one. IT enabled the creation of a common input-output mechanism (e.g., read and write) that will not break down due to unknown syntax, or noise in information-theoretic terminology. This approach was the main catalyst behind the creation of multiple XML parsers, such as IBM's XML4J, Sun's Project X (part of the Java programming language and virtual machine), Microsoft's MSXML, Oracle's XML Parser for Java, and many Open Source XML parsers. Each parser claims the ability to traverse any XML file. After much hyperbole and fanfare, the computing community realized that XML did not reduce variety as it was supposed to. This should not be a

surprising result, because XML apparently was not build with the law of requisite variety in mind.

Annotating data sources with ontologies: computerized ontologies DDLs, such as RDF and OWL were suggested as semantic heterogeneity regulation mechanism. Similarly to XML, what they really offered was expanded DDL variety and an additional operational layer that does not address the law of requisite variety for semantic heterogeneity. The new mechanism put even more fuel in the creative minds of data modelers and standards would-be builders. Hundreds of new ontologies have been created, adding to the already vast variety of data structures and meta-data repositories. At best, ontologies can act as a constraining mechanism similar to a standard. However, computerized ontologies do not have the restraints and constraints imposed by standards creating bodies, and the clout such bodies have to keep all their users from deviant behavior, such as the introduction of noise into the standard. After going through its own hyperbole cycle, the computing community realized that ontologies did not reduce variety as they were supposed to. This should not be a surprising result, because ontologies apparently were not build with the law of requisite variety in mind.

13.3.4 The Law of Requisite Variety and Semantic Heterogeneity Resolution

Chapter five (page 82) discusses in detail several approaches to resolving semantic heterogeneity automatically. Each one of the approaches, implemented in projects such as TSIMIS, WHIRL, COMA++ and others attempt to create a regulator. The method favored by their creators is computerized reasoning about the meaning and resemblance of heterogeneous objects in terms of either their linguistic or structural representation, or a combination of the two. None of the approaches addresses the law of requisite variety.

None makes an attempt to create a regulator in Ashby's sense. All approaches create and then depend on some repository whose variety is a derivation of the variety it encounters. Therefore, the repository always remains a sub-set of the variety it attempts to mediate (regulate). These regulators do not have variety that matches or exceeds the systems they attempt to regulate, which is required of them to function as intended, according to the law of requisite variety.

13.4 Practical Ramification

The need to automate data integration created three types of solutions, to date. One is the resolution of semantic heterogeneity, structural dissimilarities and other problems through the use of standards. The second approach is the creation of new DDLs in the hope that these will create a regulator that can manage the environment's variety well enough to allow for automatic data integration. The third approach is the attempts to develop algorithms and other software-based mechanisms to act as regulators.

It appears that only the first approach enables mass data integration automation, and even that requires human intervention at the mapping phase between applications' data structures and standards. The second approach did not address the law of requisite variety, and the third attempts to create a universal regulator without regards to the need to constraint the variety.

Practitioners and researchers alike would be more productive and perhaps more successful if they refrained from repeating their predecessors' solutions shortcomings. Such a bold move will also spare industry and academia the need to comprehend new seasonal technological windows-dressing in the form of fresh vocabulary that is full with hot air yet void of real change (let alone progress), as chapters 11 and 12 have

demonstrated. To keep them busy and perhaps develop a working solution we suggest an approach that has a variety constraining mechanism and an adequate regulator. Unlike standards, the constrained variety is quite large, flexible, and not pre-defined with little hope for amendments. This new approach is explained in chapter 14.

13.5 Chapter 13 Summary and Implications to the Research

At different times, a particular niche of a technology might lag, creating a reverse salient in comparison to progress that was achieved in coupled technologies. Computer hardware and networking are two areas where tremendous complex evolution has been achieved. In contrast, there are principle attributes of CAS shared by all generations of DDLs that remain invariant over the years. The chapter proposed a method for measuring CAS attributes in DDLs for the purpose of evaluating if new DDL that may be proposed in the future actually differ from their predecessors. Used properly, the method can indicate if required CAS properties are better satisfied, compared to previous DDLs.

The chapter applied the law of requisite variety to the analysis of existing approaches to automatic data integration. The analysis reveals that none of the approaches satisfied the law of requisite variety; all of the regulators described in chapters 4 and 5 do not have sufficient variety in them. It appears that a drastically different approach to creating DDL is needed if the law of requisite variety is to be satisfied. The following chapter proposes an idea in that direction.

CHAPTER 14

TOWARDS IMPROVED CONSTRUCTS FOR AUTOMATIC INTEGRATION

Having exactly one meaning per word resolves all ambiguity problems, a root cause for impeded automatic integration. Removing ambiguity is attempted by mechanisms such as statistical guesswork, data dictionaries and ontologies. This approach has severe limitations: the number of concepts (size of vocabulary) in a given data dictionary or ontology or lexicon is fixed at any given time, thus eliminating the possibility of any new variety to occur, which renders the existing "solutions" being a closed system with open system aspirations only. A second limitation is the need for a vocabulary commitment - agreeing that an ontology properly represents a domain's conceptual elements and their relations. Unless participating sources annotate their data structures such that eventually they point to the same ontology, the integration challenge is shifted from the XML or XSD layers to the RDF or OWL layer, and from there to humans, adding complexity without real benefits vis-à-vis automatic integration.

A data definition language that is designed for automatic data integration of heterogeneous sources should satisfy some CAS characteristics: ability to selectively map to the variety presented by a system's environment; autonomously maintain the mapping as long as needed; ability to add, remove or update its own elements and relations dynamically. This requires the ability to build a regulator that has at least the same variety it needs to regulate, such that it satisfies the law of requisite variety. In CAS terminology, the DDL should be able to "make sense" of some of at least some of the variety in the environment by means of some mediator (regulator), create tension and

sustain it (preserve meaning). To "make sense" it should have a perfect disambiguation mechanism, or be build on foundations with no ambiguity and therefore perfect entropy (H = 1).

The creation of a data structure is an exercise in "ontological classification... organizing a set of entities into groups, based on their essences and possible relations"(Shirky 2004). "Classification involves the orderly and systematic assignment of each entity to one and only one class within a system of mutually exclusive and non-overlapping classes; it mandates consistent application of these principles within the framework of a prescribed ordering of reality"(Jacob 2004). Data structured are controlled vocabularies of canonical terms for describing every concept in a domain. DDL do not impact at all the very essence of the variety present in data element names. A DDL is nothing more than a packaging method. One can have a rather abstract data structure in mind, and then express it in either COBOL or XML in a manner that the two are identical, except for their syntax rules. This is clearly shown in Figure 87. All DDLs examined in this research, with no exception, allow for the same type of natural language ambiguity to exist. Perhaps a drastic change in the essence of the signifiers' variety is called for, rather than yet another attempt at wrapping signifiers in a "novel" manner utilizing neologism for camouflage. The most advanced knowledge representation approaches are the other end of this spectrum – attempting to represent every type and piece of knowledge under the sun in some canonical form. This approach requires a regulator that is theoretically infinite in its variety capacity. Suffice to note that knowledge representation methods have not delivered a panacea, to put it mildly. This research advocates lowering expectations, and limit the regulator to handling an

enumerable set – variety that can be generated using some relatively simple derivation rule. Such a regulator should be a canonical control system, one that is completely reachable and completely observable.

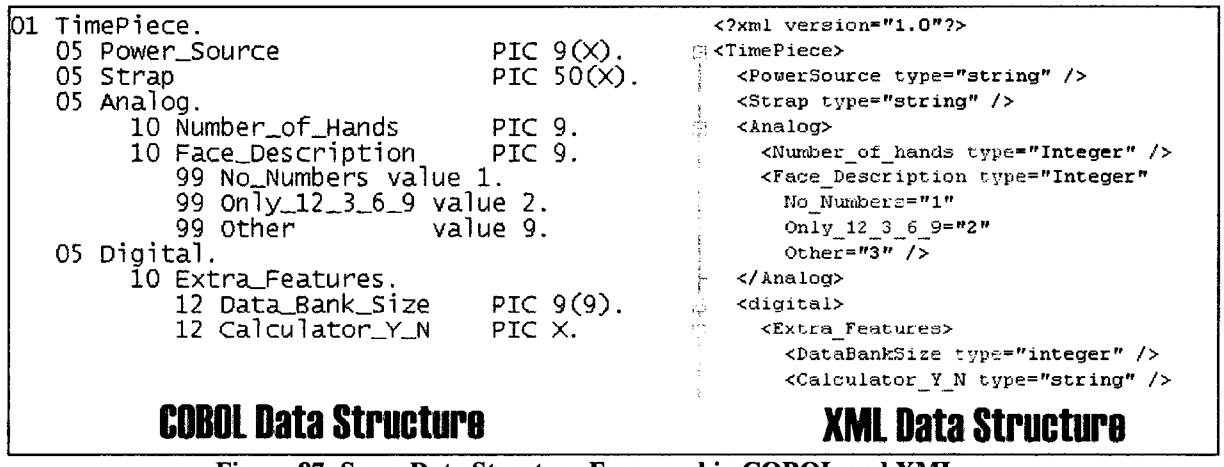


Figure 87: Same Data Structure Expressed in COBOL and XML

One existing implementation of a canonical system that comes to mind is the symbolic language of chemistry. Admittedly, chemistry is not expressive enough for all the needs of contemporary chemists, but it is a working solution, and for us it offers an intriguing idea for a DDL that supports automatic data integration and satisfied the law of requisite variety. The periodic table offers unambiguous building blocks that by themselves have meaning for chemists and make some sense of the world. There are fewer than 100 elements in Mendeleev's periodic table, and they suffice to describe all known matter in our universe. Using a set of rather simple rules, one can combine two or more building blocks (atoms) to create new concepts, namely molecules, in chemistry. As long as the rules are understood and followed, a transmitter of information can create a new concept that doesn't exist up to that point and the recipient will be able to understand the concept using the same rules that created the new concept. For example, oxygen and hydrogen ("O" and "H" respectively) are two such building blocks, and each carries

meaning. Their combination H_2O is "legal" according to the rules of chemistry. The new concept, that doesn't exist in the periodic table, is understood by anyone knowledgeable about chemistry. The same person will reject the concept $H_{2.5}O$ (2.5 particles of H) because the newly created complex concept violates the set of rules.

14.1 The GlossoMote Language Characteristics and Evaluation

This section addresses the third research question: can a better DDL be built? Our objective here is to suggest a mathematically sound solution for maximizing the effectiveness / efficiency of DDL, and use this to design a "GlossoMote DLL" in a way that achieves or nearly achieves the said goal. We predict that such a solution will permit fully automatable integration. We propose these axioms and rules:

- An atomic vocabulary of tokens having zero redundancy (maximum entropy)
- Every atomic token has exactly one meaning.
- Atomic tokens concatenation is allowed, for the creation of more complex concepts
- To convey maximum amount of information, any message X should have H(X)=1 (maximum entropy)
- Meanings of the tokens can be represented as a 1 to 1 mapping from symbol to meaning and from meaning to symbol.

14.1.1 Intuitive Proof

As intuitive proof we demonstrate the implementation using a *GlossoMote language* comprised of ten tokens or building blocks. Unlike EDI & SWIFT, one can combine any number of tokens to create a meaningful data field (or XML tag). For our purposes no repetition of tokens is allowed (nor necessary). The following are characteristics of the GlossoMote language:

Size of alphabet: 10 atomic tokens

Number of meanings per token = exactly 1

Atom repetition: atoms cannot be repeated in a token

Entropy = 1

The number of permissible combinations for ten tokens when picking a subset of *k* tokens from the 10 available, without replacement, and without regard to the order in which the tokens of the subset are placed (or picked) is denoted as ${}_nC_k = n!/(k!(n-k)!)$ Since k may take any value [0,10], the total number of possible expressions is: $1+\sum_{k=1}^{10} n!/(k!(n-k)!)$ (1 is for the empty expression). This gives 1024 meaningful and unique complex expressions.

The possible variety is constrained by a simple set of rules; its scope (number of members in this set) can be calculated and each one can be generated at any given time. This can create a regulator that matches the variety of any data structure implemented using the GlossoMote language rules and axioms. Such a regulator satisfied the law of requisite variety, something not achieved to date by any other approach addressed in this research.

There is no need for a data dictionary or an ontology to "make sense" of newly created concepts using atomic tokens. There's no need for a complex lexicon such as WordNet either, thus reducing the complexity and effort of disambiguation, relation guessing and erroneous mapping. Having one meaning per concept satisfied the most demanding of Sowa's requirements – having a mapping from a symbol *s* to a meaning *m* and vice-versa: $f(s) \rightarrow m$ and $g(f(s)) \rightarrow s$. The GlossoMote language has in fact maximum entropy as the calculation in Figure 88 proves.

Token	Frequency	P(word)	Log2 P(word)	P(word) * Log2 P(word)
SUG	1	0.1000	-3.3219	-0.3322
SDE	1	0.1000	-3.3219	-0.3322
ZMN	1	0.1000	-3.3219	-0.3322
RUAH	1	0.1000	-3.3219	-0.3322
REUT	1	0.1000	-3.3219	-0.3322
WX	1	0.1000	-3.3219	-0.3322
SHAM	1	0.1000	-3.3219	-0.3322
T/TD	1	0.1000	-3.3219	-0.3322
LAHZ	1	0.1000	-3.3219	-0.3322
RMK	1	0.1000	-3.3219	-0.3322

# of Tokens	10	
# of Occurences	10	
H-Maximum	3.3219	$\log_2$ of 10
H-Actual	3.3219	- Sum ($P_i * \log_2 P_i$)
H-Relative	1.0000	(H-Actual) / (H-Maximum)
Redundancy	0.0000	1 - Relative Entropy

Entropy of TOY Alphabet (tokens)

Figure 88: GlossoMote Language Tokens and Entropy

Figure 88 lists the 10 tokens and shows the entropy calculation for the "language". Using any "legal" combination of tokens, one can create a large number of new concepts that should be understood without the aid of a dictionary. The rules are as follows:

- The tokens represent atomic concepts relating to weather (such as location, wind, time, precipitation etc.)
- Location, magnitude, time are expressed in whole numbers, to the right of the token.

Using the rules, a token (which is also a data structure) such as SDE1RUACH100ZMN3 is a legal token, comprised of the following atomic concepts and values: SDE1 RUACH100 ZMN3; To "make sense" of this information one needs to refer to the tokens' equivalent of the "periodic table". Suppose that SDE represents the concept of location, RUACH represents the concept of wind, and ZMN represents the concept of time, then we have RUACH100 (very strong wind) SDE1 (location number 1) and ZMN3 (03:00am). Similarly, RUACH12ZMN23SDE2 means mild wind at 10:00pm at Location "2". The order of the tokens has no meaning, making the language less restrictive.

This approach provides the vital variety for a CAS, provides a mechanism for the creation and maintenance of tension free of human intervention, and has the desired level

of entropy, which is 1. It satisfies the definition of a canonical control system, thus satisfying the law of requisite variety.

14.1.2 Analysis by Lexical Matrix

As a safety measure it makes sense to analyze the effectiveness of the GlossoMote language and compare it using a non-CAS based method, such as a lexical matrix. Signal-meaning mappings in the form of lexical matrixes associate words and meanings (Hurford 1987). Lexical matrixes are not bound to any domain or language. They can be used for signal-meaning mappings between a dog and its handler, jungle drummers and tribesmen, or signal-meaning mapping of humans communicating using natural language. "A lexical matrix is a convenient description of arbitrary relations between discrete forms and discrete concepts. [It] is an integral part of the human language system. It provides the link between word form and word meaning. A simple lexical matrix is also at the center of any animal communication system, where it defines the associations between form and meaning of animal signals." (Komarova and Nowak 2001). Lexical matrixes have been used with languages other than English in the Information Retrieval field (Chklovski, Mihalcea et al. 2004).

Let Q and P be stochastic matrixes whose entries are [0,1] and each of their rows sum to one. Two matrixes define the language of a data source as L=(Q , P) whereas the Global Schema's language L' = (Q' , P'). A *payoff* (Nowak and Krakauer 1999) is the number of objects that can be communicated between the data source and the data receiver weighted by their probability of correct communications.

	A	B	C	D	E	F	G	H	I	J	K	L
1												
2			SUG	SDE	ZMN	RUACH100	RUACH12	M88	M00	D35	LAHZ	TOTAL
3		SUG	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
4		SDE	0.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
5		ZMN	0.0000	0.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
6		Breeze	0.0000	0.0000	0.0000	0.2000	0.8000	0.0000	0.0000	0.0000	0.0000	1.0000
7		Tornado	0.0000	0.0000	0.0000	0.8000	0.2000	0.0000	0.0000	0.0000	0.0000	1.0000
8		Hail	0.0000	0.0000	0.0000	0.2000	0.2000	0.2000	0.2000	0.1000	0.1000	1.0000
9		Rain	0.0000	0.0000	0.0000	0.1000	0.1000	0.7000	0.1000	0.0000	0.0000	1.0000
10		Thunder	0.0000	0.0000	0.0000	0.2000	0.2000	0.2000	0.2000	0.2000	0.0000	1.0000
11		Dust Storm	0.0000	0.0000	0.0000	0.2000	0.2000	0.2000	0.2000	0.2000	0.0000	1.0000

Figure 89: Stochastic Matrix for Transmitter

For illustration purposes, suppose that the transmitter needs to send a message about a tornado (see Figure 89 cell B7), a natural phenomena that is associated with wind. The concept of wind in our GlossoMote language atomic alphabet is "RUACH", which is required for said message. The illustration contains two hypothetic message strings having the token RUACH, and only one with strong winds (RUACH100, cell F7 in Figure 89). To make the illustration real, we allow for some error in the correct creation of a complex token ("molecule") that represents a tornado, hence the 0.8 probability given in cell F7, and 0.2 probability for cell G7.

The recipient of the message receives the complex token SDE1RUACH100ZMN3 and needs to understand it. Isolation of the three atomic tokens allows analysis of the parts of the message using the same rules that created it.

Figure 90 illustrates the recipient's stochastic matrix where cell S15 has the just-received token. The wind component (RUACH) needs to be mapped to a concept available in column "O". We assume an error could occur in the mapping, hence we allocate only 0.8 probability of correct mapping to "Tornado", rather than a mapping to "Breeze" for instance. See cell S20 in

Figure 90.

	N	O	P	Q	R	S	T	U	V	W	X	Y
14	Global	Schema										
15	MATRIX	P'	SUG	SDE	ZMN	SDE1RU ACH100 ZMN3	SDE2RU ACH2Z MN1	REUT3W X4SHA M88	ID5SHA M00	SHAM0T/ TD35	LAHZ	TOTAL
16		SUG	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
17		SDE	0.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
18		ZMN	0.0000	0.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
19		Breeze	0.0000	0.0000	0.0000	0.2000	0.8000	0.0000	0.0000	0.0000	0.0000	1.0000
20		Tornado	0.0000	0.0000	0.0000	0.8000	0.2000	0.0000	0.0000	0.0000	0.0000	1.0000
21		Hail	0.0000	0.0000	0.0000	0.2000	0.2000	0.2000	0.2000	0.1000	0.1000	1.0000
22		Rain	0.0000	0.0000	0.0000	0.1000	0.1000	0.7000	0.1000	0.0000	0.0000	1.0000
23		Thunder	0.0000	0.0000	0.0000	0.2000	0.2000	0.2000	0.2000	0.2000	0.0000	1.0000
24		Dust Ston	0.0000	0.0000	0.0000	0.2000	0.2000	0.2000	0.2000	0.2000	0.0000	1.0000

Figure 90: Stochastic Matrix for Receiver

The demonstration at this point has one string created by a transmitter using its active matrix (Figure 89) -- mapping a concept into a message string. The string needs to be understood by a receiver, using a set of rules and atomic tokens only. This is represented by the receiver's passive matrix P' shown in figure 90.

A calculation of the payoff of the GlossoMote language yields a 60% payoff – about twice as much as the Real Estate payoff which is less than 30% under similar conditions as described in the Data Integration Potentiometer article (Rohn 2006) whose final results are shown in Table 98

I	J	Qij	P'ji	Q'ij	Pji	Qij*P'ji	Q'ij*Pji	Total
1	1	0.2511	0.1445	0.1923	0.2345	0.0363	0.0451	0.0814
1	2	0.1322	0.2933	0.2094	0.3259	0.0388	0.0682	0.1070
...	...							
7	5	0.1549	0.2425	0.2614	0.0964	0.0376	0.0252	0.0628
						Total payoff		1.4339
						Maximum payoff		5.00
						% Payoff		**28.6800**

Table 98: Payoff Calculation (truncated) for RETS and MISMO (Rohn 2006)

If we were to remove the assumption that a mapping error may occur the language yields 100% payoff, as one would expect to see in chemical formulas. We attribute the payoff improvement of the "GlossoMote language" to the fact that we did not create yet another DDL but rather, addressed the reverse salient that all DDLs carry through their

generations, namely – semantics; we use a technique that has been very successful in the scientific field of chemistry. The technique fosters the creation of new complex concepts based on a well defined finite set of atomic concepts. The technique allows for morphogenic evolution of concepts while providing for meaning interpretation based on rules rather than on pre-defined lexicons.

14.2 Analysis using CAS Framework

The evaluation framework suggested in this research investigates three CAS characteristics – Variety, Tension, and Entropy.

14.2.1 Variety

The GlossoMote language "system" provides for constrained variety – a desirable characteristic as it allows for the creation of a regulator that satisfies the law of requisite variety, and satisfies the business need to formally express their view of themselves and their environment. Data modelers can create a large variety of dissimilar constructs; yet the rules of creation limit them to the creation of constructs that can exist in the universe defined by the "periodic table" of that universe.

14.2.2 Tension

We evaluate tension's existence by means of possible bijective mapping between two data structures or their subsets. The manner in which structures are constructed in the "GlossoMote language" precludes the possibility of creating an element (either simple or complex) that is expressed in more than one way. That is, the GlossoMote language set of rules constraints the creation of new concepts (elements or structures) to only unambiguous and unique expressions. Using the aforementioned token "RUACH100"

the set of rules do not allow expressing same as "RUACH10+ RUACH90", or "10RUACH10" for example. Similarly, if "MROTS" is not present in the set of atomic symbols, then "MROTS55", for example, would not be a legitimate symbol, and therefore would not need to be mapped, because it cannot and does not exist in that universe. In other words, the GlossoMote language is a Canonical System that is completely reachable and completely observable – a meaning preserving system, as explained in section 13.2.4.

Hence, every concept created from the set of elementary symbols in accordance with the system's rules is assured to have unambiguous meaning and assured to be mappable to its counterparts in other sets of concepts derived from the same alphabet (symbol set) and rules. Therefore, the GlossoMote language mathematically guarantees a bijective mapping will exist for every well-formed concept. In other words – if there is a need to create tension – it can be created and maintained. Further, such a system reduces tension to a binary state, precluding the need to address varying degrees of tension.

This is a major departure from the longitudinal tension pattern discovered and discussed in this research. The departure is in fact an improvement over existing methods, because meaning preservation is essential for automatic data integration that aspires to be correct and complete.

14.2.3 Entropy

Figure 88 shows that in fact entropy equals to 1, which is a departure from the typical entropy in existing DDLs. It is a desirable result, for several reasons: first, standards (e.g., EDI, SWIFT) have such entropy, and standards work well. Second, entropy of 1 is the

most efficient communication channel, also a desirable characteristic as it implies no redundancy.

14.3 Folksonomy as a DDL Alternative

This research has shown that controlled vocabularies and their relations (e.g., data structures) for describing concepts in a domain created by experts have CAS characteristics that are unfavorable to automatic data integration. A need to derive semantic knowledge by consensus agreement and the need to develop tools to support such collaboration was identified as early as 2001(Behrens and Kashyap 2001). They suggested linking formal semantics with deeper meaning as reflected by consensus discovered among users on the Semantic Web. Such collaborative tagging by non-experts became known as Folksonomies. Unlike taxonomies or data structures, Folksonomies are a flat tagging system that can identify objects as being many things simultaneously. "Collective tagging has the potential to exacerbate the problems associated with the fuzziness of linguistic and cognitive boundaries"(Golder and Huberman 2006). Folksonomies are typified by having a large number of users, lack of central coordination, and non linear dynamics (Halpin, Robu et al. 2007). Such systems are known to produce over time power law distribution. Such distributions produced by complex systems are often scale-free: regardless of how larger a complex system grows, the distribution's shape remains stable.

Our analysis (see section 14.3.3 on page 263) shows that social-tagging yield results having different CAS characteristics than controlled vocabularies. Folksonomies could be more favorable to some specific application of data integration.

14.3.1 Methods for Creating Folksonomies

Folksonomy are tags and annotations created by consumers and creators of Web content, either in isolation or collaboratively. For example, the website http://del.icio.us encourages its members to annotate (associate) their Web bookmarks (URL's) with one-word descriptors ("tags"). A user may assign multiple tags to a single bookmark. Users can view the most popular tags assigned to the bookmark, as well as the number of users who have bookmarked it and some other users' talk-back on the content of the bookmark (DELICIOUS 2007).

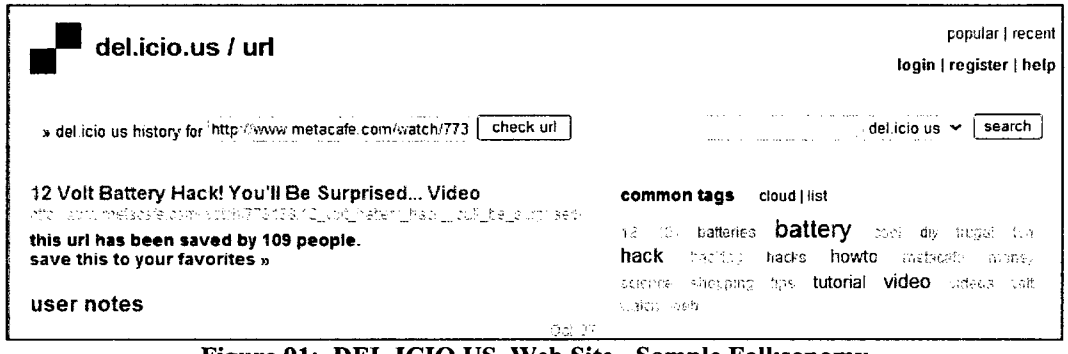


Figure 91: DEL.ICIO.US Web Site - Sample Folksonomy

Figure 91 is a snippet of the DEL.ICIO.US website. It shows on the left a bookmark and on the right common tags associated with that bookmark. The tag's font size is proportional to the tag's popularity. Tags are created by individual users who are not required to collaborate in the process of the tagging. A more collaborative tagging approach is exemplified by ESP GAMES, where two randomly selected players who view an arbitrary image served to their respective browser and they need to agree on a common tag that describes the word(von Ahn 2003; von Ahn and Dabbish 2004).

Folksonomies could be created and mediated by data modelers using recommender systems built for that purpose. Such an approach may yield data structures annotated with Folksonomies. The annotations themselves could be part of a regulator mechanism that

satisfies the law of requisite variety, because they are likely to have more variety than the data structure itself. As the Folksonomy grows so does the variety of the hypothetical regulator, and this is a desirable trait. The idea of data structures annotated with Folksonomies has not been proposed nor tested in our literature review of the subject. This approach requires additional research to determine if it has the potential to better facilitate automatic data integration.

14.3.2 Statistical Characteristics of Folksonomies

A longitudinal study of users and tagging on delicious.com revealed that the number of tags in each user's tag list obeys power laws. "One might expect that individuals' varying tag collections and personal preferences, compounded by an ever-increasing number of users, would yield a chaotic pattern of tags as time goes on. However, it turns out that the combined tags of many users' bookmarks give rise to a stable pattern in which the proportions of each tag are nearly fixed" write Golder and Huberman. Figure 92 clearly shows a Zipf distribution, similar to what we have obtained for the various DDLs examined hereto. Golder's results were obtained from a study of 226 users and 68,668 bookmarks. Very similar results have been obtains a year later by (Cattuto, Loreto et al. 2007; Halpin, Robu et al. 2007). Folksonomies may have additional uses (and potential abuses) because "over time, users' lists of tags can be considered descriptive of the interests they hold as well as of their method of classifying those interests" (Golder and Huberman 2006).

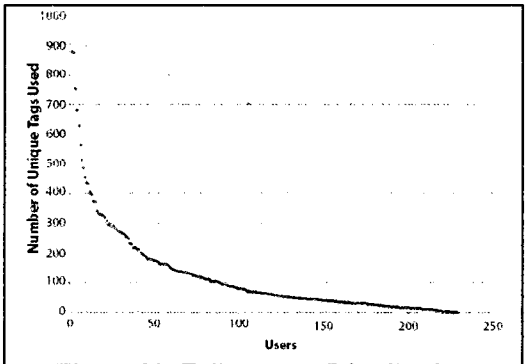


Figure 92: Folksonomy Distribution (Golder and Huberman 2006)

14.3.3 Folksonomy Analysis using CAS Framework

Variety: The aforementioned Zipf distribution reported in three independent research efforts is indicative of high variety, a characteristic that is typical of complex systems in complex environments. As explained in chapter 6, variety is essential for the viability of CAS. The variety found in Folksonomies is identical to that found in DDLs examined in this research.

Tension: every tag in a Folksonomy is mapped to an object. Although not all tags are synonyms, the tags, especially those with high agreement, signify the same "thing". Even if older tags are replaced by new ones, tension is not lost. However, the mapping is usually many (tags) to one (object), which could be a limitation rather than a positive trait. Folksonomies appear to preserve meaning almost ideally, if only the mathematical mapping is taken into account. However, tags obtained through the ESP Game may relate to only one small artifact or the tone of a picture, rather than the entire object. For example, Figure 93 has a tag entitled "USO" and another one entitled "Banner", each pointing to a single artifact inside the image. The tags lack data about the region to which they apply. This poses new challenges that need to be overcome by additional research and professional implementation.

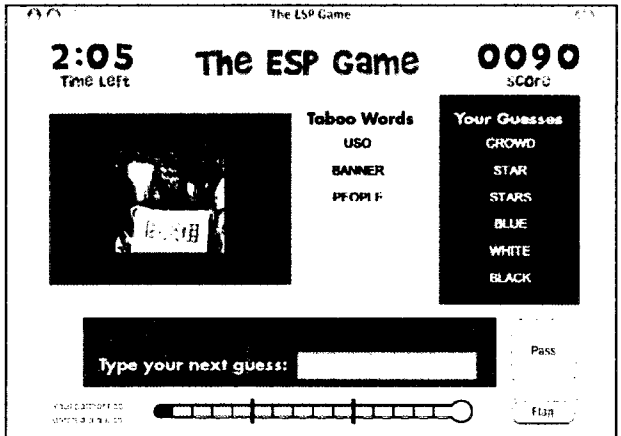


Figure 93: Screenshot from the ES Game

Entropy: it appears that entropy depends entirely the tagging system's design. The ESP Game' *Taboo Words* is a list of previously agreed upon tags, and each tag is unique, as

the case is with SWIFT or EDI. However, lists created by systems such as Delicious allow the same tag to be used multiple times by various users, creating entropy that is similar to the one found in DDLs. Since entropy of 1 is ideal per the analysis of this research, Folksonomy systems designed with automatic data integration in mind can be and should be designed such that they will produce entropy of 1.

14.3.4 Folksonomy Conclusions

Folksonomies exhibit variety; Folksonomies create strong tension between the object being tagged and the tag itself, hence are ideal for a one-way meaning preservation; Folksonomies can be designed to have entropy of 1. A set of tags associated with an object is finite, and all the set's elements are observable and independent of each other, implying they could be a canonical system if designed properly. The significance and implications of canonical systems are discussed in 6. The CAS attributes of Folksonomies indicate they could be an improvement over existing DDLs for the purpose of automatic data integration. Determination if this direction may bear fruit, necessitates additional research to determine if it is possible to create a regulator algorithm that satisfies the law of requisite variety for Folksonomies. If such regulator can be created, then Folksonomies could replace or at least augment DDL for specific computerized applications, giving rise to fully automatic integration that is complete and accurate.

14.4 Chapter 14 Summary

Folksonomies may seem similar to standards, because a group of humans produces each one through consensus. The similarity is superficial, because standards (EDI, SWIFT, etc.) are closed systems that do not easily change. In contrast, Folksonomies are dynamic open systems that can change at will. Unlike standards, Folksonomies may have various

levels of entropy, that solely depend on the design of the system that creates a Folksonomy. The overall CAS characteristics of Folksonomies are better suited for automatic data integration than any DDL examined in this research.

The GlossoMote languages create finite sets only. The entire set of elements is observable. Elements in these sets are linearly independent of each other, implying that they are completely reachable. Therefore, the GlossoMote language is a Canonical System. Since "the variety in a system's input equals the variety in its output if and only if the system is canonical" (Casti, 1985) we have a mechanism that is analogous to a noiseless communication channel. This implies constrained variety and maximum entropy. The GlossoMote language precludes the possibility that two distinct combinations of "atomic elements" (input sequences) yield the same outcome (state sequence), and vice versa. This implies unambiguous correct and complete mapping potential, which in CAS terminology it is said that the proposed GlossoMote language can create and maintain tension. Therefore, the GlossoMote language is fundamentally different from existing DDLs and their associated technologies and algorithmic corollaries.

CHAPTER 15

IMPLICATIONS, SUMMARY AND FUTURE RESEARCH

It appears this research is the first one to analyze DDLs using CAS theory. The research evaluated thirteen DDLs representatives spanning over forty years of computing history. Measurements of Variety, Tension, and Entropy form the basis for a longitudinal qualitative and quantitative analysis. These three fundamental constructs are static over time, defining a clear reverse salient in computing. The study has shown that even DDLs designed with automatic integration in mind did not improve the potential of automatic integration, suggesting that human intervention at a certain point in the data integration process is responsible for any implemented integration. Automatic methods of integration attempted to create regulators (mediators), but none satisfied the law of requisite variety, therefore deferring some or all the resolution of semantic heterogeneity to human operators.

15.1 Answering the Research Questions

Several implied answers to the three research questions have been given in previous chapters, along with partial answers to same. The purpose of this section is to clarify the implied answers, make an overt connection between them and the findings and analysis. This section provides answers to all three questions in one place.

The theoretical necessary requirements for a DDL to fully support automatic data integration from autonomous heterogeneous data sources are three:

1. *Availability of a regulator that satisfies LRV.* Satisfying a data integration goal requires mapping between a system's data structure and external data

structures. To create such mappings it is necessary to have a mediation mechanism ("regulator") that can automatically create some isomorphism between a given data structure and other data structures of interest. To do so the mediation mechanism must satisfy LVR.

No approach reviewed in this work is successful at creating mappings without satisfying LVR. That is, there is no finding that may be grounds to reject the requirement. Approaches to building a regulator, such as those mentioned in sections 4.3, 4.4, 4.5, 13.3.3 and 13.3.4, indicate that the need to create an LRV satisfying regulator is not fully understood. The design GlossoMote provides an example of a DDL that satisfies LRV, hinting that such a novel approach is feasible.

2. *Ability to create and sustain tension* between (partially) mapped data structures. Tension is created through isomorphism that preserves meaning. For automatic data integration to occur, such a mapping needs to be sustained until such time that it is no longer required. The existence of an $f(x)$ and its inverse $g(f(x))$ have been assessed by attempting to map each data structure to at least one other data structure in the sample. Numerous data integration approaches reviewed in this work. None has a mechanism to automatically create tension. None is able to existing sustain tension (created by human intervention) when a mapped data structure changes autonomously. All data definition languages, from the 1960's to date, do not preserve meaning. This includes predefined agreed upon standards, such as EDI and SWIFT.

However, predefined standards are "closed systems", hence, meaning is preserved by isolation from the environment.

3. *Behave like a noiseless communications channel* (e.g., having entropy=1). Uncertainty about the meaning of a field is analogous to noise. The more uncertainty the more the noise. Full certainty implies a noiseless communications channel. Unambiguous standards such as EDI and SWIFT provide a noiseless communications channel. All other DDLs allow usage of natural language, which introduce noise to the system. Redundancy mechanisms, such as a Data Dictionary for a database or ontologies to be used with XML data structures have addeded layers of complications (not complexity, though) yet do not amount to a viable solution.

The answer to the first research question provides the theoretical necessary requirements to fully support automatic data integration from autonomous heterogeneous data sources.

The second research question asks if there has been real advancement in DDL design towards automatic data integration. In other words, do new DDLs progressively meet the theoretical requirements explained above? The answer is a firm "no". Contemporary DDLs do not differ from older DDLs in any of the three key measurements: Variety, Tension, and Entropy:

Variety: All DDLs except EDI and SWIFT exhibit indistinguishable levels of variety measured by Zipf distribution of words and Zipf distribution of meanings.

Tenssion: no DDL has a tension creation mechanism.

Entropy: EDI and SWIFT have entropy = 1. All other DDLs have entropy in the vincinity of 0.9

The third research question asks if there is a better way to approach the design of DDL that fully support such integration. The proposed GlossoMote provides a qualified positive answer: it has a mediation mechanism that satisfies LVR; it has a tension creation mechanism; and, it creates a noiseless communications channel having entropy equals to one.

15.2 Philosophical Approaches Relevant to Future Research

For any data integration mechanism to work autonomously, it needs to satisfy Ashby's Law of Requisite Variety. The mechanism, either part of a DDL or an external component that works in tandem with the DDL (just as a car engine is regulated by a gear box), should have at east the same amount of variety as the DDL allows for. Currently only humans are capable of being such regulators. This is expected to be changed for automatic data integration to be realized.

Churchman's classification of inquiring systems (Churchman 1971) helps place data integration approaches in perspective, and point to possible weaknesses. For example, most XML integration projects are characterized by a need for consensus in the system to be usable. Integration engines attempt to generate a consensus view of autonomous and heterogeneous resources (e.g., the Global Schema). Data integration projects yield results that are incomplete and ambiguous, requiring manual intervention for disambiguation, conflict resolution and incompleteness resolution. Even with manual intervention there might not be a "best" solution, because "best" depends on the specific needs of the

inquirer. Even with a single inquirer, needs may change or there could be multiple needs, each one commanding its own "best" consensus view.

Five Western traditions of philosophical inquiry ascribed to Hegel, Kant, Singer, Leibniz and Locke are identified by Churchman. The Lockean and Leibnitzian are characteristic of the dominant (and limiting) MIS model (Mason and Mitroff 1973). According to the Leibnizian information system approach, there could not be more than one XML schema for real estate, for example. However, as shown in this work, there are multiple viable schemas for real estate, and each one is probably valid from a subjective perspective. Data modeling has been conducted as Lockean and Leibnitzian activity (Hirschheim, Klein et al. 1995). That is, that there exists a single correct all encompassing portrayal of the issue at hand. Hirschheim et al. refer to database design as a positivist activity, and name it *Functionalism*, "a paradigm that is best developed and most proven in applications development". Per Hirschheim et al. the real world is observer independent, measurable and subject to precise definition. According to this approach, data models can be determined by objective process and described in neutral language.

A Kantian information system is an amalgamation of the Leibnitzian and Lockian approaches because it contains both theoretical and empirical components. A Kantian integration system is one that uses axiom about a domain to create a consensus view of XML heterogeneous and autonomous resources in that domain. It is possible to design such systems in certain domains. For example, domains that are closely governed by legal requirements will have a set of given definitions and relations that serve as axioms

to an integration information system. However, other domains are not well suited for a Kantian data integration.

Two principles guide Singer's inquiry to understanding the world. The first premise establishes a system of measures. Its goal is to find the degree to which differences among group member's opinions can be resolved by the measuring system. The second premise is the strategy of agreement guided by ethics that yields new knowledge that is useful for society at large, not juts to the inquirer. A Singerian information system is an exercise in continual learning and adaptation.

Hegel's approach to world understanding is based on the dialectic between thesis and antithesis, leading to synthesis. An Hegelian information system is a one that offers a synthesis of different points of view (Haynes 2001).

Future research into DDL needs to divorce itself from Lockean and Leibnitzian approaches, as these have shown to create a reverse salient in the area of data integration. New research directions need to experiment with DDLs that are derived from the Hegelian approach or from the Singerian approach. For example, DDL that support Folksonomy annotations created by subject matter experts (SME) are expected to evolve constantly, creating new relations within the Folksonomy and removing irrelevant terms in the process. This appears to be a form of learning and adaptation, which is the Singerian approach. An ensemble of data structures created by such a DDL is a complex system. If there are two or more SME groups, each with its own perspective of the data structure, then their input may form a synthesis of different points of view. This would be typical of a Hegelian information system.

Whether Singerian, Hegelian or a combination of both, The SME input provides some of the energy required for the complex system to sustain itself. The SME input augments the system's adaptation to its environment. In return for the investment of energy, the ensemble of data structures becomes viable to human-free data integration.

The SME is not required to be human. It could be a computer application, or even an imaginary (for now) bio-computer. For example, an organic neural mesh attached to a computing device that uses the organic mass to conduct reasoning tasks. One may want to leave to science fiction for now the speculation about multiple such "brains", each with its own epistemology, hooked up in parallel to a Singerian computerized mediator. However, such ideas should not be scorned solely on the basis of technology unavailability.

15.3 GlossoMote Future Research

GlossoMote is suggested as a replacement of DDLs. It is a set of mathematically defined constraints outlining an approach that aims at the core problem common to all DDLs, namely semantics. The new approach mimics some natural phenomena such as DNA where only seed information is available, along with a set of rules to construct more complex bio-info-structures. Our approach is analogous to the language of chemistry, based on the periodic table allowing for the expression of complex molecules that make sense as long as the expressions meet the rules of chemistry. Further research is required in this direction. It appears the approach could be more suitable to simple domains rather than, say abstract fields such as philosophy. Yet it is unclear what domains would lend themselves to such an approach, more than others. Follow-up research may help

understand what are the atomic constructs necessary to support a full fledged industrial solution, and to what size is the solution scalable.

The proposed GlossoMote DDL creates enormous research potential into what would constitute a "good" or an "ideal" quasi-periodic table and its limitations. Can one be created only for specific, limited fields or industries? Is it possible to create a quasi-periodic table that describes the entire universe of tangible assets? How about a quasi-periodic table that is suitable to highly abstract ideas? Or, what are viable approaches to technically implement such a "GlossoMote language" mechanism, and is there a "best way"?

15.4 DDL Design Using a Mathematical Dynamic Model

The process of creating new DDL or improving existing ones is time consuming and consumes expensive resources. Thus it would be useful to create a mathematically sound DDL development framework against which any DDL design can be evaluated and tested before it is implemented even in the laboratory. It has been achieved for databases with the relational model (Codd 1970). Additional research is needed for the evaluation framework to reach a similar level of practical simplicity. This may necessitate the development of a dynamic mathematical model, because data integration appears to be a time-dependent process whose states change temporally. For example, one may think of the integration state $s(t)$ as a function of $s(t-1)$ where s is dependent, among other things, on the changes that occur in the target data structure and changes that occur in the source data structure. The equation $s(t)=f(s(t-1))$ is a simple dynamic system requiring serious development to become an adequate model of DDL suitable for automatic data

integration. Such a model could take into account the changing nature of Folksonomies associated with data structures, where older tags are replaced by new ones.

15.5 Folksonomies and DDL augmentation

The Folksonomies examples given in section 14.3 are limited to lists created by non-experts. Future research is required to assess the nature of Folksonomies created by a group of experts; additional research is required to understand what software configuration creates the best Folksonomies in support of automatic data integration. "Best results" needs to be defined as part of this research direction; the definition should not be deterministic. Rather, it should be Singerian, Hegelian or a combination of both.

The study and analysis of Folksonomies using CAS constructs appears to be an unexplored field of study. The brief analysis of Folksonomies vis-à-vis Variety, Tension and Entropy can be extended further, incorporating not merely the Folksonomies alone but the systems that are built to generate them. Section 14.3.3 (on page 263) hints that different applications yield different Folksonomy characteristics. Hypothesizing which variables are independent and which ones are dependent and the nature of their relations can improve our understanding of how to create Folksonomies that best match their reason for existence, using CAS properties as the bench mark.

15.6 Approach to designing a DDL

It should be noted that the proposed approach is concerned with the design of a DDL suitable for automatic data integration. Other considerations are of secondary importance, if at all.

The process begins with the design of axiomatic facts where the outcome is similar to the periodic table, along with the design of a set of rules for combining elementary facts. In other words, the rules prescribe the creation of relations. Such a mechanism has the ability to generate legitimate combinations. The combination of "things" and "relations" is in fact a system, by definition: $S = (T,R)$ where T represents "things" and R stands for "relations". The creation of such a system could be rather simple for a very small and well defined subject, as demonstrated with GlossoMote. However, for a project whose scope is an entire vertical market (or beyond) this is far from simple, probably requiring an effort similar to that of the Human Genome Project. The result is not a DDL in the sense we've seen to date. Rather, the results will be a computational Ontology. While existing computerized ontologies are a deterministic and finite list of consensus items and relations, a GlossoMote instance is not limited to a consensus enumeration of items, and may have, theoretically, an infinite number of items varying in their complexity.

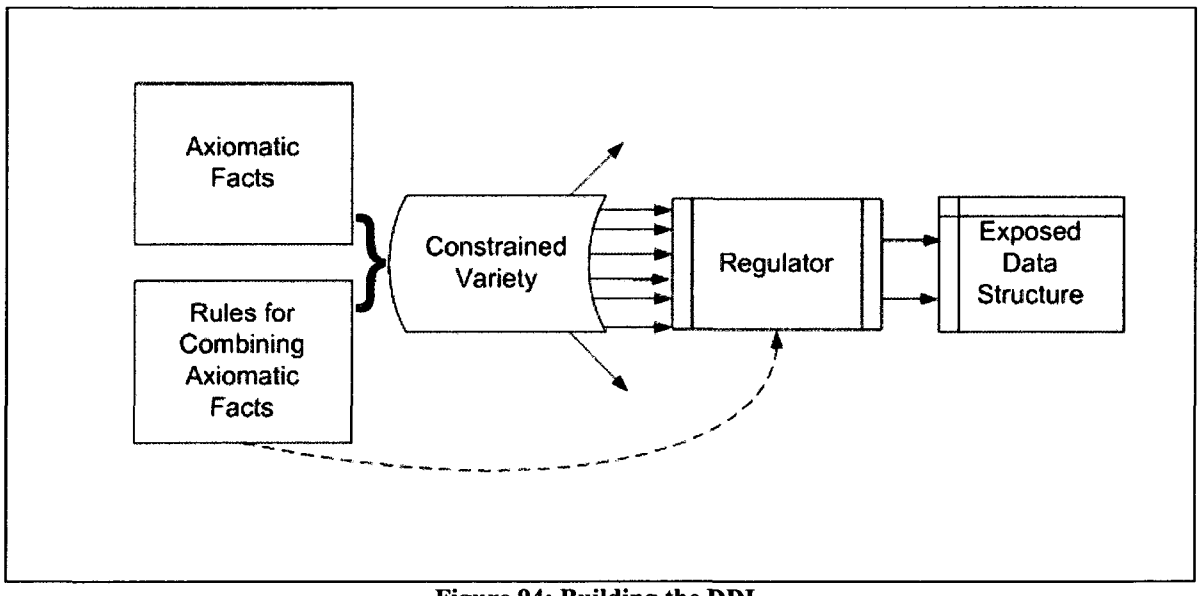


Figure 94: Building the DDL

It is assumed that the information system has data structures built using the created by the axioms-rules system, just as it is reasonable to assume, for instance, that chemists use the periodic table and do not resort to using a different axioms-rules system.

The next step in the process of the DDL is the implementation of a regulator. Its function is to sift relevant compounds from the variety created by the axioms-rules system, and map those to an information system whose scope MUST be smaller than the axioms-rules system so the entire ensemble abides by LRV. It is analogous to limiting the scope of processing to components that have, say, the element silicon, and disregard all other compounds. The regulator must not be part of the information system it assists with data integration. Its main function is map a given variety, by using the rules and axiomatic facts. For example, suppose that the axiomatic facts contain "wind", "velocity", and "temperature"; if there is a rule that permits the combination of a natural number with "velocity" and rule that permits the combination of "wind" and "velocity" then the combination "wind100velocity" is valid, and therefore possible to map to a data structure with a component named, say, "wind90velocity110velocity" which may indicate a range. The regulator will not map "wind100Temperature" to said data element. The regulator needs to calculate the strength of the tension created by a mapping as a mechanism to determine the validity and value of a mapping. Weak tension is better off replaced by a mapping that creates stronger tension when it becomes available.

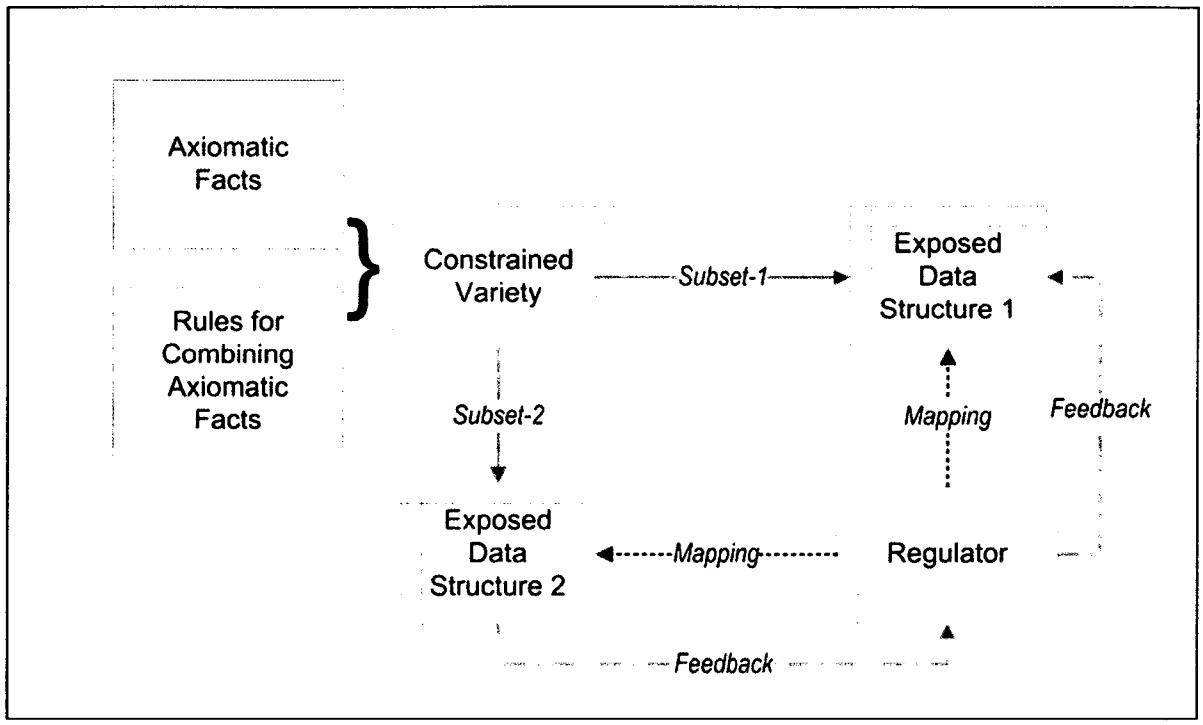


Figure 95: Integration using the DDL

A third potential step is the creation of a FEEDBACK mechanism from the data structure to the regulator, indicating the tension the system senses for a given mapping. It could be first used to assess the accuracy or suitability of the mapping to the system's goals. It could be used as a learning mechanism to improve potential new mappings. It should be used in determining the overall strength of the mapping as a decision factor if the mapping should be left intact or replaced by a better one.

In summary, the engineer whose task is to build a GlossoMote DDL needs to wrestle with the challenge of identifying the axiomatic building blocks. The engineer will probably design a system mimicking the periodic table or the DNA sequence, and the set of relations-creating rules that accompany them. The engineer will then create a regulator that abides by LRV. The regulator's task is to sift potential mappings from the

environment's constraint variety and map it to the information system that needs to integrate data from an external source.

15.7 Thoughts on Futuristic Data Integration

Data integration as understood and implemented in the beginning of the 21st century is probably very different from what it will be in the beginning of the 22nd century. Rather than mapping identical concepts expressed differently, data integration will also include the augmentation of existing data structures as part of the integration process. That is, additional data items and group will be added from the environment's variety. The internal logic of the integrating information system will also have to be augmented, for the augmentation to be meaningful and useful. Therefore, the integration process will involve data structures and logic. It s possible that the "computer program" as we know it today will cease to exist. Rather, the "computer program" of the 22nd century will consist of a component analogous to an orchestra conductor, guiding distributed software services so that the entire ensemble of logic and related data structures interact harmoniously to service the goal of the human or organization owning the process and its outcomes. Such an integration scenario is a true complex adaptive system: the growing number of interactions and relations make the system complex. Its ability to modify logic and data structure to accommodate the changing environment for the purpose of attain a goal set forth earlier is a pure expression of adaptation.

15.8 Final Statement

Assuming that information systems implementation is a rational, deterministic process is unrealistic. Neither people nor information systems implemented by others behave in a manner consistent with an implementer's view of the world (McCartney 1988). There exist several approaches to designing DDLs better suited for automatic data integrations, each one yielding a different outcome. Futuristic DDLs will probably not resemble DDLs as we know them today, just as sequential files on punch cards do not resemble contemporary databases hosted in a grid computing environment. The questions and ideas raised in this chapter will have to be addressed in future research, as they are clearly out of this work's scope.

APPENDIX A: DATA GATHERING ILLUSTRATION

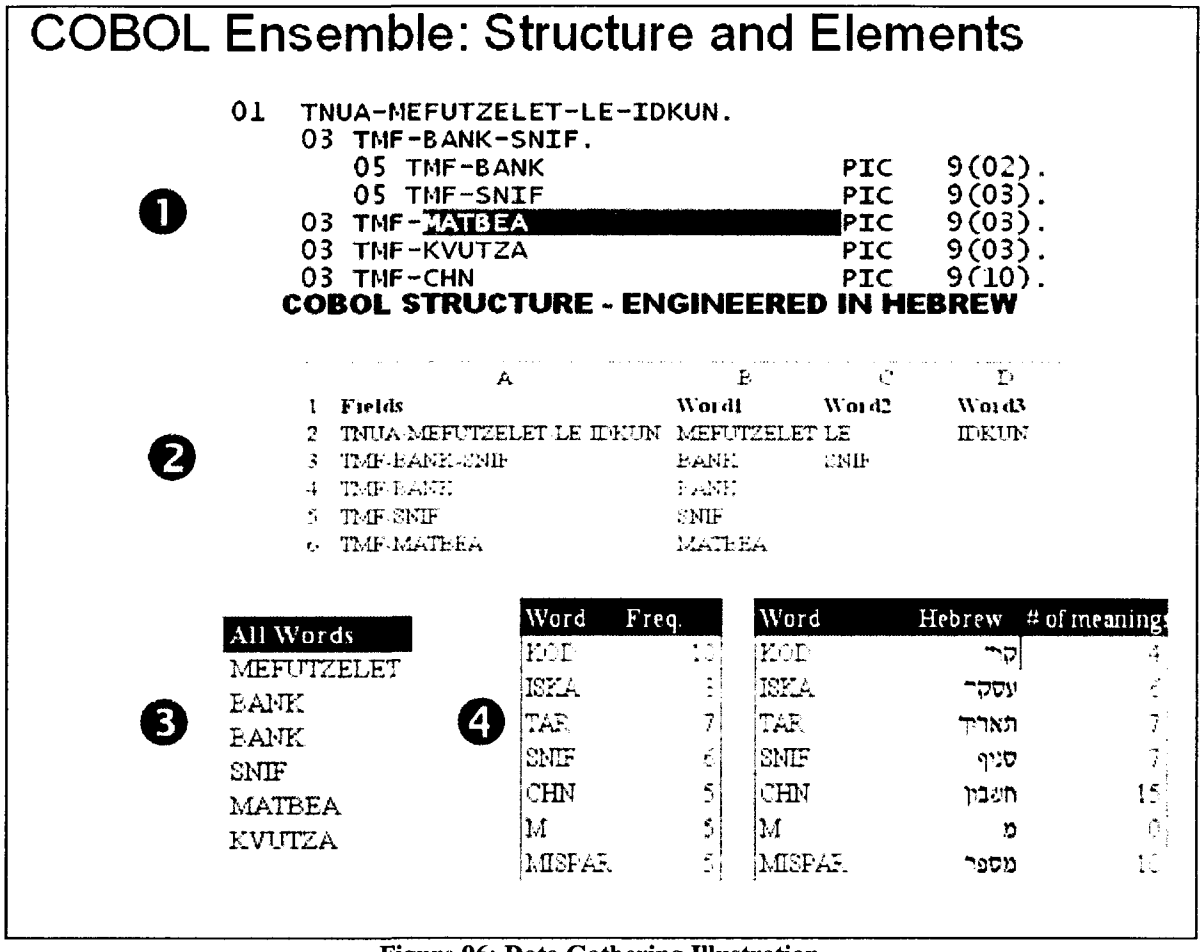


Figure 96: Data Gathering Illustration

Step	Description
1	Identify elements in the data structure
2	Copy all elements into a spreadsheet and break them down to atomic symbols (words)
3	Combine all atomic symbols into a sorted list
4	Count the number of occurrences per symbol and number of meanings per symbol

APPENDIX B: EDIFACT VERSION D.99B MESSAGES

Message ID	Message Description
APERAK	Application error and acknowledgement message
AUTACK	Secure authentication and acknowledgement message
AUTHOR	Authorization message
AVLREQ	Availability request - interactive message
AVLRSP	Availability response - interactive message
BANSTA	Banking status message
BAPLIE	Bayplan/stowage plan occupied and empty locations message
BAPLTE	Bayplan/stowage plan total numbers message
BMISRM	Bulk marine inspection summary report message
BOPBNK	Bank transactions and portfolio transactions report message
BOPCUS	Balance of payment customer transaction report message
BOPDIR	Direct balance of payment declaration message
BOPINF	Balance of payment information from customer message
CALINF	Vessel call information message
CASINT	Request for legal administration action in civil proceedings message
CASRES	Legal administration response in civil proceedings message
COARRI	Container discharge/loading report message
CODECO	Container gate-in/gate-out report message
CODENO	Permit expiration/clearance ready notice message
COEDOR	Container stock report message
COHAOR	Container special handling order message
COLREQ	Request for a documentary collection message
COMDIS	Commercial dispute message
CONAPW	Advice on pending works message
CONDPV	Direct payment valuation message
CONDRA	Drawing administration message
CONDRO	Drawing organisation message
CONEST	Establishment of contract message
CONITT	Invitation to tender message
CONPVA	Payment valuation message
CONQVA	Quantity valuation message
CONRPW	Response of pending works message
CONTEN	Tender message
CONTRL	Syntax and service report message
CONWQD	Work item quantity determination message
COPARN	Container announcement message
COPINO	Container pre-notification message
COPRAR	Container discharge/loading order message
COREOR	Container release order message
COSTCO	Container stuffing/stripping confirmation message
COSTOR	Container stuffing/stripping order message
CREADV	Credit advice message
CREEXT	Extended credit advice message
CREMUL	Multiple credit advice message
CUSCAR	Customs cargo report message
CUSDEC	Customs declaration message
CUSEXP	Customs express consignment declaration message

Message ID	Message Description
CUSPED	Periodiccustoms declaration message
CUSREP	Customs conveyance report message
CUSRES	Customs response message
DEBADV	Debit advice message
DEBMUL	Multiple debit advice message
DELFOR	Delivery schedule message
DELJIT	Delivery just in time message
DESADV	Despatch advice message
DESTIM	Equipment damage and repair estimate message
DGRECA	Dangerous goods recapitulation message
DIRDEB	Direct debit message
DIRDEF	Directory definition message
DMRDEF	Data maintenance request definition message
DMSTAT	Datamaintenance status report/query message
DOCADV	Documentary credit advice message
DOCAMA	Advice of an amendment of a documentary credit message
DOCAMI	Documentary credit amendment information message
DOCAMR	Request for an amendment of a documentary credit message
DOCAPP	Documentary credit application message
DOCARE	Response to an amendment of a documentary credit message
DOCINF	Documentary credit issuance information message
FINCAN	Financial cancellation message
FINSTA	Financial statement of an account message
GENRAL	General purpose message
GESMES	Generic statistical message
HANMOV	Cargo/goods handling and movement message
IFCSUM	Forwarding and consolidation summary message
IFTCCA	Forwarding and transport shipment charge calculation message
IFTDGN	Dangerous goods notification message
IFTFCC	International transport freight costs and other charges message
IFTIAG	Dangerous cargo list message
IFTMAN	Arrival notice message
IFTMBC	Booking confirmation message
IFTMBF	Firmbooking message
IFTMBP	Provisional booking message
IFTMCS	Instruction contract status message
IFTMIN	Instruction message
IFTRIN	Forwarding and transport rate information message
IFTSAI	Forwarding and transport schedule and availability information message
IFTSTA	International multimodal status report message
IFTSTQ	International multimodal status request message
IMPDEF	EDI implementation guide definition message
INFENT	Enterprise accounting information message
INSPRE	Insurance premium message
INVOIC	Invoice message
INVRPT	Inventory report message
IPPOMO	Motor insurance policy message
ITRRPT	In transit report detail message
JAPRES	Job application result message

Message ID	Message Description
JINFDE	Job information demand message
JOBAPP	Job application proposal message
JOBCON	Job order confirmation message
JOBMOD	Job order modification message
JOBOFF	Job order message
KEYMAN	Security key and certificate management message
LREACT	Life reinsurance activity message
LRECLM	Life reinsurance claims message
MEDPID	Person identification message
MEDREQ	Medical service request message
MEDRPT	Medical service report message
MEDRUC	Medical resource usage and cost message
MEQPOS	Means of transport and equipment position message
MOVINS	Stowage instruction message
MSCONS	Metered services consumption report message
ORDCHG	Purchase order change request message
ORDERS	Purchase order message
ORDRSP	Purchase order response message
OSTENQ	Order status enquiry message
OSTRPT	Order status report message
PARTIN	Party information message
PAXLST	Passenger list message
PAYDUC	Payroll deductions advice message
PAYEXT	Extended payment order message
PAYMUL	Multiple payment order message
PAYORD	Payment order message
PRICAT	Price/sales catalogue message
PRIHIS	Pricing history message
PRODAT	Product data message
PRODEX	Product exchange reconciliation message
PROINQ	Product inquiry message
PROTAP	Project tasks planning message
PRPAID	Insurance premium payment message
QALITY	Quality data message
QUOTES	Quote message
RDRMES	Raw data reporting message
REBORD	Reinsurance bordereau message
RECADV	Receiving advice message
RECALC	Reinsurance calculation message
RECECO	Credit risk cover message
RECLAM	Reinsurance claims message
REMADV	Remittance advice message
REPREM	Reinsurance premium message
REQDOC	Request for document message
REQOTE	Request for quote message
RESETT	Reinsurance settlement message
RESMSG	Reservation message
RESREQ	Reservation request - interactive message
RESRSP	Reservation response - interactive message

Message ID	Message Description
RETACC	Reinsurance technical account message
RETANN	Announcment for returns message
RETINS	Instruction for returns message
SAFHAZ	Safety and hazard data message
SANCRT	International movement of goods governmental regulatory message
SLSFCT	Sales forecast message
SLSRPT	Sales data report message
SSIMOD	Modification of identity details message
SSRECH	Worker's insurance history message
SSREGW	Notification of registration of a worker message
STATAC	Statement of account message
STLRPT	Settlement transaction reporting message
SUPCOT	Superannuation contributions advice message
SUPMAN	Superannuation maintenance message
SUPRES	Supplier response message
TANSTA	Tank status report message
VATDEC	Value added tax message
VESDEP	Vessel departure message
WASDIS	Waste disposal information message
WKGRDC	Work grant decision message
WKGREE	Work grant request message

APPENDIX C: COBOL STRUCTURE IN TRANSLITERATED HEBREW

```
01  TNUA-MEFUTZELET-LE-IDKUN.
    03 TMF-BANK-SNIF.
       05 TMF-BANK                  PIC   9(02).
       05 TMF-SNIF                  PIC   9(03).
    03 TMF-MATBEA                   PIC   9(03).
    03 TMF-KVUTZA                   PIC   9(03).
    03 TMF-CHN                      PIC   9(10).
    03 TMF-MISPAR-MEZAHE            PIC   9(09).
    03 TMF-SUG-TNUA                 PIC   9(02).
    03 TMF-TAR-PEULA                PIC   9(06).
    03 TMF-TAR-ERECH                PIC   9(06).
    03 TMF-SCHUM                    PIC  S9(13)V99
03 TMF-KOD-CHOVA-ZCHUT              PIC   9.
03 TMF-KOD-STORNO                   PIC   9(02).
03 TMF-PRTEI-CHN-NEGDI.
   05 TMF-SNIF-NEGDI                PIC   9(03).
   05 TMF-MATBEA-NEGDI              PIC   9(03).
   05 TMF-KVUTZA-NEGDI              PIC   9(03).
   05 TMF-CHN-NEGDI                 PIC   9(10).
03 TMF-SUG-ISKA                     PIC   9(03).
03 TMF-KOD-TASH                     PIC   9(03).
03 TMF-SNIF-YOZEM                   PIC   9(03).
03 TMF-MISPAR-AGID                  PIC   9(04).
03 TMF-MISPAR-ISKA                  PIC   9(05)COMP-3.
03 TMF-ASMACHTA                     PIC   9(05).
03 TMF-SIMUL-STAT                   PIC   9(03).
03 TMF-SHAAR                        PIC   9(05)V9(4) COMP-3.
03 TMF-CROSS-RATE                   PIC   9(05)V9(4) COMP-3.
03 TMF-KOD-TEUR                     PIC   9(02).
03 TMF-ACHUZ-RIBIT                  PIC   9(06).
03 TMF-TAR-PERAON                   PIC   9(06).
03 TMF-TKUFA                        PIC   9(02).
03 TMF-KOD-ZIHUI-TOFES              PIC   9(03).
03 TMF-MATBEA-AGID                  PIC   9(03)
03 TMF-SCHUM-MATAI                  PIC  S9(13)V99
03 TMF-KOD-PAKID                    PIC   9(3).
03 TMF-MIS-TEUR-TNUA                PIC   9(3).
03 TMF-SUG-HALVAA                   PIC   9(2).
03 TMF-MISPAR-LAKOACH               PIC   9(10).
03 TMF-CHN-EXIM                     PIC   9(10).
03 TMF-CHN-3                        PIC   9(10).
03 TMF-KVUTZA-5                     PIC   9(03).
03 TMF-SCHUM-MATACH-3               PIC  S9(13)V99
03 TMF-KOD-MAARECHET                PIC   9(03).
03 TMF-TAR-ERECH-ISKA               PIC   9(06).
03 TMF-TAR-PEULA-MEKORY             PIC   9(06).
03 TMF-ASM-SACHAR-CHUTZ             PIC   9(11).
03 TMF-MISPAR-HAMCHAA               PIC   9(10).
03 TMF-KOD-ISKA-ATIDIT              PIC   9(02).
03 TMF-M-RATZ-LE-ISKA               PIC   9(06).
```

APPENDIX D: ADABAS HEBREW DATA DICTIONARY STRUCTURE

This list was truncated to fit on one page

T	L	DB	Name	F	Length	English Translation
G	**1**	**AA**	**PB-PRATIM-KLALIIM**			**General Information**
	2	AB	PB-ZIHUI	N	12	ID number
	2	AC	PB-KOD-ZIHUI	N	2	Identification code
	2	AD	PB-SUG-RESHUMA	N	2	Record type
	2	AE	PB-SUG-PAIL	N	1	Active type
	2	AF	PB-SNIF-CHESHBON	N	9	Branch
	2	AG	PB-YACHAS	N	2	Relation
	2	AH	PB-KESHER-LA-CHESHBON	N	2	Relation to account
	2	A9	PB-ACHUZ-CHAVUT-LA-CHN	N	3	Percent debt to account
G	**2**	**AI**	**PB-SHEM**			**Name**
	3	AJ	PB-SHEM-PRATI	A	15	First name
	3	AK	PB-SHEM-MISHPACHA	A	19	Last name
	2	A0	PB-TAR-YESUD-TAAGID-8	N	8	Date corporate was established
	2	AL	PB-TAR-LEDA-8	N	8	Date of birth (8 digits)
	2	AM	PB-SHEM-MISHPACHA-KODEM	A	19	Previous last name
	2	AN	PB-MIN	N	1	Sex
	2	AO	PB-MATZAV-MISHP	N	1	Marital status
	2	AP	PB-ISUK	N	3	occupation, profession
	2	AQ	PB-MEKOM-AVODA	A	20	Place of work (e.g., employer)
	2	AR	PB-ANAF-MISHKI-AVODA	N	3	Market sector (of work)
	2	AS	PB-TAFKID-BACHEVRA	N	3	Position
	2	AT	PB-SUG-CHATIMA	N	3	Signature type
	2	AU	PB-TAR-PTICHA-8	N	8	Date account established (8 digits)
	2	AV	PB-TAR-PEULA-ACHRONA-8	N	8	Last transaction date
	2	AW	PB-SHAT-PEULA-ACHRONA	N	6	Last transaction time
	2	AX	PB-KOD-ISH-KASHUR	N	1	Related person code
	2	AY	PB-TAR-IMUT-MLM-8	N	8	Verification date
	2	AZ	PB-TAR-TOKEF-DARKON-8	N	8	Pasport expiration date
	2	A3	PB-SUG-HASKALA			Education type
G	**1**	**IA**	**PB-PIRTEY-BEN-ZUG**			**Spouse data**
G	2	IB	PB-SHEM-BEN-ZUG			Spouse name
	3	IC	PB-SHEM-PRATI-BEN-ZUG	A	15	Spouse first name
	3	ID	PB-SHEM-MISHP-BEN-ZUG	A	19	Spouse last name
	2	IE	PB-KOD-ZIHUI-BEN-ZUG	N	2	Spouse identification code
	2	IF	PB-M-ZIHUI-BEN-ZUG	N	12	Spouse identification
	2	IG	PB-MIN-BEN-ZUG	N	1	Spouse sex
G	**1**	**JA**	**PB-PIRTEY-YELADIM**			**Children Data**

REFERENCES

Abiteboul, S. (1997). *Querying Semi-Structured Data.* 6th International Conference on Database Theory (ICDT'97), Springer-Verlag, London, UK.

Abiteboul, S., Quass, D., et al. (1997). *The Lorel Query Language for Semi-Structured Data.* International Journal on Digital Libraries.

Abran, A., Ormandjieva, O., et al. (2004). *Information Theory-Based Functional Complexity Measures and Functional Size with Cosmic-Ffp.* 14th International Workshop on Software Measurement (IWSM) Konigs Wusterhausen, Magdeburg, Germany Springer-Verlag

Adelberg, B. (1998). *Nodose - a Tool for Semi-Automatically Extracting Structured and Semi-Structured Data from Text Documents.* ACM SIGIR August 1998 283 – 294.

Aguilera, V., Cluet, S., et al. (2001). *Querying Xml Documents in Xyleme.* Retrieved July 7, 2005, from http://www.haifa.il.ibm.com/sigir00-xml/final-papers/xyleme/XylemeQuery/XylemeQuery.html.

ALCTS. (1999). *Task Force on Metadata Summary Report.* Retrieved July 9, 2006, from http://www.libraries.psu.edu/tas/jca/ccda/tf-meta3.html.

Andrés, J., Navarro, J. R., et al. (2005). *Word Translation Disambiguation Using Multinomial Classifiers.* Pattern Recognition and Image Analysis: Second Iberian Conference (IbPRIA 2005), Estoril, Portugal, Springer-Verlag GmbH.

Arens, Y., Chee, C., et al. (1994). *Query Processing in an Information Mediator.* Proceedings of the ARPA/Rome Lab 1994 Knowledge-Based Planning and Scheduling Initiative Workshop.

Ashby, R. W. (1940). *Adaptiveness and Equilibrium.* Journal of Mental Science, 86: 478-484.

Ashby, R. W. (1947). *The Nervous System as Physical Machine: With Special Reference to the Origin of Adaptive Behavior.* Mind, 56(221): 44-59.

Ashby, R. W. (1956). *An Introduction to Cybernetics.* Chapman & Hall, London, UK.

Aslam, J. A., Emine, Y., et al. (2005). *The Maximum Entropy Method for Analyzing Retrieval Measures.* Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval. Salvador, Brazil, ACM Press.

Aumueller, D., Do, H.-H., et al. (2005). *Schema and Ontology Matching with Coma++* Proceedings of the 2005 ACM SIGMOD international conference on Management of data, Baltimore, Maryland, ACM Press.

Bachimont, B., Isaac, A., et al. (2002). *Semantic Commitment for Designing Ontologies: A Proposal.* Ontologies and the Semantic Web, 13th International Conference, Siguenza, Spain, EKAW.

Bailey, K. D. (1994). *Sociology and the New Systems Theory.* University of New York Press.

Bansiya, J., Davis, C., et al. (1999). *An Entropy-Based Complexity Measure for Object-Oriented Designs*. Theory and Practice of Object Systems, 5(2): 111-118.

Bar-Yam, Y. (1997). *Dynamics of Complex Systems*. Westview Press.

Bateman, J. A. (2002). *Finding Translation Equivalents: An Application of Grammatical Metaphor*. Retrieved 01 October, 2005, from http://acl.ldc.upenn.edu/C/C90/C90-2003.pdf

Batory, D. S. (1985). *Modeling the Storage Architectures of Commercial Database Systems*. ACM Trans. Database Syst., 10(4): 463-528.

Bayardo, R. J. J., Bohrer, W., et al. (1997). *Infosleuth: Agent-Based Semantic Integration of Information in Open and Dynamic Environments*. ACM SIGMOD Record.

Bayle, A. J. (1989). *Frames: A Heuristic Critical Review*. Computers and Communications, Scottsdale, AZ. IEEE.

Bechhofer, S., Horrocks, I., et al. (2001). *Oiled: A Reason-Able Ontology Editor for the Semantic Web* Working Notes of the 2001 International Description Logics Workshop (DL-2001), Stanford, CA, USA, CEUR-WS.org.

Bechhofer, S., van Harmelen, F., et al. (2004, 10 February 2004). *Owl Web Ontology Language Reference*. W3C Recommendations Retrieved July 7, 2005, from http://www.w3.org/TR/owl-ref/.

Behrens, C. and Kashyap, V. (2001). *The "Emergent" Semantic Web: A Consensus Approach for Deriving Semantic Knowledge on the Web*. International Semantic Web Working Symposium.

Berger, A. and Lafferty, J. (1999). *Information Retrieval as Statistical Translation*. Proceedings of the 22nd annual international ACM SIGIR conference on research and development in information retrieval. Berkeley, California, United States, ACM Press.

Berger, A. L., Della Pietra, S., et al. (1996). *A Maximum Entropy Approach to Natural Language Processing*. Computational Linguistics, 22(1): 39-71.

Berners-Lee, T., Hendler, J., et al. (2001). *The Semantic Web*. Scientific American(May 2001): 34-43.

Blasgen, M. W., Astrahan, M. M., et al. (1981). *System R: An Architectural Overview*. IBM Systems Journal, 20(1).

Bohan, N., Breidt, E., et al. (2000). *Evaluating Translation Quality as Input to Product Development*. 2nd International Conference on Language Resources and Evaluation, Athens, Greece.

Boyarsky, A. and Gora, P. (2000). *A Comparative Statistical Dynamical Analysis of Hebrew Texts*. Montreal, Canada, Department of Mathematics and Statistics, Concordia University.

Bray, T., Paoli, J., et al. (2000, 04 February 2004). *Extensible Markup Language (Xml) 1.0 (Third Edition)*. W3C Recommendation 3rd Edition. Retrieved July 7, 2005, from http://www.w3.org/TR/REC-xml.

Brickley, D. (1979). *Visicalc Information: History and Commentary from the Guys Who Created It*. Retrieved July 7, 2005.

Brickley, D. and Guha, R. V. (2000). *Resource Description Framework (Rdf) Schema Specification 1.0, 2000*. Retrieved June 05, 2002, from http://www.w3.org/TR/rdf-schema/.

Bricklin, D., Kapor, M., et al. (2003). *The Origins and Impact of Visicalc* Mountain View, CA, The Computer History Museum and Microsoft Corporation: LEcture given at the Computer History Museum.

Buckley, W. (1967). *Sociology and Modern Systems Theory*. Prentice-Hall, Inc., Englewood Clifs, NJ.

Buckley, W. (1998). *Society - a Complex Adaptive System*. Gordon and Breach Publishers.

Burrell, G. and Morgan, G. (1979). *Sociological Paradigms and Organizational Analysis*. Heinemann, London, UK.

Campbell, A. E. (2000). *Ontological Mediation: Finding Translations across Dialects by Asking Questions*. Ph.D. dissertation, S. C. Shapiro - advisor. State University Of New York at Buffalo.

Carey, M. J., Haas, L. M., et al. (1995). *Towards Heterogeneous Multimedia Information Systems: The Garlic Approach*. 5th International Workshop on Research Issues in Data Engineering - Distributed Object Management. Washington, DC, USA, IEEE Computer Society.

Casti, J. L. (1985). *Canonical Models and the Law of Requisite Variety*. Journal of optimization theory and applications, 46(4).

Cattell, R. G. G. and Barry, D. K. (1997). *The Object Database Standard : Odmg 2.0*. Morgan Kaufmann Publishers, San Francisco, Calif.

Chklovski, T., Mihalcea, R., et al. (2004). *The Senseval-3 Multilingual English-Hindi Lexical Sample Task*. Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text, Barcelona, Spain.

Churchman, C. (1971). *The Design of Inquiring Systems: Basic Concepts of Systems and Organizations*. Basic Books, Inc., New York, NY.

Churchman, C. and Ackoff, R. (1950). *Purposive Behavior and Cybernetics*. Social Forces, 29(1): 32-39.

Cluet, S., Delobel, C., et al. (2001). *Your Mediators Need Data Conversion*. The 27th VLDB Conference, Rome, Italy, ACM Press.

Cluet, S. and Siméon, J. (1999). *Using Yat to Build a Web Server*. Selected papers from the International Workshop on The World Wide Web and Databases, Springer-Verlag.

Codd, E. F. (1970). *A Relational Model of Data for Large Shared Data Banks*. Communications of the ACM, 13(6): 377-387.

Cohen, W. W. (2000). *Data Integration Using Similarity Joins and a Word-Based Information Representation Language*. ACM Trans. Inf. Syst., 18(3): 288-321.

Colton, C. and Duffee, M. (1940). *An Introduction to Abstract Algebra*. John Wiley & Sons, Inc, New York.

Corcho, Ó., Fernández-López, M., et al. (2002). *Webode: An Integrated Workbench for Ontology Representation, Reasoning, and Exchange*. Ontologies and the Semantic Web, 13th International Conference, Siguenza, Spain, Springer.

Dalvi, N. and Suciu, D. (2005). *Answering Queries from Statistics and Probabilistic Views*. Proceedings of the 31st international conference on Very large data bases. Trondheim, Norway, VLDB Endowment.

daml.org. (2004, 30 April 2004). *Daml Ontology Library*. Retrieved 01 October, 2007, from http://www.daml.org/ontologies/.

Dasgupta, A., Kumar, R., et al. (2005). *Variable Latent Semantic Indexing*. Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining. Chicago, Illinois, USA, ACM Press.

Date, C. J. (1990). *An Introduction to Database Systems*. Addison-Wesley Publishing Company.

Deerwester, S., Dumais, S. T., et al. (1990). *Indexing by Latent Semantic Analysis*. Journal of the American Society for Information Science, 41(6): 391 - 407.

DELICIOUS. (2007, 10 October 2007). *Del.Icio.Us Web Site - Sample Folksonomy*. MS IE7 Retrieved 10 October, 2007, from http://del.icio.us/url/42dd3dea5c3980e24e1cda7e93ef7533.

Delobel, C., Reynaud, C., et al. (2003). *Semantic Integration in Xyleme: A Uniform Tree-Based Approach*. Data & Knowledge Engineering, Special issue: Data integration over the Web 44(3): 267-298.

Diab, M., Resnik, P., et al. (2004). *An Unsupervised Method for Word Sense Tagging Using Parallel*. from http://www.stanford.edu/~mdiab/papers/ustACL02.ps

Doan, A., Madhavan, J., et al. (2002). *Learning to Map between Ontologies on the Semantic Web*. The 11th International WWW Conference, Hawaii.

Domingue, J. (1998). *Tadzebao and Webonto: Discussing, Browsing, and Editing Ontologies on the Web*. 11th Knowledge Acquisition for Knowledge-Based Systems Workshop, Banff, Canada.

Dospisil, J. (2003). Software Metrics, Information and Entropy. *Practicing Software Engineering in the 21st Century*, Idea Group Publishing: 116-142.

Dragoon, A. (2004). *Banks Fight Customer Flight. CIO Magazine*.

Draper, D., Halevy, A., et al. (2001). *The Nimble Xml Data Integration System (Nimble Technology, Inc.)* CoopIS 2001.

Dublin Core. (2005). *The Dublin Core Initiative*. Retrieved July 7, 2005, from http://dublincore.org/about/.

Duschka, O. M. and Genesereth, M. R. (1977). *Query Planning in Infomaster*. the Twelfth Annual ACM Symposium on Applied Computing (SAC97), San Jose, CA, ACM Publishing.

Duschka, O. M. and Genesereth, M. R. (1997). *Query Planning in Infomaster*. the Twelfth Annual ACM Symposium on Applied Computing (SAC97), San Jose, CA, ACM Publishing.

Eduard, C. D., Clement, Y., et al. (2006). *Meaningful Labeling of Integrated Query Interfaces*. Proceedings of the 32nd international conference on Very large data bases - Volume 32. Seoul, Korea, VLDB Endowment.

Emmelhainz, M. (1990). *Electronic Data Interchange a Total Management Guide* Van Nostrand Reinhold.

Farquhar, A., Fikes, R., et al. (1997). *The Ontolingua Server: A Tool for Collaborative Ontology Construction*. International Journal on Human-Computing Studies, 46(6): 707-727.

Fernandez, M., Florescu, D., et al. (1997). *Strudel: A Web Site Management System*. Proceedings of the 1997 ACM SIGMOD international conference on Management of data. Tucson, Arizona, United States, ACM Press.

Fernandez, M., Florescu, D., et al. (2000). *Declarative Specification of Web Sites with Strudel*. Very Large Data Bases (VLDB), 9(1): 38-55.

Florescu, D., Fernandez, M., et al. (1998). *Web-Site Management: The Strudel Approach*. Data Engineering Bulletin, 21(2): 14-20.

Frame, M. and Mandelbrot, B. (2003). *A Panorama of Fractals and Their Uses*. Retrieved August 11, 2003, from http://classes.yale.edu/fractals/Panorama/SocialSciences/Linguistics/Linguistics.html.

Gangemi, A., Pisanelli, D. M., et al. (2001). A Formal Ontology Framework to Represent Norm Dynamics. *Second International Workshop on Legal Ontologies*. R. Winkels, University of Amsterdam.

Garcia-Molina, H. and al., e. (1995). *The Tsimmis Approach to Mediation: Data Models and Languages* Next Generation Information Technologies and Systems (NGITS-95) Naharya, Israel.

Garcia-Varea, I. and Casacuberta, F. (2005). *Maximum Entropy Modeling: A Suitable Framework to Learn Context-Dependent Lexicon Models for Statistical Machine Translation*. Machine Learning 60(1-3): 135 - 158.

Gardent, C. and Webber, B. (2001). *Towards the Use of Automated Reasoning in Discourse Disambiguation*. Journal of Logic, Language and Information, 10(4): 487–509.

Golder, S. A. and Huberman, B. A. (2006). *Usage Patterns of Collaborative Tagging Systems* Journal of Information Science, 32(2): 198-208.

Goldfarb, C. F. (1973). *Design Considerations for Integrated Text Processing Systems*, IBM Cambridge Scientific Center Technical Report G320-2094.

Greaves, M. (2004). *2004 Daml Program Directions*. Retrieved 27 October, 2005, from http://www.daml.org/listarchive/daml-all/0301.html.

Griswold, W. G. and Notkin, D. (1995). *Architectural Tradeoffs for a Meaning-Preserving Program Restructuring Tool*. IEEE Transactions on Software Engineering, 21(4): 275-287.

Gruber, T. (1993). *A Translation Approach to Portable Ontologies*. Knowledge Acquisition, 5(2): 199-220.

Hakimpour, F. and Geppert, A. (2005). *Resolution of Semantic Heterogeneity in Database Schema Integration Using Formal Ontologies*. Inf. Tech. and Management, 6(1): 97-122.

Halevy, A. (2005). *Why Your Data Won't Mix: Semantic Heterogeneity*. ACM Queue, 3(8): 50-58.

Halpin, H., Robu, V., et al. (2007). *The Complex Dynamics of Collaborative Tagging*. Proceedings of the 16th international conference on World Wide Web. Banff, Alberta, Canada, ACM Press.

Hammer, J., Garcia-Molina, H., et al. (1997). *Extracting Semi-Structured Information from the Web*. Workshop on Management of Semi-structured Data, Tuscon, Arizona.

Hammer, J. and McLeod, D. (1993). *An Approach to Resolving Semantic Heterogeneity in a Federation of Autonomous, Heterogeneous Database Systems*. International Journal of Cooperative Information Systems (IJCIS), 2(1).

Hannon, B. and Ruth, M. (1997). *Modeling Dynamic Biological Systems*. Springer.

Harhalakis, G., Lin, P., et al. (1994). *Implementation of Rule-Based Information Systems for Integrated Manufacturing*. IEEE Transactions on Knowledge and Data Engineering 6(6): 892 - 908.

Haynes, J. D. (2001). *Churchman's Hegelian Inquiring System and Perspectival Thinking*. Information Systems Frontiers, 3(1): 29-39.

Heflin, J. (2000). *The Shoe Faq*. Retrieved July 7, 2005, from http://www.cs.umd.edu/projects/plus/SHOE/faq.html.

Hirschheim, R., Klein, H., et al. (1995). *Information Systems Development and Data Modeling – Conceptual and Philosophical Foundations*. Cambridge University Press, Cambridge, MA.

Höpken, W. (2005). *Harmonise Ontology*. E. Rohn. Innsbruck, Austria, ECCA – Etourism Competence Center Austria: email with the Harmonise Ontology attachment and meta-data attachement.

Horrocks, I., Patel-Schneider, P. F., et al. (2003). *From Shiq and Rdf to Owl: The Making of a Web Ontology Language* Elsevier's Journal of Web Semantics

Hovy, E. and Nirenburg, S. (2005). *Approximating an Interlingua in a Principled Way.* Retrieved 01 October, 2005, from http://ucrel.lancs.ac.uk/acl/H/H92/H92-1052.pdf

Hsu, C. (1991). *The Metadatabase Project at Rensselaer.* ACM SIGMOD Record 20(4): 83-90.

Hsu, C. and Rattner, L. (1993). *Metadatabase Solutions for Enterprise Information Integration Problems.* DBPL, 24(1): 23-35.

Hurford, J. (1987). *Biological Evolution of the Saussurean Sign as a Component of the Language Acquisition Device.* Lingua, 77(2): 187-222.

Husbands, P., Simon, H., et al. (2005). *Term Norm Distribution and Its Effects on Latent Semantic Indexing* Information Processing & Management 41(4): 777-787

IAIABC. (2005). *Claims Release 3 Update.* Retrieved July 05, 2005, from http://www.iaiabc.org/edi/news_docs/ebriefing/2004/2004-12-17_e-briefing.htm.

IBM (2000). *Enterprise Cobol for Z/Os Language Reference Manual # Gc27-1411-03,* IBM.

INRIA. (2002). *The Caravel Project.* Retrieved June 05, 2003, from http://www-caravel.inria.fr/.

Ives, Z. G., Halevy, A. Y., et al. (2001). *The Tukwila Data Integration System.* Retrieved July 7, 2005.

Jacob, E. K. (2004). *Classification and Categorization: A Difference That Makes a Difference.* Library Trends, 52(3): 515-540.

Jos, Kahan, et al. (2001). *Annotea: An Open Rdf Infrastructure for Shared Web Annotations.* Proceedings of the 10th international conference on World Wide Web. Hong Kong, Hong Kong, ACM Press.

Katz, B. and Lin, J. (2002). *Annotating the Semantic Web Using Natural Language.* 2nd Workshop on NPL and XML (NPLXML 2002) Taipei, Taiwan, MIT Artificial Intelligence Laboratory.

Kim, H., Park, S., et al. (2005). *A Framework for the Integration of Multimedia Data.* Journal of Object Technology, 4(July 2005): 27-35.

Kirk, T., Levy, A. Y., et al., Eds. (1995). *The Information Manifold.* Information Gathering from Heterogeneous, Distributed Environments.

Kishida, K. (2005). *Technical Issues of Cross-Language Information Retrieval: A Review.* Information Processing and Management, 41(3): 433.

Knox, R. E. (2004). *Hype Cycle for Xml Technologies for 2004,* Gartner Group: 25.

Knox, R. E. and Abrams, C. (2003). *Hype Cycle for Xml Technologies for 2003,* Gartner Group.

Knox, R. E., Abrams, C., et al. (2006). *Hype Cycle for Xml Technologies, 2006,* Gartner Group: 46.

Kolaitis, P. G. (2005). *Schema Mappings, Data Exchange, and Metadata Management.* Proceedings of the twenty-fourth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems. Baltimore, Maryland, ACM Press.

Komarova, N. and Nowak, M. A. (2001). *The Evolutionary Dynamics of the Lexical Matrix.* Bulletin of Mathematical Biology, 63: 451-484.

Kotok, A. (2000). *A Survey of Xml Business Data Exchange Vocabularies.* Retrieved July 7, 2005, from http://www.xml.com/pub/2000/02/23/ebiz/.

Lassila, O. and Swick, R. R. (1999) *Resource Description Framework (Rdf) Model and Syntax Specification.* W3C Recommendation, 1999

Lee, J. and Malone, T. (1990). *Partially Shared Views a Scheme for Communicating among Groups That Use Different Type Hierarchies* ACM Transactions on IS.

Lee, K. J. and Kim, J. H. (2005). *Sentence Compression Learned by News Headline for Displaying in Small Device.* Lecture Notes in Computer Science.

Levi, A. Y., Rajaraman, A., et al. (1996). *Querying Heterogeneous Information Sources Using Source Descriptions.* The 22nd International Conference on Very Large Databases (VLDB-96) Bombay, India.

Lim, S.-J. and Ng , Y.-K. (1999). *Webview: A Tool for Retrieving Internal Structures and Extracting Information from Html Documents* 6th International Conference on Database Systems for Advanced Applications IEEE Computer Society.

Linden, A., Buytendijk, F., et al. (2003). *New Technologies Will Change the Way We Manage Information. Gartner Research Report.*

Liu, L., Pu, C., et al. (2000). *Xwrap: An Xml-Enabled Wrapper Construction System for Web Information Sources.* IEEE Computer.

Liu, S., Mei, J., et al. (2005). *Xsdl: Making Xml Semantics Explicit.* Lecture Notes in Computer Science.

Liu, S., Mei, J., et al. (2005). *Xsdl: Making Xml Semantics Explicit.* Lecture Notes in Computer Science, 3372(2005): 64-83.

Luke, S., Spector, L., et al. (1996). *Ontology-Based Knowledge Discovery on the World-Wide Web.* Proceedings of the Workshop on Internet-based Information Systems, AAAI-96, Portland, Oragon.

Lyytinen, K. (1987). *Different Perspectives on Information Systems: Problems and Solutons.* ACM Computing Surveys, 19(1).

Macrae, D. G. (1951). *Cybernetics and Social Science.* The British Journal of Sociology, 2(2): 135-149.

Manola, F. and Miller, E. (2004). *Rdf Primer.* W3C Recommendation Retrieved July 7, 2005.

Marakas, G. M. (1999). *Decision Support Systems in the 21st Century.* Prentice-Hall, Inc., Upper Saddle River, NJ.

Maron, M. and Kuhns, J. (1960). *On Relevance, Probabilistic Indexing and Information Retrieval*. JACM (July 1960).

Mason, R. O. and Mitroff, I. I. (1973). *A Program for Research in Management Information Systems*. Management Science, 19(5): 475-485.

McCartney, G. (1988). *Implementing Computer-Based Systems*. 16th annual ACM SIGUCCS Conference on User Services Long Beach, California, United States, ACM Press.

MDLI. (2005). *Edi in Minnesota* Retrieved July 05, 2005, from http://www.doli.state.mn.us/edi_2.html.

Medicare. (2005, June 27, 2005). *Medicare Edi Statistics*. Retrieved July 5, 2005, from http://www.cms.hhs.gov/providers/edi/edistat.asp.

Metadatabase. (2003). *An Information Integration Theory and Reference Model*. Retrieved July 11, 2006, from http://viu.eng.rpi.edu/mdb/iitrm.html.

Miller, G. A. (1953). *What Is Information Measurement?* American Psychologist, 8(3-12): 3.

Miller, G. A. (1995). *Wordnet: A Lexical Database for English*. Communications of the ACM, 38(11): 39-41.

Minsky, M., Ed. (1975). *A Framework for Representing Knowledge*. The Psychology of Computer Vision. McGraw-Hill, New York.

Minsky, M. (1984). *A Framework for Representing Knowledge*. MIT AI Memo 306.

Mota, L. and Botelho, L. (2005). *Owl Ontology Translation Based on the O3f Framework*. Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems. The Netherlands, ACM Press.

Mota, L., Botelho, L., et al. (2003). *O3f: An Object Oriented Ontology Framework*. Proceedings of the second international joint conference on Autonomous agents and multiagent systems. Melbourne, Australia, ACM Press.

Motro, A. and Buneman, P. (1981). *Constructing Superviews*. SIGMOD International Conference on Management of Data,, Ann Arbor, Michigan ACM Press.

Mowshowitz , A. (1981). *On Approaches to the Study of Social Issues in Computing* Communications of the ACM, 24(3): 146-155.

MyFamily.com, I. (2006). *Discover Your Family Story*.

Nabisco Inc. and Rohn, E. (1993). *Cipps System Requirements Specifications*. East Hanover, NJ.

Nowak, M. A. and Krakauer, D. C. (1999). *The Evolution of Language*. Proceedings of the National Academy of Sciences of the United States of America, 96(14).

Noy, N. F. (2004). *Semantic Integration: A Survey of Ontology-Based Approaches*. SIGMOD Rec., 33(4): 65-70.

Noy, N. F., Sintek, M., et al. (2000). *Creating Semantic Web Contents with Protege-2000* IEEE Intelligent Systems.

OBO. (2007). *Obo Foundry Ontologies.* Retrieved 01 October 2007, from http://www.obofoundry.org/.

OWL Recommendations. (2004, 10 February 2004). *Owl Web Ontology Language Semantics and Abstract Syntax* W3C Recommendation Retrieved July 7, 2005, from http://www.w3.org/TR/owl-absyn/

Papakonstantinou, Y., Garcia-Molina, H., et al. (1995). *Object Exchange across Heterogeneous Information Sources* Int. Conf. on Data Engineering (ICDE).

Pickering, A. (2002). *Cybernetics and the Mangle: Ashby, Beer and Pask.* Social Studies of Science, 32(3): 413-437.

Pierce, J. R. (1961). *Symbols, Signals Adn Noise.* Harper & Brothers, New York.

Pinter, C. C. (1982). *A Book of Abstract Algebra.* McGraw-Hill Book Company, New York.

POC. (2007, 25 September 2007). *Plant Ontology.* Retrieved 01 October 2007, 2007.

Polderman, J. W. and Willems, J. C. (1998). *Introduction to Mathematical Systems Theory - a Behavioral Approach.* Springer.

Protege. (2007, 17 September 2007). *Ontologies Registry.* Retrieved 01 October, 2007, from http://protegewiki.stanford.edu/index.php/Protege_Ontology_Library.

Quass, D., Rajaraman, A., et al. (1995). Querying Semistructured Heterogeneous Information. *Deductive and Object-Oriented Databases*: 319-344.

Raymond, R. C. (1950). *Communications, Entropy, and Life.* American Scientist, 38(April 1950): 273-278.

Rohn, E. (1982). *Employees Driver's License and Vehicle Data Exchange for Insurance Premium Billing.* Tel-Aviv, Israeli Ministry of Defense.

Rohn, E. (1983). *Financial System - Budget Allocation Modules.* Tel-Aviv, Israeli Ministry of Defense.

Rohn, E. (1985). *Bhp Salesforce Application - Misc. Modules.* Tel-Aviv, Israel, Bank HaPoalim LTD.

Rohn, E. (2006). *Data Integration Potentiometer in Dermis.* The 3rd International ISCRAM Conference, Newark, NJ.

Rohn, E. (2007). *A Survey of Schema Standards and Portals for Emergency Management and Collaboration.* The 4th International ISCRAM Conference, Delft, Holland.

Rohn, E. and Klashner, R. (2001). *A Survey of Xml Standards,* NJIT, Newark, NJ.

Rohn, E. and Klashner, R. (2004). *Hidden Disorder in Xml Tags.* Proceedings of the Americas Conference on Information Systems, New York, NY.

Rosenblueth, A., Wiener, N., et al. (1943). *Behavior, Purpose and Teleology.* Philosophy of Science, 10(1): 18-24.

Salton, G. (1991). *The Smart Document Retrieval Project.* Proceedings of the 14th annual international ACM SIGIR conference on Research and development in information retrieval. Chicago, Illinois, United States, ACM Press.

Salton, G. and Lesk, M. E. (1965). *The Smart Automatic Document Retrieval Systems\—an Illustration.* Commun. ACM, 8(6): 391-398.

Salton, G., Wong, A., et al. (1975). *A Vector Space Model for Automatic Indexing.* Commun. ACM, 18(11): 613-620.

Schrodinger, E. (1944). *What Is Life? : With Mind and Matter and Autobiographical Sketches* Cambridge University Press, , Cambridge, UK.

Searle, J. R. (1969). *Speech Acts, an Essay in the Philosophy of Language.* Cambridge University Press, Cambridge, MA.

Shannon, C. E. (1948). *A Mathematical Theory of Communication.* Bell Systems Technical. Journal, 27(July 1948).

Shirky, C. (2004, Spring 2005). *Ontology Is Overrated: Categories, Links, and Tags.* Retrieved 10 October, 2007, from http://www.shirky.com/writings/ontology_overrated.html.

Shu, N. C., Housel, B. C., et al. (1975). *Convert: A High Level Translation Definition Language for Data Conversion.* Proceedings of the 1975 ACM SIGMOD International Conference on Management of Data, San Jose, California, ACM Press.

Shu, N. C., Housel, B. C., et al. (1977). *Express: A Data Extraction, Processing, and Restructuring System.* ACM Transactions on Database Systems 2(2).

Sinka, M. P. and Corne, D. W. (2005). *The Banksearch Web Document Dataset: Investigating Unsupervised Clustering and Category Similarity.* J. Netw. Comput. Appl., 28(2): 129-146.

Sowa, J. F. (1999). *Knowledge Representation: Logical, Philosophical, and Computational Foundations* Brooks Cole Publishing Co., Pacific Grove, CA.

Sowa, J. F. (2001). *Meaning Preservation in Translation.* Retrieved June, 2003, from http://users.bestweb.net/~sowa/logic/meaning.htm.

Stohr, E. and Nickerson, J. V. (2003). *Enterprise Integration: Methods and Direction.* Oxford University Press, Oxford.

Suciu, D. (1998). *An Overview of Semi-Structured Data.* SIGACTN, 29(4): 28–38.

Sure, Y., Erdmann, M., et al. (2002). *Ontoedit: Collaborative Ontology Development for the Semantic Web* Proceedings of the first International Semantic Web Conference 2002 (ISWC 2002), Sardinia, Italia.

Swartout, B., Patil, R., et al. (1997). *Ontosaurus: A Tool for Browsing and Editing Ontologies* Retrieved July 7, 2005, from http://ksi.cpsc.ucalgary.ca/KAW/KAW96/swartout/ontosaurus_demo.html.

SWIFT. (2005, updated May 2005). *Financial Institution Transfers Mt200 - Mt293.* Retrieved 21 Sep. 2005 from http://www.swift.com/index.cfm?item_id=42772.

SWIFT, (2005). *Standards*. Retrieved July 05, 2005, from http://www.swift.com/index.cfm?item_id=41946.

SWIFT, (2005, updated May 2005). *Swift2005 Securities Markets Mt568 - Mt599*. Retrieved 21 September, 2005, from http://www.swift.com/index.cfm?item_id=42772.

SWIFT, (2005). *Swiftnet Fin Traffic Sent Per Month* Retrieved July 05, 2005.

SWIFT, (2007, updated 25 June 2007). *Mifid Is Coming and Swift Is Ready*. Retrieved July 19, 2007, from http://www.swift.com/index.cfm?item_id=62349.

Thakkar, S., Knoblock, C. A., et al. (2003). *A View Integration Approach to Dynamic Composition of Web Services* ICAPS Workshop on Planning for Web Services, Trento, Italy.

Tomasic , A., Amouroux, R., et al. (1997). *The Distributed Information Search Component (Disco) and the World Wide Web* Proceedings of the 1997 ACM SIGMOD, ACM Publishing.

Tomasic , A., Amouroux, R., et al. (May 1997). *The Distributed Information Search Component (Disco) and the World Wide Web* Proceedings of the 1997 ACM SIGMOD, ACM Publishing.

Troncy, R., Isaac, A., et al. (2003). *Using Xslt for Interoperability: Doe and the Travelling Domain Experiment*. Proceedings of the 2nd International Workshop on Evaluation of Ontology-based Tools Sanibel Island, Florida, USA, CEUR-WS.org.

Turoff, M., Chumer, M., et al. (2004). *The Design of a Dynamic Emergency Response Management Information System (Dermis)*. The Journal of Information Technology Theory and Application (JITTA), 5(4).

Unitt, M. and Jones, I. C. (1999). *Edi - the Grand Daddy of Electronic Commerce*. BT Technology Journal, 17(3).

Uschold, M. and Gruninger, M. (2004). *Ontologies and Semantics for Seamless Connectivity*. SIGMOD Rec., 33(4): 58-64.

Verheyden, P., De Bo, J., et al. (2004). *Semantically Unlocking Database Content through Ontology-Based Mediation*. Second International Workshop on Semantic Web and Databases, SWDB, Toronto, Ontario - Canada, Springer-Verlag Berlin Heidelberg.

Vincent, V. (1991). *Semantic Heterogeneity as a Result of Domain Evolution*. SIGMOD Rec., 20(4): 16-20.

Von-Wun, S., Chen-Yu, L., et al. (2003). *Automated Semantic Annotation and Retrieval Based on Sharable Ontology and Case-Based Learning Techniques*. Proceedings of the 3rd ACM/IEEE-CS joint conference on Digital libraries. Houston, Texas, IEEE Computer Society.

von Ahn, L. (2003, 10 October 2007). *The Esp Game*. Retrieved 10 October, 2007, from http://www.espgame.org/cgi-bin/description.

von Ahn, L. and Dabbish, L. (2004). *Labeling Images with a Computer Game*. Computer Human Interaction (CHI 2004), Vienna, Austria.

W3C RDF Schema. (2000). W3C Recommendation Retrieved January 10, 2001, from http://www.w3.org/2000/01/rdf-schema#

Walmsley, J. (1992). *The Foreign Exchange and Money Markets Guide*. Wiley, New York.

WEBONT. (2001). *W3c Daml+Oil Project*. Retrieved 03 March, 2003, from http://www.w3.org/2001/sw/WebOnt/.

Wiederhold, G. (1992). *Mediators in the Architecture of Future Information Systems*. IEEE Computer, 25(3).

Williams, A. B., Padmanabhan, A., et al. (2005). *Experimentation with Local Consensus Ontologies with Implications for Automated Service Composition*. IEEE Transactions on Knowledge & Data Engineering 17(7): 1041-4347.

WORDNET. (2005). *Wordnet Website*. 2005, from http://www.cogsci.princeton.edu/cgi-bin/webwn1.7.1.

Wu, D. and Wong, D. (1998). *Machine Translation with a Stochastic Grammatical Channel*. Annual meeting of the association for computational linguistics, Association for Computational Linguistics.

XML.GOV. (2007). *Registires*. Retrieved 01 October, 2007, from http://xml.gov/registries.asp.

XML.ORG. (2007). *Schema Registry No Longer in Service*. Retrieved 01 October 2007, 2007, from http://www.xml.org/message.php.

XPath and W3C. (1999, 16 November 1999). *Xml Path Language (Xpath) Version 1.0*. W3C Recommendation Retrieved 01 October, 2005.

Xyleme. (2000). *About Us*. Retrieved July 7, 2005, from http://www.xyleme.com.

Yan, L. L. (1997). *Towards Efficient and Scalable Mediation: The Aurora Approach*. Proceedings of the 1997 conference of the Centre for Advanced Studies on Collaborative research: 23.

Youyong, Z. (2005). *Umbc Travel Ontology*. Retrieved 19 May, 2005, from http://taga.umbc.edu/ontologies/travel.owl.

Yu, C. and Popa, L. (2005). *Semantic Adaptation of Schema Mappings When Schemas Evolve*. Proceedings of the 31st international conference on Very large data bases. Trondheim, Norway, VLDB Endowment.

Zadeh, L. A. and Desoer, C. A. (1963). *Linear System Theory; the State Space Approach*. McGraw-Hill, New York,.

Zhan, C., Lu, X., et al. (2005). *A Lvq-Based Neural Network Anti-Spam Email Approach*. SIGOPS Oper. Syst. Rev., 39(1): 34-39.

Zhang, Y. T., Gong, L., et al. (2005). *Corpus-Based Word Sense Disambiguation Using Naive Bayesian.* Zhongnan Daxue Xuebao (Ziran Kexue Ban)/Journal of Central South University (Science and Technology), 36(SUPPL.): 483.

Zipf, G. K. (1949). *Human Behavior and the Principle of Least Effort: An Introduction to Human Ecology.* Addison-Wesley, Reading, MA.

Zou, Y. and Finn, T. (2003). *Taga: Trading Agent Competition in Agentcities.* IJCAI 2003 Eighteenth International Joint Conference on Artificial Intelligence, Acapulco, Mexico.