Enforcing Trade Secrets among Competitors on the Semantic Web

Choudhry Muhammad Zaki Malik

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Science and Applications

Dr. Athman Bouguettaya, Chair
Dr. Mohamed Eltoweissy
Dr. Denis Gracanin

August 18th, 2004
Falls Church, Virginia, USA

Keywords: Trade Secrets - B2B - Semantic Web - Web Service - Perturbation.

Enforcing Trade Secrets among Competitors on the Semantic Web

Choudhry Muhammad Zaki Malik

(ABSTRACT)

In this thesis, we present a novel approach for the preservation of trade secrets in a Business-to-Business (B2B) environment that involves trade among competitors. The Web provides a low cost medium for B2B collaborations. Information exchange may take place during such a collaboration. The exchanged information may be of a sensitive nature, forming a business *trade secret*. The *open* nature of the Web calls for techniques to prevent the disclosure of trade secrets. The emerging Semantic Web is expected to make the challenge more acute in terms of trade secret protection due to the automation of B2B interactions. In this thesis, the different businesses are represented by Web services on the envisioned Semantic Web. We propose a Peer-to-Peer (P2P) approach for preserving trade secrets in B2B interactions. We introduce a set of techniques based on *data perturbation* for preserving data privacy. The techniques presented in our thesis are implemented in *WebBIS*, a prototype for accessing e-business Web services. Finally, we conduct an extensive performance study (analytical and experimental) of the proposed techniques.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Business-to-Business (B2B) trade constitutes a significant portion of E-commerce [31]. Businesses have sensitive information that is essential for gaining and maintaining a competitive edge in the marketplace. As a result of B2B trade, privileged information may be exchanged between businesses. If sensitive information is disclosed to other entities, businesses may suffer insurmountable losses. This type of information is called a *trade secret*. For example, if the chemical formula for *Coke*, which the *Coca Cola company* keeps "under lock and key in a bank vault" [77] is disclosed to a competitor, it may lose its market share. The economic value of a trade secret lies in it not being readily ascertainable by other businesses [89]. The trade secret definition is closely dependent on the business nature. For example, the formula that the *Coca Cola company* uses for its flagship beverage is its trade secret. Similarly, in an online retail business like *Amazon* [7], the list of customers, sales projections, marketing plans or other forms of sensitive data may form its trade secret.

Businesses have faced the issue of maintaining ownership and preserving the control over their sensitive information including their trade secrets, since the inception of trade. Traditional non-technical solutions for preserving sensitive information include patents, copyrights, trademarks, trade regulations and contracts

between businesses. Patents are issued for a specific time limit. The information becomes available for public use after the patent *expires*. Copyrights regulate documents but not the information contained in them. Likewise, trademarks only specify what naming conventions businesses can use to name their products. Contracts are used in situations where the trading parties are known to each other. Any misuse of secret information results in "legal" actions taken against the defaulting business. Similarly, trade regulations are defined to provide legal safety for businesses. Contracts and regulations may not be effective in cases where law enforcement is difficult and the legal boundaries are not clearly defined (e.g., the Web). Therefore, there is a need of technical solutions for preserving trade secrets in inherently untrusted environments such as the Web.

1.1 Trade Secrets

Businesses need to protect information that is specific to their functioning and that is essential for gaining or keeping the competitive edge in the marketplace. Sensitive information regarding a business is termed as a trade secret of the business and can be defined as a formula, process, device, managerial list, etc., which provides a company with an advantage over its competitors that are lacking such an item. More formally, a trade secret is defined in the Uniform Trade Secret Act [89] as some information that derives some economic value for the owner. This information may include a formula, pattern, compilation, program, device, method, technique, or process, that is not easily observable using proper and legal means. It includes technical, production, marketing or sales and financial information associated with a business. A trade secret could fall in any one of these divisions of information, or it could be defined as a combination of these divisions. Technical and production trade secrets are subject to disclosure by means of theft or visiting the production sites. Securing facilities that hold such information may provide a

solution, but it has to be secured from *alien* as well as *resident* subjects (employees, partners, etc.). Similarly, financial information in form of company budgets, operation costs, profit or loss statements, etc. are at a higher risk of being disclosed from *resident* subjects. This is due to the nature of such information, as it is seldom communicated to parties outside the corporate boundary. Marketing or sales information on the other hand, is a form of trade secret that can be disclosed through electronic transactions. In E-commerce interactions, huge amounts of data are exchanged and sensitive information as the buying habits of customers, customer management techniques, customer lists, etc. can be made public without due consent. It may seem trivial, but the customer lists as a form of marketing and sales information holds great importance for several businesses. Table 1.1 shows the different forms of trade secrets that belong to different divisions of information. A concise overview of some of the different items that can constitute a trade secret is given below.

Technical Information	*Production Information*	*Sales Information*	*Financial Information*
Formula	Cost/Price data	Proprietary info.	Budgets
Prototype	Production machinery	Sale forecasts	Product Margins
Experimental data	Manufacturing technology	Customer lists	Operating costs
Design drawings	Production know-how	Buying habits	Profit/Loss statements
Research and development reports	Production process specifications	Customer confidence management	Internal financial documents

Table 1.1: Classification of Trade Secrets

Formula

A formula for a compound used in making a product falls in the category of "technical information" that a business may term as its trade secret. The formula differentiates the final manufactured product from similar products in that industry. For instance, the chemical formula used by *Coke* differentiates its product from other cola manufacturers. Similarly, the formula used by *Häagen-Dazs* makes

this particular brand of ice cream different.

Process of Manufacturing

A trade secret could constitute the different processes a business employs in manufacturing a product. Sensitive information about the processes of manufacturing may include the manufacturing technology or the machinery involved. Different firms use different processes to produce essentially the same product. For example, potato chips manufacturers use different techniques in the production of chips. It is the dissimilarity of the manufacturing process that sets *Lays* apart from *Ruffles*.

Treatment of Materials

The way in which a business treats the raw materials for production activities can also constitute a trade secret. Such information falls under the "production information" division of sensitive information. Businesses treat materials differently according to the production process specifications employed at their respective sites or according to the general know-how of the product. This knowledge is derived from past experiences and preferences, and businesses consider such information private. For example, in the leather business, the final product is differentiated on the basis that the raw material was processed. Juice producing companies like *Tropicana* or *Sunny Delight*, employ different strategies for the treatment of materials. The treatment methodologies constitute a trade secret for these companies.

Pattern of a Machine

In the world of electronics, it is the pattern of circuitry that sets one product apart from the other. This form of a trade secret spans across the "technical" as well as "production information" divisions of information. The pattern of machine can include prototype information, design drawings or research reports for making the product in the desired manner. Also, the manufacturing technology used in choosing a specific pattern for a product is of private nature. For instance, the way in

which a circuit is laid out in an *Intel* microprocessor is different from an *AMD* microprocessor.

List of Customers

In certain businesses, a list of customers is of primary significance. Consider the case of producing a commodity where several manufacturers produce for a list of customers. It is this list of customers that is vital for ensuring the business between the producer/supplier and the buyer. The contracts between such businesses are carefully negotiated and the announcement of a customer list would mean that other suppliers could target the same customer. The list of customers is also particularly important for retail businesses that employ different techniques to attract customers. In an E-commerce scenario, this list gives a sense of *loyalty* that customers develop over time with that business. This loyalty is established due to a number of factors that include better product offerings, timely delivery, after-sales service, discounted prices, etc.

It should be noted that all information that a business deems as being secret might not actually be a trade secret. Medicinal drug patents are one example. The patent owner is protected by the law that no other competitor will make the drug till the patent expires, and there is no compulsion to make the ingredients public. However, competitors can and in most of the cases discover the "secret ingredients" that make up the drug. Therefore, this information although being secret and sensitive cannot be termed as a trade secret. In order for some information to be considered as a trade secret for a business, that information should be of some value to both, the *owner* business and its competitors. Moreover, such information should be protected from being known to people involved in the business, either directly (employees) or indirectly (partners, traders, etc.), and suitable measures should be taken to "guard the secrecy of the information" [90].

If the information is already known outside the business then it is not a "secret" in true sense of the word. Similarly, if the information is accessible to a large

number of people, then there is a potential for the information to flow to unwanted hands. This is one of the highly reported cases of *information theft* and some protection regarding such *loss* is provided by the law. However, it should be noted that legislations work only if a case about the "replication" of trade secrets is established. Slight modifications to the original information can get away from prosecution.

1.2 Collaboration among Competitors

Competitors need to collaborate and complement each other to create market value [15]. For example, airline companies often form alliances to cut their expenses and better serve their customers, cellular phone providers collaborate with their competitors to provide extended coverage plans. Similarly, online retailers may collaborate with their competitors to provide better service to their customers. In this case, the customer information may be exchanged between the collaborating businesses. This information collected over a period of time forms a customer list, which usually forms a business trade secret in online retailing [77]. Because of the inherent type of the business, the trade secret of a business may be disclosed as a result of information exchange [16]. Assume that *Aracron* and *Zak & Mak* (*Z&M*) are online book retailers. *Aracron* may experience high customer demand of an item which may create a shortage of stock. The nature of the demand is such that it needs to be fulfilled in a short duration of time e.g., an event or a season. It may be the case that *Aracron's* normal procurement methods may not replenish the stock to honor customer demand in the required time. In such a situation, it may need to go to an alternative source to satisfy that demand to ensure customer satisfaction and loyalty. Satisfied customers tend to buy other items in the same transaction [38, 43]. A competitor, say *Z&M* could provide the requested item to *Aracron* in the required time (geographical proximity can be one of the reasons).

In business situations where the number of satisfied customers dictate the profits (e.g., online retailing), customer information, factors effecting their interests, market data, etc., typically constitute a business trade secret. In online retailing, the customer list of a business usually forms the trade secret [77]. The trade secret of a business may be disclosed as a result of the information exchange. For example, *Z&M* may acquire information about *Aracron's* customers as a result of their interaction and use it to its advantage. The *fear* of losing sensitive information to a competitor may obstruct a business (*Aracron*) intent to collaborate. In this thesis, we focus on the customer personal and purchase information, as a business primary trade secret. The aim is to protect the customer related information so that the requesting business does not lose its *share* of the market.

Businesses may need to collaborate with previously *unknown* businesses for providing better service to their customers. In the pre-Web era, B2B interactions were conducted on a smaller scale due to the high infrastructure cost and scarcity of players. The global reach of the Web introduces a large number of businesses that can provide the desired goods and services in an *effective* manner. The widespread use of the Web may act as *Achilles' heel* in terms of trade secret protection. Information is easily accessible while mechanisms to protect it are not available. This is due mainly to the *open* nature of the Web where information is accessible to a large number of *a priori* unknown participants. It is evident that in such a scenario it becomes difficult for a business to have contractual agreements with all potential B2B participants. First, the enormous number of possibilities that exist make the task of contract management an intractable choice. Second, businesses may provide a product or service at one point in time and may retract from offering it later on. Such a transient nature of online B2B relationships obstructs the notion of contracts. Third, enforcement of laws on the Web is elusive. The interacting businesses may be located at different countries (or continents) which are governed by different laws. A business may be held accountable for its actions in one country while in the other it may avoid any prosecution. While this work focuses on

B2B collaboration between competitors, it also applies to collaboration involving outsourcing relationships. For example, in Government-to-Business (G2B) transactions, citizens are sometimes served through third party contractors. In course of such service, citizen information is submitted to these contractors. Citizen privacy is at risk as he/she may not want any agency/business to get hold of the private information. Our proposed solution provides privacy protection in this case as citizen data cannot be collected and mined without the citizen's consent.

1.3 Trade Secret Protection on the Semantic Web

The above mentioned problems of preserving trade secrets are expected to exacerbate on the emerging Semantic Web. The Semantic Web is hailed as the next revolution on the Web. It is expected that "inter-business" communication would be facilitated by the Semantic Web. The vision of the Semantic Web is to make the available information understandable by machines [12]. Information would be exchanged between *automated* processes to carry out the required B2B tasks. In this case they would also be entrusted with the critical task of managing trade secrets. The problem of trade secret protection on the Web is expected to intensify because of the reliance on *automated* information handling.

Web services are expected to be the key enablers of the Semantic Web. The W3C defines a Web service as "a software system designed to support interoperable machine-to-machine interaction over a network" [95]. Web services acting as proxies for businesses involved in a B2B activity would provide an efficient and low-cost interaction mechanism. In traditional commerce or electronic interactions with "human-in-the-loop", a collective decision about the flow of information in B2B interactions can be taken at anytime by the human participant. If the participant *fears* information disclosure, the interaction can be stopped in favor of a better alternate. In contrast, human participants would assume minimal or

no control on the Semantic Web. For example, an online retail business such as *Aracron* may need to collaborate with previously *unknown* businesses located in different geographical regions of the world. To provide expedited service for *time sensitive* requests of customers located in the U.K., it may need to invoke the Web services provided by *Imperial books* (a U.K. based bookstore) or any other competitor in that region. Similarly, a customer located in Ireland may be better serviced through collaboration with an Irish business. These collaborations are expected to be formed transiently on the Semantic Web. It is expected that the *best* alternatives will be explored in an *automatic* manner by Web services without much human intervention.

The *automated* handling of information and difficult enforcement of legislation across borders when trading on the Semantic Web requires technical solutions to ensure that trade secrets be preserved. A B2B activity can be divided into three distinct stages, namely: discovery, negotiation and fulfillment. Automated processes are responsible for handling information in the three stages on the Semantic Web. The discovery stage is where businesses place advertisements of their offerings and other businesses may consume those following some standardized protocols. The negotiation stage is one where the businesses interact with their potential collaborators to see if they can agree on mutually acceptable terms of business. This is done through exchange of proposals describing constraints on acceptable terms. These terms could include a definition of the good or service being traded, price, delivery date, etc. The agreed transaction is carried out in the fulfillment stage. In *standard* business practices, trade secrets are preserved by constant monitoring at each stage. Businesses are held accountable for any behavior that lies outside the terms of trade. The Semantic Web does not offer the same protection, as *automated* processes (mainly Web services) are responsible for majority of the work. Any secret information that is divulged to one service may flow to *unwanted* destinations. Thus, Web services would need to be equipped with mechanisms to prevent the disclosure of trade secrets. We propose a framework for B2B inter-

actions on the Semantic Web where business competitors can trade knowing that they are in full control of their customer data, i.e., trade secret.

1.3.1 Data Perturbation

Our proposed solution is based on *data perturbation* to protect customer data misuse. We assume that B2B Web service collaboration between competitors is mainly driven by *time sensitive* demand. Major events, holiday seasons (e.g., Christmas or Thanksgiving), or special days (Valentine's day) require the online retail businesses to deliver the requested items in minimal time to maintain the returning customers and acquire new ones. It means that all items be shipped directly to the customer from the competitor's site to shorten delivery time. The competitor may use data mining techniques to extract sensitive information (e.g., name, address, buying habits, etc.,) from the data collected as a result of the interaction. If the data is perturbed in a way that hinders such mining, the trade secret of a business may not be disclosed.

Data perturbation approaches work by transforming the "original data" (e.g., adding random *noise* to the data) such that the individual data values are not identifiable [88]. Studies show that humans can tolerate mistakes to a certain level in words [74]. This implies that inconsistencies in spellings seldom inhibit the understanding of a word. For example, the word *VIRGINIA* is understood even if it is misspelled as *VIRG1NlA*. This result forms the basis of our proposed approach. In normal business practice, the items are delivered through a postal agency (e.g., US. Mail, Fedex, UPS, Airborne). We propose to perturb the delivery information in a manner that deters mining and knowledge extraction but does not hinder the delivery process. After the initial post office pre-sorting based on Zip codes (using bar codes, etc.,) most of the handling of the mail is done by human workers for delivery. It is assumed that the mail personnel or post office workers will be able to tolerate some perturbation in the data. For instance, a delivery item is likely to

reach its destination even if it is misspelled as in the example above.

The perturbations are done in a manner that mail delivery personnel are able to *understand* the address information, i.e., the intentional changes do not hinder its intended purpose [73], the delivery of mail items. However, to a data mining program, the perturbed words would appear distinct information, thus, deterring information extraction. Therefore, any mining software would find it challenging to associate the two orders coming from the same street, state, etc. The subtle lexical difference in the word makes it useless for mining purposes.

1.4 Need for Competitor Collaboration: Motivation

To understand the need for competitor collaboration driven by high demand coupled with the potential of trade secrets disclosure, consider the case of end of year holidays. The sales of consumer goods are usually the strongest during this period. They may account for 20% or more of the annual sales [58]. The three month period around Christmas can reach 40% of all the sales [37, 71]. For instance, 48% of toys, games and hobby goods are sold in this period. Also, jewellery and watch purchases account for 71% of total sales in the Christmas season [58]. Retailers usually plan carefully for this season relying on a wide range of marketing and data analysis tools. Customer data and loyalty are key to a successful marketing campaign. In this respect, forecasting the items and types of merchandise that are expected to be in great demand are crucial to securing a competitive edge. The outcome depends largely on mining customer data and ensuring that customers are loyal and will come back to that provider. However, ensuring customer loyalty comes at a price. Retailers must ensure that customers are satisfied with the service. This is critical for *time sensitive* purchases such as those occurring in end of the year holidays. In this case, retailers (such as *Aracron*) may turn to competitors

(such as *Z&M*) to acquire out of stock items that usually would not be available within the required period of time from the usual suppliers.

There are two types of situations that could endanger businesses in their B2B transactions trying to put customer service above any other consideration. The first case occurs when a trade secret involving single items could be compromised. For instance, consider the case of a color version of a classic movie like *Gone with the Wind* being just released. This item may become very popular as a gift during the end of year holiday. As a result, retailers may mine purchase data of this item to better predict similar single-item purchases for the next end of year holiday. For instance, retailers would keep up-to-date on related releases (classic movies on DVD, old car models on new chassis, etc.,) to be able to respond to market demand on time.

To further elucidate this problem, consider a case of audio compact disc (CD) sales. Assume that due to the popularity of a music album, its demand increases at an unexpected rate. In such a case businesses may need to outsource from competitors to keep their *loyal* customers. It may be the case that certain genre of CDs are popular among different people e.g., rock and rap may be popular among college students while opera or classical may be a choice of people belonging to another group. It would be beneficial for any business to employ direct targeting techniques and not send discounts or other incentives to buy classical music to college students. It is the norm that college students live in close proximity of the school. At times, many students may live on the same street (dormitories, fraternities, or due to other cheap housing). If a business can establish such an association with street addresses, it could reduce its marketing costs. Moreover, if the customer list is made due to orders coming from a competitor (our case under study), the business could offer incentives to those group of individuals and *gain* their loyalty.

Assume that *Z&M* has collected some information about various *Aracron* customers over a period of time (Note that here we assume *Z&M* and *Aracron* are

Artist	*Date*	*Genre*	*Name*	*Address*	*City*	*State*	*Zip Code*
Metallica	02.10.04	Rock	Izzy George	1266 Lawrence Street	Waco	TX	67208
MegaDeth	02.13.04	Rock	Zashi Zbarov	987 Lawrence Street	Waco	TX	67208
Pantera	05.25.04	Rock	Rad Meters	1345 Lawrence Street	Waco	TX	67208
Anthrax	07.21.04	Rock	Mark Waugh	885 Lawrence Street	Waco	TX	67208
Mozart	07.21.04	Classical	Lars Ulrich	525 Lawrence Street	Waco	TX	67208
Slayer	07.25.04	Rock	Glen Gary	612 Lawrence Street	Waco	TX	67208

Table 1.2: Similar Information Clustered by Z&M's Web Service

online retailers selling a number of things besides books) . As a result of cluster analysis, it may form the list of records shown in Table 1.2. It is evident from the list that orders from "Lawrence Street" are mainly for *rock music* CDs. An association rule that may be derived as a result of this data cluster may be of the following form:
if Street-Name = 'Lawrence Street' *then* Popular-Genre = 'Rock'

Z&M could utilize this information to perform direct marketing of *rock* CDs at Lawrence Street. In contrast, if the information cannot be clustered on basis of similarity, *Z&M* would have to market the CDs for a much larger geographical region. For instance, as mentioned earlier in this thesis report, we do not perturb the Zip code data in our proposed approach. In this case, the marketing would need to be done for the entire '67208' Zip code. Table 1.3 shows the perturbed records which do not reveal the crucial information about Lawrence Street. Since, the street names are perturbed, a data mining program would treat each instance of the street name to be distinct, thereby inhibiting data clusters or associations. In the various *security* mechanisms, the information that is *encrypted* is mostly not human readable. However, in our perturbation scheme we have to keep the information readable to humans for delivery purposes.

Artist	*Date*	*Genre*	*Name*	*Address*	*City*	*State*	*Zip Code*
Metallica	02.10.04	Rock	Izzy George	1266 Lawrence Street	Waco	TX	67208
MegaDeth	02.13.04	Rock	Zashi Zbarov	987 Lrwȧence Street	Waco	TX	67208
Pantera	05.25.04	Rock	Rad Meters	1345 Läwr̲enc̃e S̆retet	Waco	TX	67208
Anthrax	07.21.03	Rock	Mark Waugh	885 L̄awr̂ence Str̄eet	Waco	TX	67208
Mozart	07.21.04	Classical	Lars Ulrich	525 Lance Street	Waco	TX	67208
Slayer	07.25.04	Rock	Glen Gary	612 Laurele Street	Waco	TX	67208

Table 1.3: Clustering Inhibited by Perturbation

In Table 1.3, three *originally* distinct street names are shown, while records two, three and four are shown as perturbed versions of the 'Lawrence Street'. The human readers (postal workers) can distinguish that 'Lawr̂ence Str̄eet' is the same as 'Lawrence Street'. However, an automated computer program may not conclude the same. Moreover, even if it is established that the data is perturbed, the computer program may not associate 'Lawr̂ence Str̄eet' with 'Lawrence Street', as the perturbed version could also be for 'Lance Street' or 'Laurele Street'. For instance, 'L̄aurẽlle Street' has the same number of characters (some perturbed) as 'Lawrence Street'. If the collected data consists of such thousands of *similar perturbations*, with multiple field perturbations, the data mining program may not be able extract useful associations or clusters.

The second case happens when a trade secret that involves groups of related items, i.e., categories, could potentially be divulged. Consider the case of video games that usually become popular in the end of year season. Such purchases may include either new video games, game players or both. Typically, the video game industry releases these items close to the holiday season to maximize their advertising campaign and profits. This usually means that retailers may suffer a shortage in supply within a short period of time. Examples include *Sony's PlayStation 2*, that became unavailable within a few days on the market. In this case, retailers may need to resort to competitors to satisfy their customers' requests.

In the process, competitors may use the requested information to their advantage for other high season purchases of similar types of items.

The categories of different items that may be grouped together mostly relate to single individuals (although not always). For instance, consider the case of a customer named 'Jaymz Het' who is loyal to *Aracron* and always buys from them. It may be the case that over a period of time majority of Jaymz's requests are routed to *Z&M* for fulfillment. Even if we assume that only a few requests are forwarded to *Z&M*, there is still a possibility of *unauthorized* information flow from *Aracron* to *Z&M*. For example, *Z&M* may use the customer data it gathers as a result of interaction with its competitor *Aracron*, and establishes relationships, associations, etc. with data collected about the customer from other sources. In this case, the customer's privacy becomes a prime concern. Table 1.4 shows the data collected by *Z&M* over a period of two years. Clearly, 'Jaymz' has an interest in *Nascar* racing and related materials. In the future, *Z&M* can employ marketing techniques that involve different *Nascar* products for Jaymz.

Item	*Date*	*Name*	*Address*	*City*	*State*	*Zip Code*
Nascar Magazine	08.11.03	Jaymz Het	20 Billz Place	Turin	WA	50205
Nascar T-Shirt	12.10.03	Jaymz Het	20 Billz Place	Turin	WA	50205
Nascar Clock	05.22.04	Jaymz Het	20 Billz Place	Turin	WA	50205
Nascar Towel	09.13.04	Jaymz Het	20 Billz Place	Turin	WA	50205

Table 1.4: Customer Information Collected Over a Period of Time

Jaymz's information, if perturbed by the requesting business, i.e., *Aracron* in this case, may look like as shown in Table 1.5. If *Z&M* collects this information, then even a period of time it would not be able to extract the needed associations as each record would appear to be of a distinct individual. This would protect Jaymz's privacy as *Z&M* was not authorized in the first place to get the personal information. Moreover, *Aracron*'s trade secret would be preserved, as Jaymz may not be targeted for campaigns involving *Nascar* materials, due to preservation of

sensitive information.

Item	*Date*	*Name*	*Address*	*City*	*State*	*Zip Code*
Nascar Magazine	08.11.03	Jaỹmz Het	20 Bi$\tilde{\tilde{l}}$lz Place	Turin	WA	50205
Nascar T-Shirt	12.10.03	Jymaz Hēt	20 Bil̤lz Plcae	Turin	WA	50205
Nascar Clock	05.22.04	Jâỹmz Ȟet	20 Blilž Plācç	Turin	WA	50205
Nascar Towel	09.13.04	J̲aymž Hėt́	20 B̂l$\bar{l}$iz Pačle	Turin	WA	50205

Table 1.5: Perturbed Customer Information

The above two cases illustrate the problem of conflicting interests faced by online retailers. On the one hand, retailers would like to keep their customers coming back, i.e., achieving customer loyalty. On the other hand, they need to make sure that their competitive edge remains intact when interacting with competitors. Therefore, there is a need of devising a strategy that aims at finding the most *profitable balance* between *customer satisfaction* and *non-violation of trade secrets*. This work aims at providing a conducive environment for B2B interactions in untrusted environments. The premise is that B2B collaborations would flourish if the infrastructure provides solutions to preserving trade secrets.

1.5 Case Study: A Book Ordering System

Assume that *Aracron* and *Z&M* are direct competitors in the book retail market. In our framework, both businesses are represented by Web services on the Semantic Web. We also assume that *Aracron* holds a "contingency relationship" with *Z&M* for selling books. The nature of the relationship is such that the *Aracron* Web service forwards a customer's request to a *Z&M* Web service if it does not hold the requested item in stock. We assume that when all the business partners of *Aracron* respond negatively to an item query, the *Aracron* Web service may forward the request to its competitor Web service of *Z&M* as opposed to turning the customer

away. Upon receiving the request, *Z&M* Web service honors it by sending the required material directly to the customer through a delivery agency. The whole process is shown in Figure 1.1.

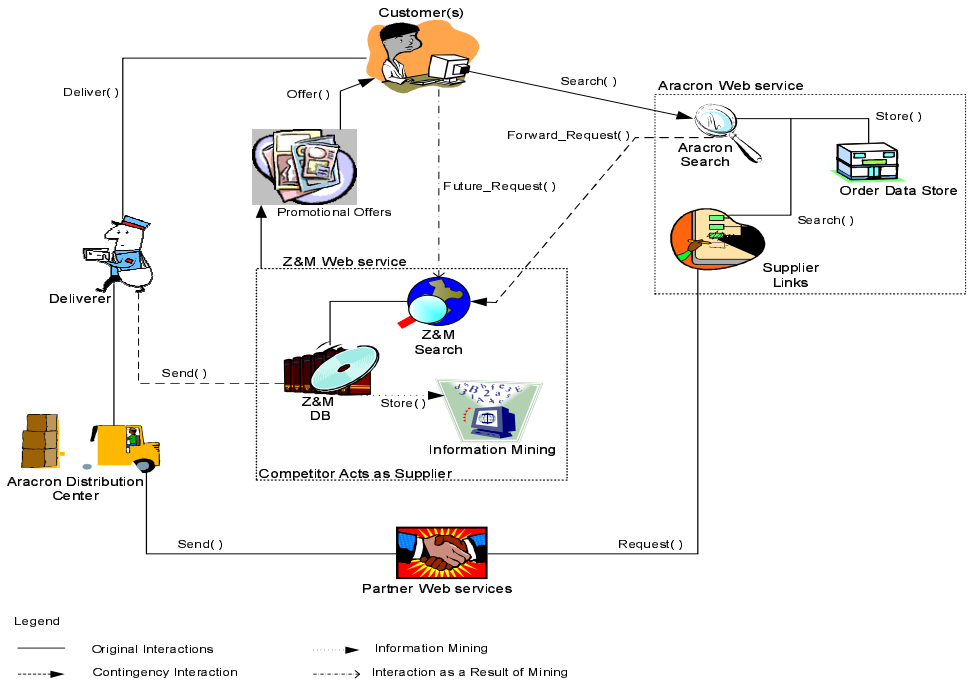

Figure 1.1: Interaction Between Two Business Web Services

The motive behind forwarding this request to the competitor is to ensure *customer satisfaction* and *loyalty*. Businesses gain customer loyalty due to a number of factors that may include superior service, flexible payments and member discounts among other things. Since *Aracron* and *Z&M* are competitors, they would want to protect their market share. For example, consider the case of an *advertising strategy* of a business. This strategy may provide the business with plans about the target market for the advertising campaign, its duration, the expected turnover and the types of products to advertise. It may depend on a number of

input data items that constitute the trade secret. Possible input items include market analysis data as sale durations, competitor estimates, partner production information, price information and customer details. Knowledge of only some data items may not prove to be adequate in divulging the business trade secret. The correct *mix* and details of all items is required to reveal the trade secret.

A competitor may want to establish a relationship between the popularity of a book, its price and the intended customer audience. The price information and information about the popularity of a book may be available publicly, e.g., book "sales ranks" at *Amazon* [7]. Table 1.6 gives a sample of data collected by *Z&M*. In case of *time sensitive* demand, it is important that items be delivered directly to the customer within the requested time framework.

A *Z&M* Web service may keep a track of the order information and may develop a customer list information over a period of time. It may then use inferential techniques to predict the demand on the required product. The Web service can utilize the acquired information to ameliorate its market strategy. For example, it may infer that people who live in VA buy books on the genre of *ethics*. It may then start to attract VA resident customers by offering incentives on books with ethics related titles. Buyers are usually attracted to the *best* possible offer. Customer *loyalty* becomes a low priority in these situations. The buyer may potentially move its business to *Z&M*. As a result, *Aracron* may lose its competitive edge. In this scenario, the customer information forms the trade secret of the business. In this work, we focus on the customer list as a business trade secret. It is our aim to develop techniques that facilitate data sharing for the purpose of B2B collaboration while preserving the *secrecy* of sensitive attributes. We provide a framework that enables competitive businesses to collaborate with each other without the fear of losing their trade secrets.

Book Id	*Req. Qty.*	*Rank*	*Price ($)*	*Date*	*Genre*	*Name*	*Address*	*City*	*State*	*Zip Code*
469	1	15,000	29	10.31.03	History	Sam Monts	75 Hill Avenue-3	Trent	CA	33456
570	3	954	59.95	01.03.03	Biology	Mary Floren	423 Xavor Road	Dallas	TX	87231
520	1	2,356	34.45	12.25.03	Ethics	Billy Bowden	69 Stuart Place	Vienna	VA	22309
525	8	1,152	17.35	07.21.03	Ethics	Zachary Mills	5 Feroze Road	Austin	VA	54600

Table 1.6: Information Stored by Z&M's Web Service

1.6 Thesis Statement

Web service security and privacy have recently taken a central stage as emerging research areas. Several techniques have been proposed in this regard. Standardization efforts are also under way for supporting Web service security and privacy. However, these techniques and standards provide little or no support for the preservation of trade secrets. The various efforts for privacy preservation do not encompass Business-to-Business (B2B) interactions. Additionally, the legislative protection is not clearly enforceable in a Semantic Web environment. In competitive situations, the flow of trade secrets may prove harmful for a business to keep its revenues. Competitive B2B collaborations mainly involve sharing of data. This shared data may form the trade secret of a business (which may include the customer list).

A probable approach to dealing with the issue of e-business trade secret preservation is to share data in a manner that does not divulge sensitive information. This could be done by examining each data item using human intervention. It is a tedious process and great amounts of efforts are required to achieve high levels of privacy. We propose a framework for the automatic *enforcement of trade secrets in B2B interactions among competitors*. We employ the notion of data perturbation to achieve business data privacy in a seamless manner. Businesses can outsource from competitors without the fear of losing their trade secrets and in turn their

market share.

1.7 Organization

The remainder of this thesis report is organized as follows.

In Chapter 2, we present an in-depth study of business interaction technologies in the pre Web services and Web services eras. We illustrate major interaction technologies including data interchanges, components and workflows. The major B2B industry initiatives that have adopted XML as the base technology for data interchange are also discussed. The role of Web services in enabling B2B interactions on the Semantic Web are also detailed. At the end of the chapter, we discuss some major complexities that arise due to the B2B information exchanges.

In Chapter 3, we show the various business interaction models adopted by different businesses. The choice of the model depends on the business nature. Therefore, we present a brief overview of each model in context of the Semantic Web, i.e., how each model would fare on the Semantic Web. We propose a P2P approach for managing interactions in this highly dynamic environment. The flow of data in a P2P model is also detailed.

In Chapter 4, we propose a *perturbation model* for business Web services on the Semantic Web. We present three major methods for P2P interaction data where the correctness of data dictates its use.

In Chapter 5, we describe the experiments done so far on the basis of our proposed approach. We describe an implementation of the approach in the *WebBIS* prototype and present an extensive study using an analytical approach.

In Chapter 6, we describe the major techniques, standards, and platforms for data privacy in a B2B scenario, that are most closely related to our research on *trade secrets*.

In Chapter 7, we provide the concluding remarks and discuss directions for

future research.

Chapter 2

Using Web Services for B2B Interactions

In this chapter, we present a discussion of the different approaches that facilitate Business-to-Business (B2B) interactions. We begin by providing a detailed account of the technologies used in the *pre-Web* era. Then we list a few limitations of these techniques. Then we discuss how Extensible Markup Language (XML) has changed B2B interactions. The essential concepts behind Web services and how they tackle the B2B issues are detailed next. Lastly, we overview some of the open issues in B2B interactions complexity.

2.1 B2B Interactions

An important challenge in B2B E-commerce is *interaction*. Interaction is defined as consisting of *inter-operation* and *integration* with both internal and external enterprise applications [32]. B2B applications are composed of autonomous, heterogeneous, and distributed components. Thus, interaction has been a major concern over the years. As an active research topic, it has encompassed areas such as

databases, knowledge-based systems, and digital libraries [14]. Interactions in B2B E-commerce offers unique challenges because of issues such as scalability, volatility (dynamism), autonomy, heterogeneity, and legacy systems. B2B E-commerce requires the integration and inter-operation of both applications and data. Disparate data representations between partner's systems must be dealt with. Interaction is also required at a higher level for connecting front-end with back-end systems, legacy data sources, applications, processes, and workflows to the Web, and trading partners' systems which may include competitors. The limitations of traditional Enterprise Application Integration (EAI) systems led to the current efforts around Web services as well as the shift to a service-oriented paradigm in application development.

To better understand the limitations of current systems, consider a procurement scenario, where a company (acting as a customer) needs to order goods from another company (a competitive supplier). The supplier then processes the order and delivers the goods, either directly (if it has goods in stock) or by requesting that the goods be shipped by a third party. Once the order is processed, the payment is made. For all the parties involved (customers, vendors, and the competitors), it would be very beneficial if the whole procurement process was automated, all the way from requesting quotes to processing payments. Today, in all but very few cases, even if business processes within a company are automated, business processes across companies are carried out manually. The "integration" is performed manually by means of employees who access the internal systems (for example to retrieve the list of products to be ordered) and then communicate with other companies by filling out Web forms (e.g., to order the goods) or via email or fax. Lower costs, streamlined and more efficient processes, ability to monitor and track process executions, and ability to detect and manage exceptions drive the need for automation. However, none of the conventional technologies have been able to fully address the challenges posed by the above scenario.

2.2 Pre-Web Services B2B Interactions

The need for B2B interactions has spurred technological growth in E-commerce. These technologies have been around for more than three decades providing businesses, such as the banking industry, with a secure framework for sharing and exchanging data electronically. The most widely used and earliest framework is the *Electronic Data Interchange* (EDI) standard that runs on dedicated computer networks. The advances in technology have given rise to a new breed of affordable software for distributed messaging and computing that can securely run on public computer networks: *component-based frameworks*. With corporate takeovers and consolidations coupled with just-in-time inter-enterprise cooperation on the Web, the need arose for enabling *inter-enterprise workflows*. *Virtual Enterprises* [10, 35, 36], that are seen as the future in B2B E-commerce will heavily draw on these solutions.

There are several successful examples of business integration. Broker companies such as Ariba or CommerceOne represent some of them. The purpose of these brokers is to facilitate integration by performing functions analogous to those of centralized enterprise middleware, from supporting binding to routing messages among the services provided by the different companies. However, the lack of support by major software vendors for the formats and protocols defined by these brokers and the trust-related problems that undermine any centralized solution have resulted in limited acceptance for these solutions.

Other successful examples of B2B integration are systems based on *Electronic Data Interchange For Administration, Commerce and Transport* EDIFACT [51]. For instance, the US retailer *Wal*Mart* was able to automate some of its cross-enterprise processes, in particular with respect to co-managing the inventory with its suppliers, setting stock levels, and automatically ordering supplies when in-stock levels were low. In spite of the success stories, standards like EDIFACT and systems based on such standards have never become widely adopted for a

variety of reasons. First, designing such systems is typically an ad hoc endeavor and the result of a one time programming effort. The lack of standards and the lack of an appropriate infrastructure (from middleware to networks) made each one of these systems unique in that each one of them had to implement everything almost from scratch. In addition, the underlying hardware and communication support was very heavy-handed. In terms of networks, before the Web appeared, communication often used to take place through leased lines to obtain the necessary bandwidth and security guarantees. In terms of computer cycles, most of these systems were very heavy. As a result, such systems were expensive to develop, almost impossible to reproduce, difficult to maintain, and could not be adapted to new technologies. Moreover, because of the development effort and costs involved, only large companies could afford deploying such systems.

The Internet alleviated some of these design problems by allowing designers to replace leased lines with a network that was pervasive and more cost-efficient. Nevertheless, the lack of standardization at the system and communication protocol levels still remained a significant hurdle in the path toward reducing the cost and complexity of building and deploying such systems. This problem was recognized years ago and there were many standardization attempts but, for reasons not always entirely rational, they had only limited success. At the core of these efforts were technologies that would allow homogeneous middleware platforms to communicate with each other (such as Inter-ORB communication via GIOP/IIOP). These technologies can be easily extended to act as the middleware for the Web. However, as it often happens, these early approaches were never widely used and, in time, were obscured by new developments.

The Web constituted an important step toward facilitating application integration. In fact, it probably was the crucial step toward systems that were more than isolated, ad hoc efforts. The Web brought standard interaction protocols (HTTP) and data formats (XML) that were quickly adopted by many companies, thereby creating a base for establishing a common middleware infrastructure that reduces

the heterogeneity among interfaces and systems. However, HTTP and XML by themselves are not enough to support application integration. They do not define interface definition languages, name and directory services, transaction protocols, and the many other abstractions are crucial to facilitate integration. It is the gap between what the Web provides (HTTP, XML) and what application integration requires that Web services are trying to fill. In the "pre-Web services era" three major technology initiatives have tried to solve the integration problem. These are electronic data interchange-based solutions, component technology-based solutions and workflow-based solutions. In the following we overview each of these technologies in detail.

2.2.1 Electronic Data Interchange (EDI)

[67]

The inter-organizational application-to-application transfer of business documents between computers in a compact form is defined as EDI [67, 100]. Its primary aim is to minimize the cost, effort, and time incurred by the paper-based transfer of business documents [1]. EDI requires that trading partners exchanging a business document agree on the format of the document.

2.2.1.1 B2B Interactions in EDI-based Solutions

EDI standards provide a single homogeneous solution for content interoperability. In EDI, Value Added Networks (VANs) are used to handle message delivery and routing among business partners. These define a set of types for describing business documents. However, there is a limited (albeit large) number of predetermined documents supported by EDI standards. Companies are limited to a set of EDI documents for which standards already exist [1]. It would be difficult for trading partners to conduct transactions whose parameters are not included in an EDI document. In that regard, EDI is hardly flexible in its ability to expand the set of

supported document types. The introduction of a new type or changing an existing type of business transaction may be complex and time consuming [1]. This kind of changes requires modification to the configuration of the translation software and must be validated in the related standard or EDI guideline committee which usually takes a long time [1]. For example the *EDI Guideline Consistency Subcommittee* (EGCS) is responsible for the content and maintenance of all *Telecommunications Industry Forum* (TCIF) EDI-maintained code lists [8]. Any modification to these code lists has to be reviewed by the EGCS. The EGCS is also responsible for notifying the *TCIF Secretariat* of any changes in the electronic documentation.

Security and heterogeneity hold strong significance in an EDI-based B2B E-commerce solution. The document exchange takes place over private or value-added networks. Business partners do not concern themselves with those security issues encountered in public networks. Moreover, business partners do not need to directly reference each other's systems. Therefore, critical security issues are not present. All partners are required to comply with the EDI standard. As a result, heterogeneity is not a problem. However, understanding all information in an EDI document is not a simple task. For example, there are data elements in EDI document whose sole purpose is to indicate the start and end of a message. The EDI approach rates low in autonomy, because partners are not free to introduce local data formats, or use other standards. All changes must be approved by the standard committees [1].

The major drawback in developing and maintaining an EDI-based solution is the cost of establishing a new relationship. This usually requires a significant overhead. However, several EDI implementations have shown impressive results as set in the example of *Scientific and Engineering Workstation Procurement* (SEWP) [66]. Because EDI is based on proprietary and expensive networks, organizations, predominantly small and medium, could not afford EDI. They were, *de facto*, excluded from being partners with larger organizations that mandate the use of EDI [1, 51]. Typically, VAN services entail three types of costs: account

start-up costs, usage of variable costs, and VAN-to-VAN interconnect costs for the number of characters in each document [51]. The final cost of an EDI solution depends on several factors such as the expected volume of documents, economics of the EDI translation software, and implementation time. Maintenance fees and VAN charges can vary considerably and as such affect the cost of EDI systems. Some VAN providers do their billing on a per document basis. Others charge based on the number of characters in each documents [51]. It has been reported that 90% of the Fortune 500 companies in the United States use EDI; only 6% of the other 10 million companies can make that claim [1]. Efforts to reduce the cost of using VAN networks include Internet-based EDI solutions such as EDIINT [49] and *Open Buying on the Internet* (OBI) [70].

The standard documents and actual formats that would be exchanged, are negotiated and a set of implementation conventions are agreed upon for each EDI deployment. This negotiation and agreement process represents a significant cost in EDI deployment. To address this issue, EDIFACT and *American National Standards Institute* (ANSI) X.12 have undertaken an effort to standardize sets of documents for various industries. For example, ANSI X.12 has recently released a set of standard EDI document definitions for the health care industry. Using these industry standard document definitions, the customizations required per relationship can be reduced, although per-relationship work is generally still required. Additionally, once implementation conventions are decided upon, custom integration work must be performed at both partner organizations for the existing enterprise systems to process the EDI documents. Purchasing a commercial EDI system, integrating it with the enterprise systems, and writing custom code to translate the EDI system document definitions to the corresponding enterprise system records is typically involved.

2.2.1.2 Internet-based EDI Initiatives

Extensions have been made to the traditional EDI to facilitate business integration. For instance, business documents in EDI standards have been mapped to XML documents (e.g., XML/EDI [101]). More specifically, the combination of EDI and Internet technologies seems to overcome several shortcomings of the traditional EDI (e.g., VAN charges). Indeed, several organizations are already using EDI for transacting over the Internet. For example, EDI purchase orders and invoices are now routinely exchanged via the Internet by NASA, Sun Microsystems, and Cisco systems. Major Internet-based EDI initiatives include EDIINT (*EDI over the Internet*) [49] and OBI [70].

EDIINT [49] – Internet is used as a communication medium in EDIINT instead of VANs. Other than that, it is similar to the traditional EDI. The aim is mainly to reduce EDI communication charges due to the use of VANs. EDIINT was initiated by the *Uniform Code Council* (UCC) to standardize the method to exchange EDI documents over the Internet. One of the first EDIINT standards to emerge (in 2000) was EDIINT AS1 (*Applicability Statement 1*). EDIINT AS1 set the rules to exchange EDI documents using SMTP protocol. The second standard (completed in 2001) was EDIINT AS2 standard. The communication of EDI documents using the HTTP protocol is supported.

Security concerns over the Internet were an impediment to use the medium for exchanging critical business information. thus, a security mechanism (EDIINT AS2) using *Pretty Good Privacy* (PGP) encryption and digital signatures [48] was proposed. The standards referenced by EDIINT AS2 include RFC1847 and MIME Security with PGP [48].

OBI [70] – OBI aims to complement EDI standards, not replace them. It leverages EDI to define an Internet-based procurement framework. OBI targets only

business transactions that involve non-strategic materials. More precisely, OBI is intended for high-volume, low-dollar amount transactions, which account for 80% of the purchasing activities in most organizations. These are transactions for *Maintenance, Repair, and Operations* (MRO), office supplies, laboratory supplies, and other indirect materials.

Message exchange takes place through the HTTP protocol. OBI relies on the ANSI X12 EDI standard to describe the content of order documents. Order documents are encapsulated in OBI objects. OBI objects also encapsulate other non-EDI messages such as buyers' and sellers' digital signatures. OBI does not introduce a specific model for describing locally maintained information (e.g., product and price information). This information may be described in the partner's database. At the business process level, OBI defines a simple and pre-defined operational protocol for Internet-based purchasing. This protocol consists of a number of commonly agreed upon activities (e.g., select a supplier, create order) for purchasing non-strategic material (e.g., office supplies, laboratory supplies). In fact, this protocol only specifies the way partner OBI systems interact. It is the responsibility of each partner to integrate its internal applications (catalogs, inventory and order management systems, etc) with OBI servers.

As with EDI, OBI provides a robust security infrastructure. It uses the SSL (*Secure Sockets Layer*) [68] over HTTP for securing communications. It also uses digital signatures and digital certificates for ensuring messages authenticity and integrity. OBI rates higher than EDI with regard to the scalability. OBI targets simple and pre-defined purchasing transactions. Also, it offers lower entry cost as it is an Internet-based framework.

2.2.2 Components

Components are program modules that can be independently developed and delivered [13, 86]. They may be newly developed or wrap existing functionalities

provided by databases, legacy systems or packages. Although most of the fundamental ideas that define object technology are applicable to components, components are not necessarily created using object-oriented tools and languages [61, 44]. For example, components may be realized using a functional language, an assembly language, or any other programming language [86].

The availability of a *middleware* that provides more effective ways of programming is important to the development of distributed component-based applications. The development of component-based applications generally requires the interconnection of geographically distributed components. A *component middleware* is an infrastructure that supports the creation, deployment, and interactions among components [91]. Three major component middleware frameworks that have been developed during the past decade are CORBA, DCOM and EJB. A brief overview of each is listed below:

CORBA (*Common Object Request Broker Architecture*) [72]: CORBA provides mechanisms to support platform heterogeneity, transparent location and implementation of objects, interoperability and communication between software components of a distributed object environment [72]. It is the standard promoted by the *Object Management Group* (OMG), an international industry consortium. The backbone of CORBA is the *Object Request Broker* (ORB) which allows communication between client and server components.

DCOM (*Distributed Component Object Model*) [62]: DCOM is Microsoft's technology for distributed components. DCOM allows components to communicate across system boundaries. For components to interact, they must adhere to a specific binary structure. The binary structure provides the basis for interoperability between components written in different languages[62].

EJB (*Enterprise Java Beans*) [76]: In EJB, business logic may be encapsulated as a component called an *enterprise bean*. It provides a separation between the business logic and the system-level details. This separation extends Java's "Write Once, Run Anywhere" portability to allow Java server components to run on any

EJB-compliant application server [76]. The *container* is the core of EJB component model. It provides a runtime environment that hosts and controls the beans.

The component-based approach for B2B E-commerce is suitable in situations where the number of partners within an enterprise is small [23].

2.2.2.1 CORBA-based B2B E-commerce

The use of ORBs in CORBA hides the underlying complexity of network communications from application developers. When a client issues a method invocation on a server component, the ORB intercepts the invocation and routes it across the network to the appropriate server. It is also possible that components distributed on different ORBs communicate over the Internet through the *Internet Inter-ORB Protocol* (IIOP). Recent efforts have been made to add semantic features to CORBA through the *Electronic Commerce Domain Task Force* (ECDTF) reference model which includes a *semantic data* facility [72]. However, the model is still at its very early stage. Additionally, very little work has been done so far to define a specification for the semantic data facility.

Tight coupling and long term business relationships between components is a central feature of CORBA. Once interfaces are expressed in IDL (*Interface Definition Language*), they are compiled by an IDL compiler into *stubs* and *skeletons*. The *stub*, used on the client side, invokes remote operations via the ORB to the corresponding skeleton on the server side. The *skeleton* gets the call parameters, invokes the actual operation implementation, collects results, and returns values back to the client through the ORB. It allows both static and dynamic invocations to use all modes. The use of message driven interactions among components allows the support of loosely coupled relationships. Implementation details are normally hidden from application developers. Interfaces are the only considerations businesses must make when interacting with each other. Business partners have the latitude to implement their interfaces in ways that best fit their internal needs and

requirements.

In terms of heterogeneity, CORBA was designed to be independent of implementation languages, operating systems, and other factors that normally affect interactions. Components can be implemented using diverse programming language such Java, C++, and Smalltalk. Businesses are tightly bound to interfaces published by their trading partners. Hence, any change to a partner's interface may need the corresponding interface to be re-compiled.

The complexity of CORBA development increases the cost of entry in CORBA-based solutions for B2B E-commerce. For example, developers in CORBA must generate binary code packages and deploy them on client sides when building new applications or when modifying the interfaces of existing applications. Although the dynamic invocation interface in CORBA alleviates this problem, programming calls with such interface is fairly complicated

2.2.2.2 DCOM-based B2B E-commerce

DCOM-based solutions for B2B E-commerce allow a DCOM client to access an operation of another component using virtual lookup tables to obtain a pointer to that operation. The DCOM runtime environment ensures that the pointer is local to the invoking process by using proxies [57].

These components are also tightly coupled enabling long term business relationships. Proxies need to be created at the client side to communicate with stubs on the serving end [57]. The operation invocation process is static in DCOM which prevents establishing dynamic relationships among components. In terms of heterogeneity, current DCOM implementations are mostly based on Windows platforms although some experimentation have been done to port DCOM to other platforms (e.g., UNIX). Also, the languages that are mostly used to write DCOM components are Microsoft J++ (Microsoft's implementation of Java), C, C++, and Visual Basic. Additionally, DCOM's IDL is neither CORBA nor DCE (*Dis-*

tributed Computing Environment) compliant [57]. Security in DCOM relies on the Windows NT security model. Although this allows developers to build secure applications on Windows platforms, it is not clear how security will be provided when DCOM is used on other platforms.

2.2.2.3 EJB-based B2B E-commerce

EJB uses the *Java RMI* [84] to enable interactions among beans. The use of RMI makes the location of the server transparent to the client. Similar to CORBA and DCOM, EJB is fairly limited in terms of interactions and caters for tightly coupled and long term business relationships. Developers must define an RMI remote interface for each bean. The RMI compiler generates a stub for each remote interface. The stub is installed on the client system and provides a local proxy for the client. The stub implements all the remote interfaces and transparently delegates all method calls across the network to the remote bean. *Java Messaging Service* (JMS) that adds support for message driven beans has been recently used, extending the EJB component model to support both tightly and loosely coupled applications [23].

The EJB container provides security features to EJB components. Each deployment descriptor contains declarations about the access control for the corresponding enterprise bean. When a client calls an operation of that bean, the container is responsible for checking that the requester has the right to invoke that operation by accessing an access control list.

2.2.3 Business Process Administration Using Workflows

Workflow management is concerned with the declarative definition, enactment, administration and monitoring of *business processes*. A *business process* (or workflow process) consists of a collection of activities related by data and control flow relationships. An activity is typically performed by executing a program, enact-

ing a human/machine action, or invoking another process (called sub-process). Programs, persons, machines, and data used to perform workflow processes are called *workflow resources*. The scripting of activities and resource policies through *business process analysis, modeling, and definition tools* defines a *business process definition* (workflow schema) [26]. The *workflow enactment service* enables different parts of the business process to be enacted by providing interfaces to users, applications, and databases distributed across the workflow domain.

Workflow technology is one of the most important candidates for integrating, automating and monitoring processes [20]. Traditional workflow systems are based on the premise that the success of an enterprise requires the management of business processes in their entirety. Indeed, an increasing number of organizations have already automated their internal process management using workflows and enjoyed substantial benefits in doing so. Current business processes within an organization are integrated and managed either using *Enterprise Resource Planning* (ERP) systems (e.g., *SAP/R3*, *Baan*, *PeopleSoft*) or various workflow systems such as IBM's *MQSeries* or integrated manually in on-demand basis. However, B2B E-commerce requires the flexible support of cross-enterprises relationships. Traditional workflow systems are ineffective when we consider the needs of B2B E-commerce, with its complex partnerships, possibly among a large number of highly evolving processes.

Each commercial *Workflow Management System* (WfMS) implements its own specification language, with little attention paid to offering uniformity among products. To address this issue, the *Workflow Management Coalition* (WfMC) has defined the *Workflow Reference Model* [42]. The model includes a standardized set of interfaces and data interchange formats between workflow systems' components. The *WfMC*'s model puts more emphasis more on the syntactic integration of workflow processes. It provides little support for inter-enterprise business processes.

The overall workflow specification is partitioned into several sub-workflows, each encompassing all the activities that are to be executed by a given entity within an organization in *Distributed Workflow Systems* (DWSs) [65]. DWSs im-

pose that each organization participating in a distributed workflow deploy a full-fledged execution engine, capable of interpreting the workflow definition. The same workflow model must be adopted by each participant in the global workflow. This approach assumes that global and sub-business processes use the same process definition and data exchange model. This is a quite restrictive assumption in the context of B2B E-commerce where participants may use disparate data and process representation models and private business processes may require access to proprietary/legacy data sources and applications. In addition, DWSs assume a tight coupling model among the distributed sub-workflows. Thus, modifications to back-end applications, sub-workflows, and global workflow need to be coordinated. The cost of establishing a new relationship may be significant as business processes must be modeled and deployed in concert across all participants. DWSs are appropriate for the development of a business process of a single organization that needs to integrate multiple distributed sub-workflows.

Inter-enterprise business processes management features the separation between *public* and *private processes* [17, 26]. A *public process* defines an external message exchange of an organization with its partners according to a message exchange protocol such as EDI and RosettaNet. A *private process* describes internal executable activities that support the activities of public processes. Public and private processes interact through *process wrappers*. Process wrappers consist of pre-defined activities that can be used in a private business process to send/receive messages to/from public business processes. For example, if a public process uses *XML Common Business Library* (xCBL) [29] to represent business documents, and the private business process expects documents in *cXML* [24], the conversion between these two formats is handled by a wrapper. Private processes may also interact with back-end applications through *application adapters*. In this approach there is no requirement that local process management engines (e.g., engines which are responsible for managing private business processes) be identical. It is possible for example, that one engine is based on IBM's *MQSeries* [47] and another one is

based on HP's *Process Manager* [45].

2.3 Limitations of Conventional Systems in B2B Interactions

There are several reasons why conventional middleware platforms cannot be used in B2B interactions. The first one is that in cross-organizational interactions there is no obvious place where to put the middleware. The basic idea for conventional middleware was for it to reside between the applications to be integrated and to mediate their interactions. While the applications were distributed, the middleware was centralized (at least logically), and a single company controlled it. Adopting the same solution in this context would require that the customer, vendor, and competitor agree on using and cooperatively managing a certain middleware platform and on implementing a *global workflow* that drives the whole business process.

While this approach is feasible in some restricted settings (e.g., a very small number of companies that have frequent, close cooperation), in the general case it turns out to be an unlikely proposition. In fact, the lack of trust between companies, the autonomy that each company wants to preserve, and the confidentiality of the business transactions play against the idea of having a centralized middleware hosted by one of the participating companies or by a third party. Each company wants to control its own business operations and how they are carried out, and does not want its business transaction data to be seen by anybody other than its intended recipient.

An alternative solution for a company would be to address the problem in a point-to-point fashion, by separately tackling the integration problem with each of the participants. This means that whenever two parties (the customer and the supplier) want to communicate, they agree on using certain middleware protocols

and infrastructure. For example, they can both deploy a message broker and use it to send messages to each other, as long as this message broker provides the necessary support for wide area integration. With this approach, there is no third party involved and confidentiality is preserved, as only the intended recipient can see the business transactions.

However, since a company typically interacts with many different participants and each supplier could require the use of a different middleware platform, this leads to a scenario where a company has to support many heterogeneous middleware systems. The result is that each company must integrate these different middleware systems (not to mention purchasing and maintaining them), which were instead intended to facilitate the integration.

Another reason that makes conventional middleware unsuitable is that many assumptions that were valid in EAI do not hold here. One such difference is that EAI interactions are typically short lived, while cross-organizational interactions last longer, and sometimes much longer. Rather than calling a procedure, a method, or a function, interactions involve coarse-grained operations lasting possibly for hours or days. As an example of the delays involved, the supplier may confirm that the order has been processed only after a shipping company has physically picked up the requested goods. Such delays explain why cross-organizational interactions are mostly implemented as asynchronous exchanges. However, asynchronous interactions introduce their own problems. For example, consider the problem of providing transactional properties to the interaction between two or more parties. If the operations are long lasting, then conventional protocols such as *Two Phase Commit* (2PC) are not applicable, as they would lock resources for long period of time and therefore severely limit the possibility of executing concurrent operations. Yet, these are the protocols supported by conventional middleware and EAI tools.

2.4 XML Based B2B Interactions

The exponential growth of the Web opened opportunities for businesses to transact across all types of boundaries (geographical, national, business category, etc). It is noteworthy that the traditional approaches for B2B interactions were not devised for the Web. Therefore, early research had focused on providing a *lingua franca* for B2B E-commerce that went beyond HTML to reflect the richness of the data being advertised/published. Such an effort resulted in the development of XML (*eXtensible Markup Language*) [93]. However, XML was not developed to define semantics, description of message exchange sequences, or definition of correct interpretations of exchanged messages [17]. To address this issue, standardization committees defined XML-based B2B interaction frameworks (or standards) A parallel effort is the work on the *Semantic Web* [12].

There are a large number of XML-based frameworks for B2B interactions. An exhaustive list of XML-based B2B standardization efforts can be found in [69]. These frameworks sometimes overlap or even compete with each other [17]. The issue of interoperability has thus shifted from the level of applications to the level of standards. A trading partner has to deal with several standards at the same time. In case one trading partner exchanges messages across industries, the variety of standards is likely to increase even more [17]. One solution to deal with such problem has been described in [17] through the use of *B2B protocol and integration engines*. These execute actual message exchanges according to various standards. In what follows, we overview two of the most popular XML-based frameworks for B2B interactions.

2.4.1 RosettaNet

RosettaNet [79] aims at standardizing product descriptions and business processes in information technology supply chain applications. RosettaNet's supply chain

include information technology products (e.g., boards, systems, peripherals, finished systems) and electronic components (e.g., chips, connectors). RosettaNet focuses on three key areas of standardization to automate B2B interactions. First, the vocabulary needs to be aligned. The *RosettaNet Business Dictionary* contains vocabulary that can be used to describe business properties (e.g., business name, address, tax identifier). The *RosettaNet Technical Dictionary* contains properties that can be used to describe characteristics of products (e.g., computer parts) and services (e.g., purchase order). Second, the way in which business messages are wrapped and transported must be specified. The *RosettaNet Implementation Framework* specifies content of messages, transport protocols (HTTP, CGI, email, SSL) for communication and common security mechanism (digital certificates, digital signatures). Third, the business process governing the interchange of the business messages themselves must be harmonized and specified. *RosettaNet*'s *Partner Interface Processes* (PIPs) are pre-defined XML-based *conversations*.

The use of a vertical ontology (i.e, common vocabulary with information technology supply chain domain) contributes to solving the problem of semantic heterogeneity. RosettaNet focuses on providing a common basis for B2B public interactions via PIPs. The integration of PIPs with internal business processes is performed by partners. It also addresses some of the semantic of business layer interoperability by proposing common business interactions (PIPs). RosettaNet does not provide means to define arbitrary global business processes.

2.4.2 ebXML

Electronic Business XML (*ebXML*) [28] aims at defining a set of specifications for enabling B2B interactions among companies of any size. The basic part of the ebXML infrastructure is the *repository*. It stores important information about businesses along with the products and services they offer. At the communication layer, businesses exchange messages through the *messaging service*. One important

feature of the ebXML messaging service is that it does not rely on a specific transport protocol. It allows for the use of any common protocol such as SMTP, HTTP, and FTP.

Interaction between companies is carried out through interaction of *business documents*. A *business document* is a set of information components that are interchanged as part of a business process. Business documents are composed of three types of components: *core components*, *domain components*, and *business information objects*. *Core components*, stored in the *core library*, are information components that are re-usable across industries. *Domain components* and *business information objects* are larger components stored in the *domain library* and *business library* respectively. Core components are provided by the ebXML library while domain component and business information objects are provided by specific industries or businesses.

ebXML defines a *business process specification schema* available in UML and XML versions. The UML version only defines a UML class diagram. It is not intended for the direct creation of a business process specification but provides a representation of all the elements and relationships required for its creation. The XML version allows the creation of XML documents representing ebXML-compliant *business process specifications*. ebXML provides a set of common business process specifications that are shared by multiple industries. These specifications, stored in the *business library*, can be used by companies to build customized business processes. Interactions between business processes are represented through *choreographies*. A *choreography* specifies the ordering and transitions between business transactions. To model collaboration in which companies can engage, ebXML defines *collaboration protocol agreements* (CPAs). A CPA is an agreement by two trading partners which specifies in advance the conditions under which the trading partners will collaborate (e.g., terms of shipment and terms of payment). Note that the trade secret prevention clauses are not stated.

The ebXML infrastructure enables secure and reliable communications by using

emerging security standards (e.g., SSL and S-HTTP). In addition, digital signatures can be applied to individual messages or a group of related messages to guarantee authenticity. The initial goal of the ebXML initiative was to support a fully distributed set of repositories which is an interesting feature for improving scalability. However, to date, only a single repository is specified.

2.5 Web Services Based B2B Interactions

2.5.1 Web Services

The definition of *Web services* varies to an extent from generic to very specific and restrictive. Nonetheless, the underlying concepts and technologies are to a large extent independent of how they may be interpreted. Often, Web services are seen as applications accessible to other applications over the Web [33]. This *generic* definition classifies 'anything' with a URL to a Web service. It may include CGI scripts, or other programs accessible over the Web with a stable API, published with additional descriptive information on some service directory.

In 2001, the (then) UDDI consortium defined web services as "self-contained, modular business applications that have open, Internet-oriented, standards-based interfaces" [97]. In this definition, the emphasis is placed on the need to comply with Internet standards. Moreover, it requires the service to be open, i.e., it should have a published interface that can be invoked across the Internet. Still, the meanings of self-contained and modular are not clear from this definition.

The World Wide Web consortium (W3C) developed the above definition and refined it to be more specific: "a software application identified by a URI, whose interfaces and bindings are capable of being defined, described, and discovered, as XML artifacts. A Web service supports direct interactions with other software agents using XML-based messages exchanged via Internet-based protocols" [95]. The W3C definition is quite accurate and also hints at how Web services should

work. The definition stresses that Web services should be capable of being "defined, described, and discovered," thereby clarifying the meaning of accessible and making more concrete notion of "Internet-oriented, standards-based interfaces". It also states that Web services should be "services" similar to those in conventional middleware. Web services should be described and advertised so that it is possible to write clients that may bind and interact with them. Simply, Web services are applications that can be integrated into more complex distributed systems. This interpretation is very much in line with the approach we take in our thesis.

The W3C also states that XML is part of the solution. Indeed, XML is so popular and widely used today that, just like HTTP and Web servers, it can be considered as being part of Web technology. There is little doubt that XML will be the data format used for many Web-based interactions [5]. *Simple Object Access Protocol* (SOAP) [96], *Web Services Definition Language* (WSDL) [98] and *Universal Description, Discovery and Integration* (UDDI) [97] are the leading standards that make applications "accessible" to other applications. These are XML-based standards used for invoking, defining and discovering Web services. Web services work on the assumption that the functionality made available by a company will be exposed as a service. In middleware terms, a service is a procedure, method, or object with a stable, published interface that can be invoked by clients.

In terms of how they are used, Web services are no different from middleware services, with the exception that it should be possible to invoke them across the Web and across companies. As a consequence, Web services assume that services are loosely coupled, since in general they are defined, developed, and managed by different companies. As Web services become more popular and widely adopted, they are likely to lead to a scenario where service-oriented architectures, advocated for many years, finally become a reality. In fact, with Web services, designers and developers are led to think in the direction that "everything is a service," and that different services are autonomous and independent (as opposed to being, e.g., two CORBA objects developed by the same team). This interpretation has important

implications in that it leads to decoupling applications and to making them more modular. Therefore, individual components can be reused and aggregated more easily and in different ways.

Note that not every service available through the Web is a Web service. This is a common mistake that leads to quite a lot of confusion when discussing Web services technology. There is a difference between services in the software sense and services in the general sense, i.e., activities performed by a person or a company on behalf of another person or company. Take as examples bookstores, restaurants, or travel agencies. They all provide services. In some cases, a customer might even be able to obtain such services through the Web server of the company. Strange as it might seem at first, this is not what Web services are about. A Web service is a software application with a published a stable programming interface, not a set of Web pages.

The final key ingredient of the Web services recipe is standardization. In conventional application integration, the presence of standards helped to address many problems. CORBA and Java, for example, have enabled the development of portable applications, have fostered the production of low cost middleware tools, and have considerably reduced the learning curves due to the widespread adoption of common models and abstractions. Whenever standardization has failed or proved to be inapplicable due to the presence of legacy systems, the complexity and cost of the middleware has remained quite high and the effectiveness rather low. For Web services, where the interactions occur across companies and on a global scale, standardization is not only beneficial, but a necessity. Having a service-oriented architecture and redefining the middleware protocols is not sufficient to address the application integration problem in a general way, unless these languages and protocols become standardized and widely adopted.

Major software vendors have recently shown an unprecedented commitment to standardization, recognized this problem. These efforts in Web services have been initially driven by a small, focused group of companies, and have then been adopted

by different organizations such as OASIS (Organization for the Advancement of Structured Standards) or the W3C. These consortia attempt to standardize all the different aspects of the interaction, ranging from interface definition languages to message formats and interaction protocols.

The Web has itself been characterized by a high degree of standardization, which has allowed it to function and prosper without centralized coordination (with the exception of the Domain Name System or DNS) and has enabled its expansion at an unthinkable rate. Web technologies are now widely accepted and are very successful in enabling the interaction between humans and applications (through Web browsers and Web servers). It is therefore natural for this novel application integration technology to use the Web as its basic foundation and to try to proceed along the same, successful path taken by the Web in terms of standardization.

Observe that the commitment to standardization does not necessarily mean that there will be only one specification for each aspect of the interaction. Sometimes competing and conflicting specifications appear, possibly developed at the same time by different groups or as a result of slightly different needs. This is natural in the early days of a new technology, and does not severely limit its adoption as long as the number of such competing specifications remains relatively small, especially if they eventually converge into one commonly adopted specification. Indeed, the need for a unique solution is already bringing some order in the initially fragmented Web services landscape, and it is likely that in the end a limited number of specifications will emerge as winners.

2.6 B2B Interactions Complexity

To put the problem into perspective and appreciate why even using Web services, B2B interactions can be quite challenging, let us consider a simple case. Let us examine specifically the problem of developing Web services that enable two

business partners to interoperate. As a test case, the example is very humble and far from the most extreme ideas pursued around Web services. Yet, the example illustrates very well why Web services in their current state do not, by themselves, solve all the problems that need to be tackled.

Suppose that two business partners decide to engage in a business transaction with each other according to a well known, thoroughly designed, and well documented set of specifications. Assume as well that all legal issues (contracts, trade secrets, responsibilities, etc.) and security requirements have been discussed and agreed upon beforehand. Even under these assumptions, the work required to enable this interaction will be far from easy.

More than any other problem, no exchange between the partners will be possible until they agree on the semantics of the documents involved. In fact, no matter how precise the specifications are, there is always room for ambiguity and misunderstandings, and Web services technology does not currently offer any solution to this most important problem. As an example, take the simple case of a price field within an XML document, describing the price of a product requested by a customer. First of all, the content of the field may be expressed in different currency units. Does the specification assume a given currency, or does it allow users to qualify the price by explicitly stating the currency unit? And how should the currency unit be specified? In addition, different countries have different conventions on the decimal symbol: some use the dot, others use the comma. Depending on the domain, there may be a number of additional semantic issues that, if left undefined, may lead different partners to make different assumptions, thereby generating errors and inconsistencies. For example, the price may be interpreted as being inclusive or exclusive of sales tax; the same applies to value-added taxes. Different countries may have different taxation rules, thereby requiring further details in the specifications.

The number and types of details that must be specified is, in principle, boundless, and we could go on for several pages mentioning more sources for misinter-

pretations related to the price element. As the number of elements in an XML document grows, the challenges in the specifications and the risks of misinterpretations grow as well. This is a problem common to most vertical standards, which are often inherently complex. Indeed, these observations reinforce the well-known principle that users can easily and rapidly adopt only simple languages and protocols.

Besides imprecise specifications, another problem is that all practical situations have their own peculiarities, and may therefore require the exchange of information for which there is no room in the XML schema defined by the standardization body. Therefore, despite the adoption of a specific standard, meetings and discussions among the interested parties are required before two companies can interoperate, to agree not only on the exact meaning of each data item, but also on how to exchange additional information for which there is no room in the standard specification.

Given all the challenges that must be faced in such a simple scenario, it is easy to imagine that B2B integration becomes very hard to achieve if the problem is more complex than the one mentioned above. For example, consider the case in which the interaction is again based on vertical standards, but where business partners are dynamically selected and invoked, without prior meetings or agreements between companies. Web services technology facilitates this kind of interaction since, in addition to standard interface and conversation languages as well as basic interaction protocols, it also provides Web directories that can be used for the dynamic discovery of service providers that are able to provide a given service. However, there is no way of automatically addressing the problems of the different semantic interpretations of the XML document elements and of exchanging additional information not contemplated by the standard.

Even if we assume that these issues do not exist or have somehow been solved (as it could indeed be for very simple cases or for mature standards), doing business with previously unknown partners is something that many companies, both clients and service providers, tend to avoid for many reasons: the quality of the provider's

service may be unknown, the provider may not be trusted, or there may be no legal agreements or trust relationships to help manage disputes.

Indeed, even in the "traditional" Internet and E-commerce, when we purchase goods online, we tend to go to Web sites that we know or that are at least referred to by some portal we trust, perhaps *Yahoo* or *Amazon*, rather than buy from some random vendor we never heard of before. Furthermore, even when there is no such middleman to give us advice, we can use our own judgment, maybe based on how professional the Web site looks, or by reading the terms and conditions written on the Web site. In any case, we would very likely only be willing to pay very small amounts when buying from an unknown seller. In Web services the situation is much more complex because human beings are outside the picture. The interaction is between applications. Of course, one can think of hybrid approaches, where Web services technology facilitates the interaction, but where human beings are still involved in the service selection phase. This is the main reason why it is widely accepted at this stage that UDDI registries will be used mostly by human beings to find information rather than by programs to perform dynamic binding to Web services. But this would mean giving up many of the benefits promised by Web services, i.e., those of going from a "do-it-yourself' to a "do-it for-me" Internet.

Chapter 3

Business Interaction Models

In order to protect the trade secret of a business, we describe three main business interaction models. Businesses use these models to protect their trade secrets. Note that the efficiency of each model is dependant on the interaction environment. In the Semantic Web environment, the *Peer-to-Peer* (P2P) interaction model may prove to be effective in terms of delivery time and scalability. Thus, we focus on the P2P interaction model.

3.1 Business Centric B2B Interactions

The business centric interaction model is the traditional way of conducting a B2B activity. In this model, the buying business sends out an item request to the seller and receives the required merchandise/services. It is then the responsibility of the buying business to send the materials to the customer. Let us consider our running example of the B2B interaction between *Aracron* and *Z&M*. The traditional model that *Aracron* follows to honor customer demand is based around the notion of *trust*. *Aracron* has trust relationships with several businesses in form of partnerships. These legal agreements lay the framework and rules of the interactions. The cooperating businesses act as facilitators and do not engage in competition with

Aracron. In *time sensitive* situations, if all *Aracron* partners are unable to fulfill the customer request, the order may be forwarded to a competitor (*Z&M*). The competitor acts in a manner analogous to any facilitator and is expected to exhibit the same level of trust. In the business centric model, the only difference between a facilitator and a competitor is the frequency of interactions. The facilitators are contacted on regular basis whereas a competitor receives sporadic requests, only in case of shortages. Since contracts are maintained between competitors that dictate the terms of use of the shared information, the risk of divulging trade secrets is low.

The business centric model is useful in situations where the number of business Web services is low. Contract management is easier when the number of participants is low. As mentioned previously, a business may need to *discover* new collaborators and invoke Web services "on-the-fly" to satisfy customer demand as predicted for the Semantic Web. Therefore, contract management may prove to be cumbersome and would not be effective on the Semantic Web. In our running example, the delay experienced by the customer under this model might be a little longer, as the requested item has to be first delivered to *Aracron* and then to the customer. In *time sensitive* requests this may not be feasible as delivery delays cause customer dissatisfaction.

3.2 Consolidated B2B Interactions

A consolidated B2B interaction model is implemented in the presence of a trusted intermediary. Traditional 'e-marketplaces' follow a similar interaction model. A place where buyers and sellers come together to interact with each other is known as an e-marketplace. A third party owns the e-marketplace which mediates the interactions and provides the required trust services. It does not compete with the businesses involved. The risk of sensitive business information to flow *out* of

its control may be minimal. Apart from hiding interaction specific sensitive information, the intermediary can also provide anonymity services to the buyer. Since the identity of the buyer is hidden, the seller business cannot target the specific source. This is comparable to the transaction model of *American Express*, where the credit agency provides its users with "unreal" credit card numbers (generated randomly) for each online transaction. This guarantees anonymity and safety of the customer information. No information about either the origin or the execution of order is provided to the end parties.

The consolidated interaction model architecture is based on the idea of having a specialized delivering third party like UPS, FedEX, AirBorne Express, etc. These third parties act as intermediaries to prevent the flow of sensitive information across the interacting businesses. Since they do not compete with the businesses, they are *entrusted* with their sensitive information. Consider the example of a request forwarded by *Aracron* to *Z&M* in the presence of an intermediary. The role of the intermediary is to hide customer information as it is *Aracron's* trade secret. To hide customer details, an "interaction identification number" (I-ID) is generated by *Aracron* and sent to *Z&M* along with the item request. If *Z&M* is able to honor the demand, the items are associated with the I-ID. The trusted third party receives the customer details from *Aracron* with an I-ID and the items from *Z&M* for that I-ID. It then matches the I-ID's and delivers the items to their appropriate locations. The potential problem of the model is that it relies heavily on the capabilities of the delivery agency to provide the required functionality. In a potentially hostile environment as the Web, delivery agencies may divulge sensitive information for their own gains. It would also be difficult to detect or manage such information disclosure.

The above mentioned models fail to protect secret information in cases where the number of participants cannot be determined *a priori* and no inter-party trust can be established. The emerging Semantic Web is such an environment where flow of information between previously unknown participants may not be monitored by

trusted parties. Also, the participants may form coalitions for a short period of time, to avoid the overhead and difficulty associated with contract management. It is expected that in the Semantic Web environment, data would be exchanged between potentially hostile participants who could benefit from divulging trade secrets.

3.3 Peer-to-Peer Business Interaction Model

In this thesis, we emphasize on the Peer-to-Peer (P2P) interaction model. In our running example, the interaction model should protect customer data from being used by the competitor while enabling the delivery of customer requested items.

In a P2P business interaction model, there is no third party involved. It is expected items would directly be delivered to the customer. implicit in this model is the use of traditional shipping entities such as the US Mail, Fedex, UPS, Airborne, etc. The businesses exchange order information and products are delivered to the customer by an independent shipping entity. Such a model of interactions can be best supported using the emerging Semantic Web services technologies. They are well suited as Web services can be *discovered* at run time and operate in a P2P fashion. Direct connection can be made with a Web service that "best" fits the need of the business. There is no need for extensive contracts or trusted intermediaries. However, this rich framework (Semantic Web services) provides a fertile ground for sophisticated attack on trade secrets, as sensitive information is shared with untrusted competitors. Therefore, mechanisms need to be established that facilitate direct B2B interactions while preserving trade secrets. We propose a *data perturbation* scheme to address this issue.

In traditional databases, data perturbation techniques attempt to preserve the privacy of individual items in situations where data mining techniques are used to draw conclusions from the data. The main idea behind data perturbation is that

the original data is replaced by *altered* values to protect the privacy of individual data items [88]. In general, *alteration* is achieved by swapping the original values with similar items or adding some *noise* to the data items [27]. For example, if X represents some sensitive data item, then it is altered by adding some noise (e) to result in the perturbed data item Y, $Y = X + e$ [2]. We may also replace X by a similar item, say X_1, to preserve the privacy of the data item. Thus, $Y = X_1$, where $X_1 \simeq X$ ($\simeq$ stands for "similar but not exact"). Usually data is *altered* in a manner that maintains the "aggregate meaning" of the original data distribution obtained by using data mining techniques. It also ensures that the "actual" data values are not easily identifiable.

The non-identifiable or perturbed data preserves the aggregate meaning of the data, so that relationships between various data values in form of item associations, classifications, clustering, predictions, estimations and deviations can be inferred and understood. The need for perturbation in case of B2B collaboration is different. In a B2B collaboration, the data may not be perturbed to an extent that it is completely *unidentifiable*. Actual data values may be needed to fulfill the customer orders. However, note that the aggregate relationships between data items may also divulge the trade secret of a business. Therefore, apart from protecting the *actual* data keeping its *originality*, such relationships must not be disclosed to preserve the privacy of the business [64]. The perturbed data should facilitate collaboration but *inhibit* any data mining. Such a technique is referred to as "fuzzing" the data [22].

3.3.1 Data Flow in the P2P Interaction Model

Delivery of items in the shortest time is a requirement in *time sensitive* transactions. Figure 3.1 shows the flow of data as a customer's request is satisfied through a competitor. The order is fulfilled in a series of steps. First, the customer sends an inquiry to the *source* business. A source business is one that is in need of protecting its list of loyal customers. It searches for the item in its inventory. If the item is

out of stock, then in the second step the request is forwarded to the competitor. If the competitor has the requested item, the source is notified. Third, customer credentials are gathered and passed to the competitor. Fourth, the requested items along with the customer information that the competitor possesses are sent to the delivery agency. In the fifth step, items are delivered to the customer.

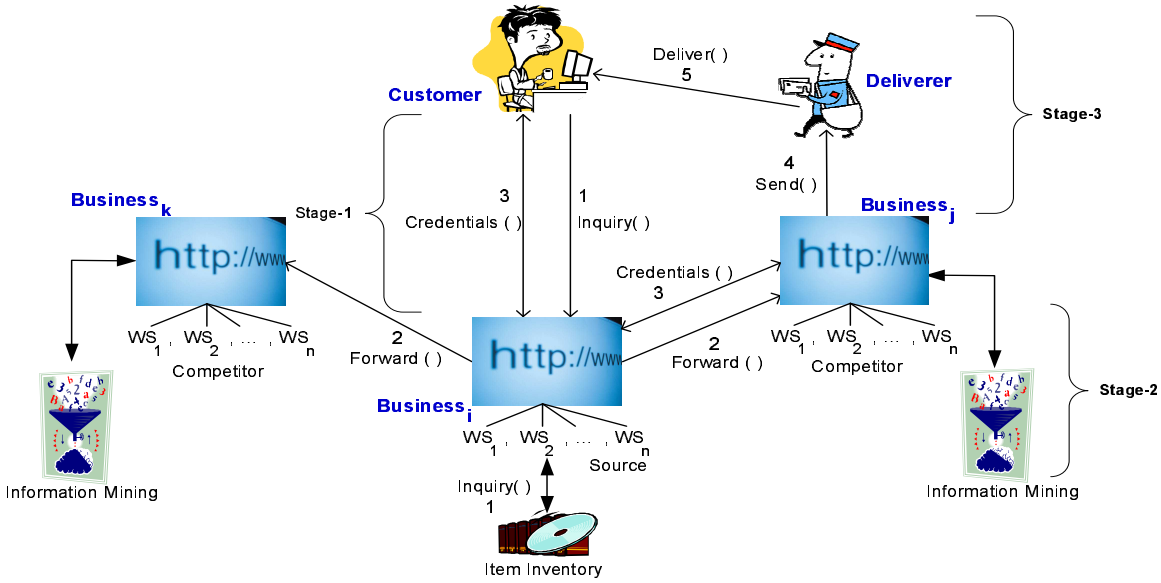

Figure 3.1: Data Flow in Request Fulfillment

In terms of data flow, a standard request may be broken into three distinct stages. In the first stage, the source receives the data. Second stage sees the data reach the competitor. In the third stage, the delivery agency receives customer data for delivery. The aim of a trade secret preservation scheme is to ensure that apart from stage one, sensitive data is not divulged in the two other stages. Customer data can be considered as "a means to an end" i.e., the orderly delivery of requested items. It serves a distinct purpose in each order stage. The stage functionalities are dependent on the data but are independent from each other.

Thus, the data may not stay *uniform* for all the stages. The idea is to *transform* the data in each stage such that it maintains its "usability" for that stage without hindering the data usability for other stages.

Figure 3.2 shows the theoretical specification of the proposed approach. The customer sends a request to an *Aracron* Web service which may need to forward it to a *Z&M* Web service. The shared data which may consist of sensitive attributes is represented by *Info*. The perturbation function (f_1) is performed at *Aracron* to transform the data to *P(Info)*. The aim of the perturbation function is to alter the data in a manner that the "new" values are distinct, i.e., *Info* $\neq$ *P(Info)*. The *Z&M* Web service can only utilize this data to honor the present request. The operations done by this *Z&M* Web service on *P(Info)* are represented by f_2 *(P(Info))*. Therefore, the data condition residing at the two hosts is represented by the in-equation f_1 *(Info)* $\neq$ f_2*(P(Info))*. It means that the data perturbed by the *Aracron* Web service does not reveal the same information at *Z&M*. The perturbed data that was received from *Aracron* is then forwarded to the delivery agency. In Figure 3.2, this is represented by P_2*(Info)*. The function on P_2*(Info)* at the delivery agency is represented by $f_3(P_2(Info))$. Since the shipping agency personnel are expected to *understand* the exchanged data, the data is almost similar to the original i.e., *Info* $\simeq$ P_2*(Info)* (where $\simeq$ stands for almost equal or "similar but not same"). Also, the functions applied by the *Z&M* Web service and the delivery agency yield different outcomes f_2 *(P(Info))* $\neq$ $f_3(P_2(Info))$. It should be noted that f_2 at *Z&M* is a cumulative function that gathers data over a period of time t, i.e., $f_2(P(Info)) = \sum_{i=1}^{n} P_i(Info)$. Therefore, every perturbation by the *Aracron* Web service over t should be distinct. This implies that from i=1 to n, no two P_i(Info) should assume the same value. If two or more perturbations, say P_j(Info) and P_k(Info) are equal ($j,k \in [1,n]$) then there is a risk of sensitive information disclosure. The employed perturbation technique should be able to handle such cases.

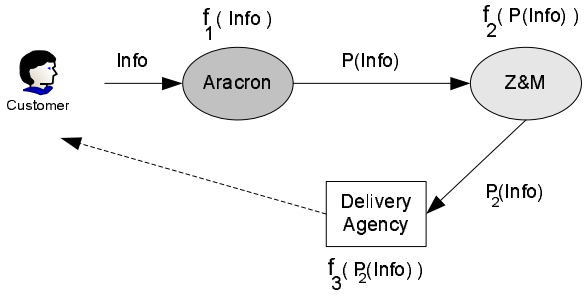

Figure 3.2: Theoretical Specification

Chapter 4

Perturbation Model

Several data perturbing methods for preserving data privacy have been proposed in the literature. Simple Additive Data Perturbation (SADP) [53], Correlated-Noise Additive Data Perturbation (CADP) [63] and Multiplicative Data Perturbation (MDP) [63] are a few examples. These techniques have focused mainly on numerical attributes. In our case, the customer's information may include only a couple of numerical attributes, i.e., the Zip code and the customer contact numbers. In addition, we do not perturb these numerical attributes. The postal systems of several countries prefer that the Zip code of the recipient be provided (Canada and U.S.A.) to speed up delivery. However, even in the absence of a valid Zip code, the items are successfully delivered [85].

In the U.S., a single Zip code holds about 25,000 different addresses on average [25]. We assume in our study that *direct* marketing techniques would not be carried out on such a large number of addresses. Therefore, it is safe to assume that the presence of correct Zip codes will not divulge private information. Note that the requesting business may decide not to give out customer contact numbers to its competitor. In cases where delivery agencies require a contact number, the business may opt to give its own contact number. In the event of a delivery agency needing to contact a customer, the business may contact the customer on behalf

of the deliverer. Thus, the risk of divulging this information to the competitor is almost non-existent. Also, the algorithm used for perturbing the attributes is not made public. It is only known to the requesting business.

The proposed perturbation techniques ensure that the *usability* of the data is *preserved after* the perturbation. We may classify these techniques into four classes: *character replacement* method, *word change* method, *character reordering* method and *hybrid perturbation method*. The focus of our work is the hybrid perturbation method as it provides the maximum number of possibilities of perturbation. A brief overview of each method is provided below.

4.1 Character Replacement Method

In this approach, we perturb the non-numeric data by replacing certain characters in a word. The characters to be replaced may be chosen at random. A character replacement method is somewhat similar to the "Morse code" scheme for transmitting textual messages. In such a coding scheme, each letter in a word is represented by a *known* character code. This facilitates the encoding and decoding of the message. On the other hand, the purpose of the *character replacement method* is to make the decoding process hard for the competitor while keeping it simple enough for the product shipper to decode and use. The basic idea behind this method is to replace a character by another *visually similar* one. To a human, "PIKE" and "PlKE" may appear the same. However, a software program makes distinctions between each character and hence the words. This method may be further subdivided into two methods based on the value of the *replacing* character.

In the first method, characters are replaced with similar characters from the same alphabet that is under use or from the numerical characters. For instance, if *Latin* alphabets are used, we may replace certain characters with visually similar characters from the same alphabet. A few examples are replacing a 'w' with 'vv', an

'I' with 'l' or 'o' with '0. This kind of perturbing technique provides limited options which may get exhausted quite easily and may present scalability problems. Thus, we need to have an extensive repository of available characters for replacement.

In the second method for character replacement, we may use visually similar characters from other alphabets. The basic problem that arises for this method is the translation and acceptance of those characters at the competitive host. If the competitor is not able to acknowledge the received characters, they may appear as "garbage", and may not be of use. Therefore, the interacting businesses need to agree upon a set of characters that are *valid* for the transaction. The "Unicode" standard [83] for representing characters may provide a solution to this problem. Unicode has been developed for the global sharing of data. It is a character set that enables data interchange between worldwide participants conforming to international standards. Unicode enables information from any of the defined languages to be stored using a single character set. This is achieved by having a unique code value for every defined character. It is worth noting here that a character is a representation of the smallest component of a language. A single character may be represented in many different ways. The different shapes in which a single character may be displayed are known as *glyphs*. A glyph is semantically different from a character. It is the different graphical representations of the *same* character that make up the glyph repertoire. For example, the character 'A' may have **A** and *A* as its glyphs among many more. The first row of Figure 4.1 shows some characters that may *appear* as an 'A', but are *different* characters when it comes to the Unicode standard. Unicode makes a distinction between characters and glyphs (which are not part of this standard). It is a widely adopted standard used by many software and hardware vendors. Major industry participants such as Oracle, Microsoft, IBM, and Sybase have incorporated it into their products. Similarly, it is required by several industry standards as XML, Java, LDAP, WML, etc.

In our proposed approach we use visually similar Unicode characters for perturbing the data. The data would remain useful even if some characters are re-

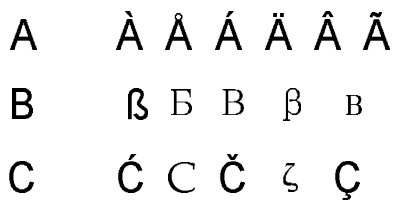

Figure 4.1: Different Options for Alphabet Replacement

placed by similar ones. Figure 4.1 shows the first three alphabets and a few of the options available to replace those characters with visually similar characters under the Unicode standard. Unicode defines a number of character representations. The first version allowed 65,536 characters to be represented using a 16-bit, fixed-width encoding scheme. The present version (Unicode 4.0) defined supplementary characters to incorporate those characters that were essential for global business markets. This enabled an additional 1,048,576 characters to be defined [83]. The new repertoire of defined characters increases the complexity of Unicode. However, it provides greater flexibility and ease for conducting global business. Note that not all characters are "valid" for replacement, i.e., they may be visually dissimilar to the *plain text* characters that are exchanged between businesses in B2B trade. It means that a significant number of Unicode characters may not be *useful*. Still, the available options would be sufficient for our perturbation needs. For example, consider the word "road" to be perturbed according to our proposed mechanism. Assume that *at a minimum*, each character in this word has five different character encoding options available. Normally, the *vowels* in Unicode have more than twelve options. Similarly, some other characters may have more than five options. However, for our current discussion it will suffice to assume five options for each character. Thus, the four letter word may be perturbed in 625 ways (5^4). Generalizing, we have:

$$N_1 = \prod_{i=1}^{n} P_i$$

where, N_1 is the number of perturbation possibilities, n is the total characters in a word and P_i is the number of perturbation possibilities for each ith character.

4.2 Word Change Method

The word change method employs techniques that change the "character composition" of the word. This could be done by either adding new characters to the word or deleting some characters from it. We may change the meaning of the word by making such changes. In case of *adding* characters to alter the word, such characters should be added that do not make the word incomprehensible or change its meaning. In our opinion, characters that are not visible on the display system may provide a practical solution. It would be safe to perturb the data by adding such *control* or *format* characters.

Control characters are not rendered on the display system (visible) themselves but influence the behavior of the displayed text. Unicode provides distinct code values for 64 control characters that are also defined in the ISO standards [83]. Most of these control characters are defined at an application level and Unicode does not explicitly dictate their use. Control characters are used for formatting, print commands, system controls, etc. A single control character may be interpreted differently in separate applications. For instance, a control character may be used to move the cursor to the beginning of a line in one application while in the other it may be used for highlighting text. Therefore, in order to provide *safe* perturbation, we need to choose control characters that have a minimal chance to create such a conflict with the applications under use. Unicode also provides several *layout* characters which are special control characters used for line breaking, word breaking, glyph selection, etc. Similarly, a separate category of characters

defined as *mathematical* operators are also defined. These are also not visible to the user but provide functions as multiplication, division, etc.

Perturbing the data with such control characters that are not visible on the display system (an LCD or through print) may provide greater flexibility in defining *dissimilar* words. The knowledge extraction methods mainly use item groupings, relations, associations, etc. In these methods, the *similarity* of items is of primary importance. This similarity (for mining purposes) is detected according to the presence and position of characters in a word. Thus, having an additional control character in a word would differentiate it with one that does not have it. For example, consider the word "Ottawa" to be perturbed. Assume that there is an *invisible* character ^D. We may perturb the data as "Ott^Daw^Da". The word would appear in its original form to the user on the display system but it would appear as a different word to the data mining program.

Removing a character from a word may also change its syntactic and phonetic composition. However, special care needs to be taken while choosing a character for removal. Just as adding, only those characters should be removed or deleted that do not change the meaning of the word altogether. The "redundant" characters in a word are considered *safe* for removal. These characters are used for phonetic emphasis. For example, "Ottawa" and "Otawa" sound the same while the latter word lacks a 't'. Similarly, "Jack" and "Jak" read almost the same. We propose to take advantage of such phonetic similarities among characters in a word to choose a character for removal.

Consider the example of the word "road", to be perturbed using this method. If we add control characters for perturbation, the number of possibilities increases rapidly. Since control characters may be inserted in any order (ordering does not matter as they are invisible), the number of possibilities becomes exponential. For instance, in a four letter word as "road", we may add control characters in *at least* five positions, i.e., -r-o-a-d- (the -'s indicating control character position possibilities). Note that, more control characters may be added between characters.

For a complete discussion of invisible character options please see the *analytical model* (Section 5.2).

To constrain our discussion to a reasonable minimum, we assume that we have only five possible positions for control characters. In other words, the number of control character slots for an 'n'-letter word is 'n+1'. Just as the alpha characters, we assume that for each control character *slot* we may have five different character options. The number of possibilities (N_2), for a number of (l) control characters is given by:

$$N_2 = ((n+1) \times l)!$$

4.3 Character Reordering Method

In this method, the order of characters in a word is altered as a perturbation technique. Earlier work on this method appeared in [74]. In that work, it was shown that humans understand and are able to reason about words even if the position of characters in a word is changed. For example, consider the following phrase: *it doesn't mttaer in waht oredr the ltteers in a wrod are, the olny iprmoetnt tihng is taht the frist and lsat ltteer be at the rghit pclae. The rset can be a toatl mses and you can sitll raed it wouthit porbelm.* A human is able to read such text without a problem because he can reason about the position of characters in a word, depending on its context. However, a software program cannot make such reasoning easily. Certain software systems may be able to recognize the *incorrect* words, e.g., intuitive spelling checking systems as the "google spell-checker". However, these can only *suggest* that a user "may" have made a mistake. These systems do not make corrections by themselves but require input from a human user. In a fully *automated* system, the software program may not be able to recognize the "rearrangement" of characters.

We assume that postal workers would be able to assess the correct word even

if it is perturbed in the described manner. Usually postmen are bound to some specific area of service. Over a period of time, they get accustomed to the area and its inhabitants. It enables them to reason about the complete address of a recipient or the actual recipient if some information is missing/incorrect. For example, if a postman has an "Ottawa Street" in his service area, then an address label that reads "Oattwaa Steret" is likely to reach its intended place. Since it is required that the first and last characters be in their original position, the number of possibilities (N_3), for a number of (n) characters is given by:

$$N_3 = (n-2)!$$

4.4 Hybrid Perturbation Method

The hybrid perturbation method is a combination of the three methods previously described. In our opinion, using a hybrid approach provides extensive perturbation options making it difficult for the competitor to *decode* the perturbation. The basic aim of our proposed perturbation scheme is to hinder any "unauthorized" disclosure of strategic information. We assume that competitors will employ methods to defeat the purpose of such perturbation. If only one proposed method is applied this may not pose significant processing challenges for competitors to extract the original data. A perturbation scheme should be designed in such a way that the *calculations* needed by the competitor to extract the original data are complex and for all practical purposes infeasible. In other words, the perturbation should be such that the procedure to extract useful and correct information is computationally expensive. Thus, we propose to use a hybrid technique that employs the methods we have presented previously.

A perturbation system that uses more than one perturbation method is robust because the competitor is not sure of record uniqueness. In absence of data cleansing, it cannot be certain about the original data and may not be able to filter out

the *noise*. To illustrate the extensive options available for perturbation using a hybrid scheme, assume that we have a system that uses both *visually similar* characters (as shown in Figure 4.1) and *control* characters for perturbation. Combining these two methods provides an effective tool for "encoding" a word. The aim of our perturbation technique is to make the inference procedure computationally *intensive*. If each word can be encoded in a variety of ways, the chances of revealing sales patterns or other sensitive information are slim. Using Unicode to perturb the customer information may provide a large number of options for each word to be encoded.

If we use a hybrid approach, the number of perturbation options is the product of the number of options obtained by employing the two techniques. Combining the perturbation options, we may have the minimal number of possibilities for each word (N) defined as:

$$N = N_1 \times N_2 \times N_3$$

$$N = \prod_{i=1}^{n} P_i \times ((n+1) \times l)! \times (n-2)!$$

Thus, for n=4 and five possibilities for each character, we may have 1.938901 x 10^{28} distinct words. The wide acceptability of Unicode means that it is safe to assume that replacement characters would not hinder the interaction. The "quality" of data, in terms of its use i.e., delivery of items to customer will be preserved. For example, an "A" could be replaced by any of the options listed in Figure 4.1. To the human reader (a delivery worker) any of the *A's* would read as an "A". However, to a computer program that organizes the data by making clusters, associations, classifications, etc., each character option would appear as a different character.

4.4.1 Example

In order to understand the different perturbation techniques and their applicability on the Semantic Web, consider the following example. Assume that a customer named "Homer Simpson" usually buys from *Aracron*. He decides to buy a new *Harry Potter book* as a Christmas present for his son Bart. Homer may postpone buying the item up to the last moment due to a variety of reasons. For example, he may be waiting for extra discounts or just busy with work. When he decides to place his order, he naturally chooses *Aracron* as it has provided satisfactory service in the past. He may interact with its provided Web services to request an item. It may be the case that due to high demand, *Aracron* has already sold all of its stock. It is assumed that such shortages are handled automatically by *Aracron*'s Web services.

Since *Aracron* is keen on attracting and keeping customers coming back, it decides to honor Homer's demand. It would typically put a request to the UDDI registry to locate an appropriate Web service. Due to the variety of service providers, a competitor's Web service may also be a potential satisfier. In case of a *match*, i.e., an appropriate service is selected, the next step is to invoke it. We assume that the order is forwarded to *Z&M* by invoking its advertised Web service. Figure 4.2 defines a standard WSDL representation for a "BookSearch" operation that has an input and output message defined for searching for a particular book. After the book is found, customer data is exchanged between Web services for delivery purposes.

To protect its customer data (*trade secret*), the *Aracron* Web service perturbs the data before sharing it. As explained earlier, not all attributes are *safe* for perturbation. Therefore, the perturbation mechanism chooses the sensitive attributes. In this case, it is the customer's information. The correct credentials of Homer are *First Name*: Homer, *Last Name*: Simpson, *Street Address*: 570 Krusty Street, *City*: Springfield, *State*: VA, *Zip Code*: 525000, *Phone*: 42-446-2409. These include seven

```
<message name="getBookRequest">
        <part name="BookID" type="xs:string"/>
</message>

<message name="getBookResponse">
        <part name="Availability" type="xs:string"/>
</message>

<portType name="BookSearch">
        <operation name="getBook">
                <input message="getBookRequest"/>
                <output message="getBookResponse"/>
        </operation>
</portType>

<binding type="BookSearch" name="BindBookSearch">
        <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />
        <operation>
                <soap:operation soapAction="http://www.zm.com/getBook"/>
                <input>
                        <soap:body use="literal"/>
                </input>
                <output>
                        <soap:body use="literal"/>
                </output>
        </operation>
</binding>
```

Figure 4.2: Standard WSDL for *Z&M*

attributes, of which two (numeric Zip code and phone) are not to be perturbed. *Aracron* provides its own phone number instead of the customer phone number (300-423-1200). The remaining five attributes are selected for perturbation. We assume that the numerical part of the street address is also not perturbed. The perturbed data is then passed on to the delivery agency Web service in the same form that *Z&M* Web service received it. Figure 4.3 shows the SOAP request for sending the customer information. The response of this SOAP request is an acknowledgment that the information has been correctly received. Note that in line 3 of Figure 4.3, the "char-set" has been defined as *UTF-16*. This is the Unicode

character set representation that includes all defined characters. This character set allows to encode and thus perturb data in numerous ways. Although UTF-8 may also be used here, it is preferred that UTF-16 is chosen to extend the perturbation possibilities.

```
POST /Search HTTP/1.1
Host: www.bn.com
Content-Type: text/xml; charset=utf-16
Content-Length: nnn

<?xml version="1.0"?>
<soap:Envelope xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
        <soap:Body xmlns:m="http://www.zm.com/getBook">
                <m:GetRecieveStatus>
                        <m:FirstName>Hómër</m:FirstName>
                        <m:LastName>$impsôň</m:LastName>
                        <m:StreetAddress>570 Krµstÿ Śtręėt</m:StreetAddress>
                        <m:City>Sρringfiεld</m:City>
                        <m:State>VÂ</m:State>
                        <m:Zip>525000</m:Zip>
                        <m:Phone>300-423-1200</m:Phone>
                </m:GetRecieveStatus>
        </soap:Body>
</soap:Envelope>
```

Figure 4.3: SOAP Request Which Returns Status

Figure 4.4 illustrates the effects of different perturbation techniques on Homer Simpson's data. Clearly, all options produce data which is different from the original. We may also employ a hybrid of all the proposed techniques to increase our available options for perturbation.

Original	**Character Replacement**		**Word Change**		**Reordering**
	Same Alphabet	*Different Alphabet*	*Add Character*	*Remove Character*	
Homer Simpson 570 Krusty Street Springfield, VA 525000 42-446-2409	Honner Slmqson 570 Krvsty Strcct Sprlmqfield, VA 525000 300-423-1200	Hómër $impsôň 570 Krµstÿ Ŝtręê† Spring*f*ıeld, VA 525000 300-423-1200	Homeer Simppson 570 Krrusty Streeet Spprinngfield, VA 525000 300-423-1200	Homr Smpson 570 Krsty Stret Sprinfeld, VA 525000 300-423-1200	Hmoer Spimson 570 Kutrsy Sretet Spingrfiled, VA 525000 300-423-1200

Figure 4.4: Perturbation Outcomes

Chapter 5

Experiments and Implementation

5.1 Experiments

As a proof of concept, we have conducted *real life* experiments using our proposed methods. The main purpose of the experiments was to show that the proposed data perturbation techniques preserve the "use" of data. Note that the use of data refers to the efficient and timely delivery of products. The experiments involved ordering of a variety of items (mainly office supplies) through a number of actual e-businesses. An order can be divided into three distinct stages namely, order placement, order processing and order delivery. The order placement stage is where the customers select the product and give their private information for delivery purposes. In the order processing stage, the business performs the necessary functions on the collected data, e.g., cleansing, mining, etc. The order delivery stage is one in which the items are collected from the business warehouse and sent to the customers through a delivery agency.

Since our experiments involved B2C interactions, the perturbations were done at the order placement stage. We used the hybrid perturbation method involving the three proposed methods. In using the *character replacement* method, the re-

placement characters from a different alphabet were chosen from the Windows-1252 character set. In our proposed techniques we assume that both businesses would use the *Unicode* character set. However, due to the present situation where not every business acknowledges Unicode, we opted for the widely accepted Windows-1252 character set. The products were ordered only from Web sites that provide services in the United States. The defined characters in Windows are few as compared to Unicode, but for the experiments, it serves the purpose. In employing the *word change* method, we also did not add any control characters due to the compatibility reasons.

In B2C scenarios, the order placement and order processing stages have a slight overlap. The business may decide to cleanse the data before the order is placed, e.g., it may not accept any "perturbed" characters. A convenient way in which businesses do this is by matching the customer address with the one provided by the postal service through commercial address matching softwares. The results of our experiments show that only 8% of the business population conduct data cleansing in this manner. The customer address is matched against a valid U.S. street address and if any "invalid" characters are found, the address is labeled unacceptable. Such systems are also able to catch and eliminate any *slight* modifications to the data by stripping off *space* or *unrecognized* characters. For instance, an address label that should read as *Elite Lane*, if input as *Elite Lana*, etc. is automatically corrected by the system. However, the label *Elite Laneg* or the like are not corrected and are deemed unacceptable. The high percentage of businesses (92%) that do not employ such cleansing techniques is a testimony to the fact that these businesses allow room for "human error." Note that even with the availability of street address matching software, businesses seldom use them. The foremost reason for not using such software is the high volume of trade these businesses have to support. In online retailing, businesses devise strategies to reduce the customer *wait-time*. If such address matching techniques are used, customers may have to face longer waits. As mentioned earlier, unintentional mistakes may be caused due to human

error, e.g., not remembering the zip code, providing an incomplete street address, missing a character in last name, etc. Another reason for not employing address matching methods is to allow room for such errors. Also, the primary objective of businesses is to collect revenue. Thus, major business resources are directed in the direction of securing payments, checking credit details, etc. Businesses are not concerned whether a given address is correct or not, as long as they are certain of their product revenue collection through a credit agency. Moreover, in international business settings, such address matching softwares may not be available at all to conduct such checks.

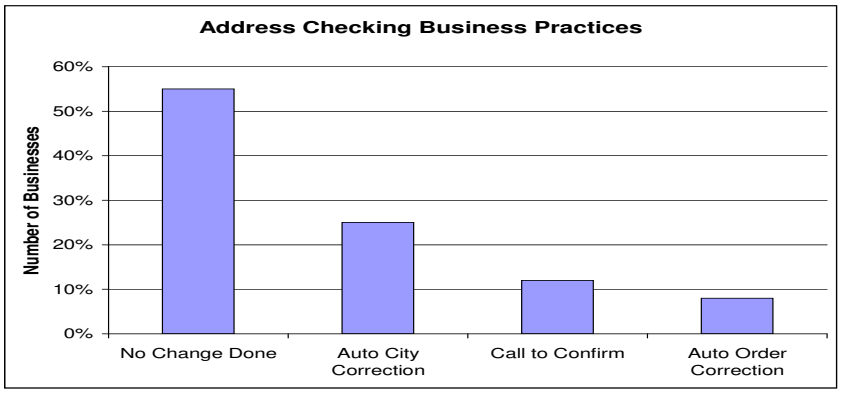

Figure 5.1: Business Practices

Our experiments show that a significant number of businesses (37%) perform *data checking* in the order processing stage. The experiments reveal that only "small" e-businesses tend to check the validity of the addresses. The smaller businesses emphasize address validation because they are not sure of revenue collection in case of incorrect addresses. The businesses have to go through lengthy procedures to recover from credit fraud. Hence, it is more profitable to go through each order for record correction rather than face lengthy credit recovery procedures. Since large businesses receive a greater number of orders, to manually go through

each order is a lengthy procedure. They tend to avoid such checks. Note that out of the 37% businesses that do some address *correction* or *verification*, only 12% businesses actually call the customer to verify the correct address if the perturbed address is not found in the address database. We have found that calling a customer is the last resort for the businesses. The small businesses manually match the customer's perturbed address against an address database and only if they are not able to read the characters, they call to make sure of the delivery address. This happens when the front-end ordering system operates on a character set different from the back-end ordering system. The *Latin* character set is universally accepted and businesses seldom face the issue of compatibility. However, in cases as the one proposed in this thesis, the use of a uniform character set is essential. Also, it is found that the remaining 25% of e-businesses only validate and correct the destination *city* and *zip code*. The street address and customer name is not validated or corrected. This proves our underlying assumption that delivery personnel do not need *exact* addresses. It is also found that 55% of the businesses do not validate the data at all and accept the customer input as provided.

The third stage is the order delivery. Our experiments show that 99% of the orders reach the customers in the expected amount of time. This essentially means that the perturbation does not hinder the delivery mechanisms. The deliveries that we received were performed by the three major vendors, i.e., UPS, Fedex and USPS. None of the delivery agencies had a problem delivering merchandise with "invalid" street addresses, city names or even zip codes. It is believed that the delivery agency's internal problems, rather than the perturbation schemes were to blame for the 1% orders that were not delivered in the required time frame and were delayed by four working days. These orders had the correct city, state and zip code information and only the street addresses were perturbed.

Figure 5.2 shows six different address delivery labels for our orders. The *numbered* boxes indicate the address labels that were received from the vendor, while the actual addresses are shown in boxes below each label. We have changed the

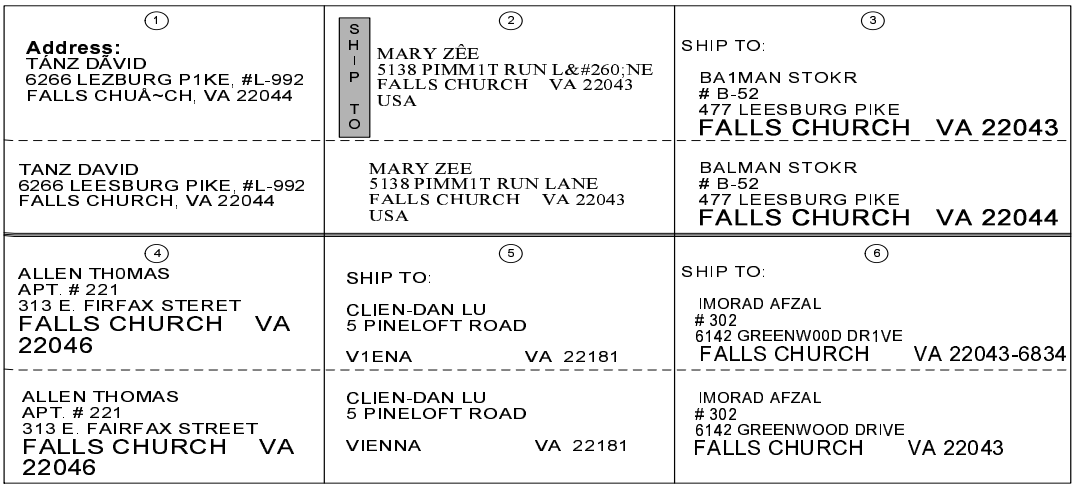

Figure 5.2: Different Order Labels

original names and numbers for privacy reasons. However, the underlying perturbation techniques are left as they were.

Label 1 : Characters Replaced

Label 1 shows an address label that was perturbed for most of the required attributes, i.e., the name, street address and city are all perturbed. The latter "A" is replaced by Á and Ã in the name, while the letter "I" in 'PIKE' is replaced by a 1. A "Z" is put in place of "ES" to take advantage of the phonetic similarity.

Label 2 : Incompatible Types

Label 2 shows an example of *incompatible* front-end and back-end ordering systems. As it can be seen from the real address, the character "A" in 'LANE' has been replaced by characters "& # 260 ;" . These characters represent the hexadecimal encoding of the replacement character as input in standard HTML. Since, the back-end system is unable to read HTML style hexadecimal code, it is passed on

as different latin characters.

Label 3 : Incorrect Zipcode
In label 3, we have perturbed the first name and the zip code information. The *in-time* delivery of the item indicates that if the city name is correct, then an invalid zip code would not hinder the delivery. A correct zip code may facilitate operations at the delivery agency for order pre-sorting, etc. but it shows that such slight *errors* are over looked by the delivery agency.

Label 4 : Illegal Characters Not Accepted
Label 4 provides an example of a business that does not accept "illegal" characters. As it can be seen, the real street address is "FAIRFAX STREET". We perturbed the data by replacing the first character 'A' and *reordering* the 'street'. The order was processed at the front-end (HTML browser) but the replacement character was not recognized by the back-end system. It was stripped off from the word, hence "FIRFAX".

Label 5 : Incorrect City
Label 5 is an example of an address that had an incorrect city name. The city name "VIENNA" was perturbed with 'I' replaced by a '1' and one 'N' stripped off. The resulting word had the same phonetics though.

Label 6 : Auto City Correction
Label 6 shows the auto city correction methods employed by the 25% of businesses. The real address label was perturbed by Cyrillic characters in the city name. The system could not understand those characters and informed the user. After the real city name was input, the zip code information was *adjusted* to have the 4-digit extension code. Note that the street address was not validated, when it was written as "GREENW00D DR1VE" with two zeros and a 1 instead of two O's and an I in

"GREENWOOD DRIVE".

5.2 Analytical Model

The aim of our proposed approach is to perturb the data in a manner that is computationally intractable for a competitor to *decode*. In this section, we present the analytical model for the proposed perturbation system and show that the competitor decoding time is exponential. Table 5.1 defines the parameters and symbols used in this section.

Performance measurement variables	
S_{At}	Number of sensitive attributes in a message
P	Number of perturbations possible per character
k	Number of characters per attribute
S_F	Size of an attribute field in the database
A_P	Total perturbations for an attribute
T_C	Time to compare two words for perturbation similarity
l	Number of invisible perturbation characters
n	Number of records in the competitor database
T_{comp}	Competitor decoding time

Table 5.1: Symbols and Variables

Figure 5.3 shows the steps for perturbing a message. It is obvious that the encoding time for the business intending to preserve its *trade secret* is minimal and almost constant. We assume that the time T_C, for comparing two words for similarity T_C is a fixed value. As shown in the previous section, a large percentage of businesses do not employ any data cleansing techniques on the orders received.

In absence of such techniques, a possible way for the businesses to "extract" information is by employing some record *linkage* method. Record linkage focuses on the process of determining the similarity between records based on some pre-defined criteria [30]. Our proposed methods make the process of record linkage practically infeasible. The time required by the competitor to *link* two "similar" records, and to extract any useful information would be non-linear.

The presence of *Unicode* characters in an order would mean that a competitor that would want to mine incoming customer data (*trade secret*), would need to first transform the incoming order string to establish a *link* with the previously received orders. One way to transform the incoming string is to compare each instance with the existing orders in the database and remove/replace and inconsistencies. However, note that due to the presence of Unicode characters, the competitor cannot be sure about the inconsistent data characters. As each Unicode character is treated distinctly, the inconsistencies are hard to reveal. The records would need to be compared with the entire database and the the process would be repeated n times. In the following, we show that the time needed to perform such decoding operations is exponential. Note that not all businesses follow the same practice. As evidenced by our experiments, 25% of businesses automatically correct the city and zip code information. We also consider a "constrained" decoding computation time for the cases where the competitor knows the correct city and zip code information.

The number of possible attribute perturbations using the hybrid approach is the product of the number of perturbations obtained from the three perturbation methods described previously. The number of characters of an attribute and the number of available perturbation options for each character are used to calculate the number of possibilities for a *character replacement* method. $\prod_{i=1}^{k} P_i$ gives the total available options for a k-character attribute. In calculating the options for the *word change* method, we take into account only the addition of invisible characters to make our analysis simple as altering an attribute by adding or removing visible characters may change the meaning of the word. Since the invisible characters

```
(01) For each sensitive attribute A_k, where 1 ≤ k ≤ n, do
(02)   A_i ← Compute Sensitivity (A_k)
(03) For each sensitive attribute A_i, where i ≤ k, do
(04)   For all previous perturbations of A_i for present Collaborator,
do
(05)     MakeList B_i of perturbed characters
(06)   if B_i is null
(07)     Perturb randomly
(08)   else
(09)     do
(10)       Perturb
(11)     while Perturbation P_i = B_i
(12)      Add(A_i, x, CollaboratorID) after successful perturbation to
history
```

Figure 5.3: Steps for Message Perturbation

can be added to the attribute in any order, we need to constrain ourselves with the maximum number of invisible characters allowed for addition. In traditional databases, the length of fields is usually fixed. It is safe to assume that the data exchange methods also abide by such a restriction. Thus, the maximum number of invisible characters that may be added is $S_F - k$, where S_F is the maximum length of allowed characters in the competitor database and k is the number of characters in the attribute. Since, the invisible characters can be added randomly in any position in the word, for a total of l invisible characters, the available perturbation options is $((S_F - k) \times l)!$. For illustration purposes, we are using $k+l$ as the possible number of options for invisible character additions. As mentioned earlier, if the first and last character of a word are left *undisturbed*, the meaning of the word can be understood through its context. Therefore, we subtract 2 from the possible perturbation number. The total number of possible reorderings of characters in

an attribute is $((k+l)-2)!$. Combining the three methods, the total number of available perturbation options is then given by:

$$A_p = \prod_{i=1}^{k} P_i \times ((S_F - k) \times l)! \times ((k+l)-2)! \tag{5.1}$$

To calculate the time required by the competitor to decode messages T_{comp} and establish any relationships, it may need to consider the *similarity* of records in the historical database. The similarity between messages may only be drawn on the basis of similar character positions and perturbations. The different numbers would depend on the total number of attributes that are perturbed. If A_{p_i} is the total number of perturbations for each attribute i, then the comparisons needed by a record with S_{At} attributes would be:

$$\prod_{i=1}^{S_{At}} A_{p_i} \tag{5.2}$$

To find out all the possible matches with previous interaction records, for a number of n records, the competitor would have to do as many comparisons. The complete transaction history database would have to be searched for possible matches. The competitor performs the total comparisons represented by equation (1) for each attribute. Therefore, the competitor computation time is given by:

$$T_{comp} = T_C \times \prod_{i=1}^{S_{At}} A_{p_i} \times n \tag{5.3}$$

Figure 5.4 shows the time it takes for the competitor to parse through all available options. It is shown that the time taken by the competitor increases with an increase in the number of available options to perturb a character. The x-axis shows the number of available perturbation options per character, and the y-axis shows the competitor decoding time. Time taken by one perturbation can be considered as a unit of time.

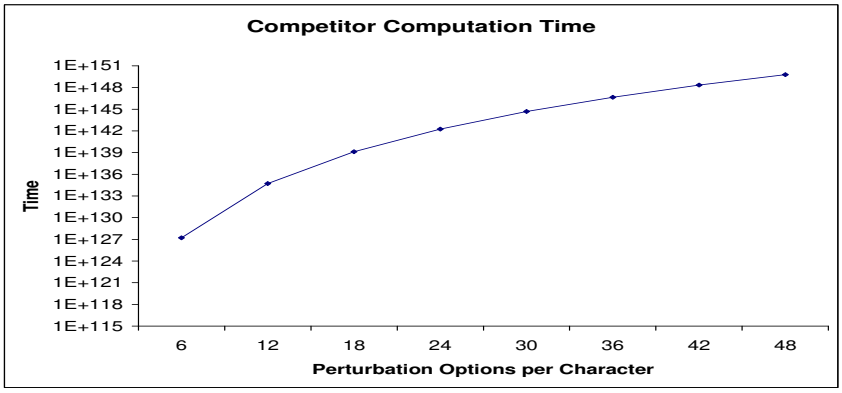

Figure 5.4: Competitor Estimated Decoding Times

Figure 5.5 shows the time it takes for a competitor to parse through all available options and compare them with the record history for similarity between records. The plot is obtained using values of n from 100 to 1000000000, where n is the number of records in the history database. Usually, data mining techniques are applied over large repositories of data. Note that all other options of the equation stated above are assumed constant in calculating competitor decoding time. In reality, these number may also change which would have a direct negative effect on competitor's computation time.

Figure 5.6 shows the effect of setting the value of n to a fixed 25,000. As explained earlier, up to 25% businesses may auto-correct the city and zip code information. If this is the case, then instead of iterating through the whole database, the addresses can be compared against "valid" street addresses in a particular zip code. On average, a single zip code in a metropolitan area has 25,000 individual addresses. It can be seen that the number of computations required by the competitor increases even if the correct city and zip code are known.

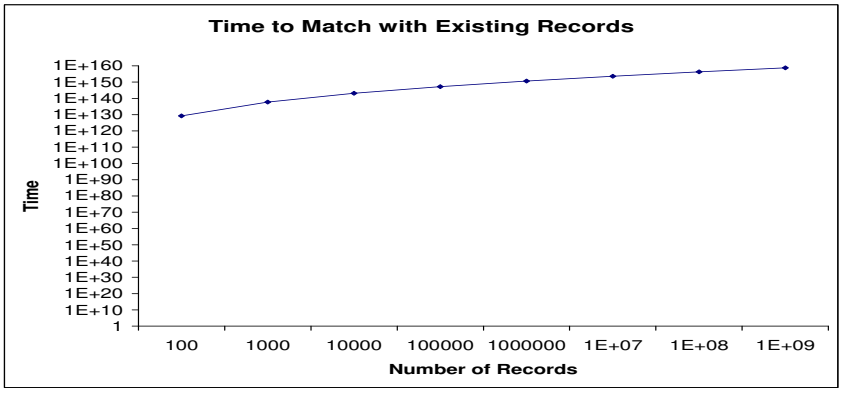

Figure 5.5: Competitor Computation Time to Match with Existing Records

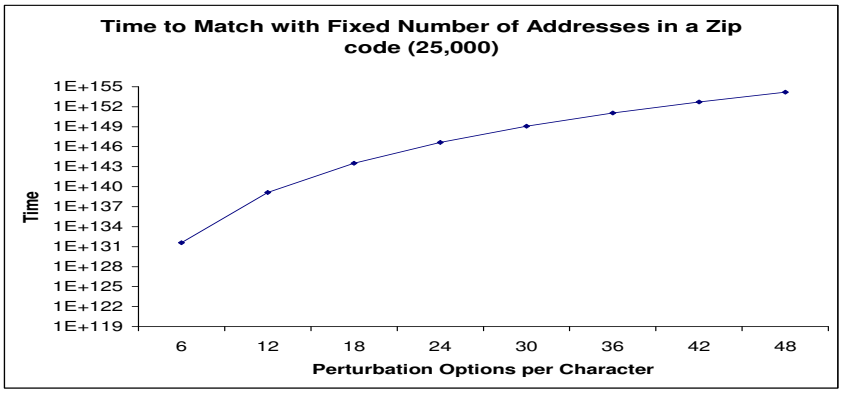

Figure 5.6: Competitor Computation Time to Match with Fixed Number of Addresses

5.3 Implementation of the P2P Trade Secret Architecture

We have implemented our proposed methods in the WebBIS system. We have used the Web services technology to represent the participating businesses. The *Web-*

BIS system architecture consists of a P2P Trade Secret interface, a trade secret manager and service area. The *WebBIS interface* (HTML/Servlet) is a graphical user interface (GUI) that provides a point-of-access to the P2P Trade Secret system. It sends user requests to the trade secret manager that is responsible for their control and execution. Web services are used to provide automatic integration functionality for *bookseller* applications. Figure 5.7 shows seven (7) businesses that offer services through the P2P Trade Secret system. Each service accesses a database (or a set of databases) on the back-end to retrieve requested item information. Partner Web services as *Aracron*, *RabzBook* and *Books4Sale* are shown under one cloud. The services that fall under the same *trust ontology* are expected to hold legal contractual agreements. Trade secret preservation mechanisms may not be necessary while communicating with the Web services from the same trust ontology due to the potential trust conditions imposed by the contracts.

Web services are used as "wrappers" for the bookseller proprietary applications. Standards such as WSDL, UDDI and SOAP are used for maintaining Web services. To automatically generate WSDL descriptions from Java class files for the applications, *Axis Java2WSDL* utility from IBM's Web Services Toolkit is used. WSDL service descriptions are published into a UDDI registry. Systinets WASP UDDI is used as the toolkit. P2P Trade Secret Web services are deployed using *Apache* SOAP. It provides not only server-side infrastructure for deploying and managing services but also client-side API for invoking those services. Each service has a *deployment descriptor*. The descriptor includes the unique identifier of the Java class to be invoked, session scope of the class, and operations in the class available for the clients. Each service is deployed using the *service management client* by providing its descriptor and the URL of the *Apache SOAP Servlet rpcrouter*. Note that for seamless integration and proper business processing, it is assumed that Web service communication (SOAP requests/responses) uses a Unicode character set, i.e., UTF-16. Figure 4.3 provides an example snippet for such a SOAP request.

The *trade secret manager* is at the core of the P2P Trade Secret system. It is

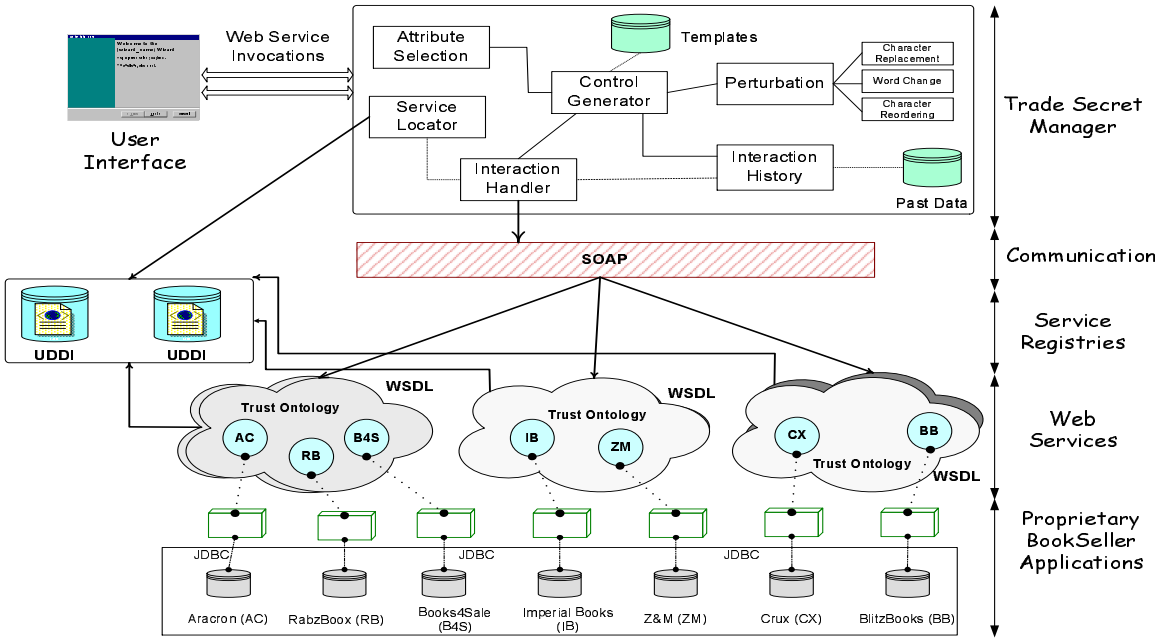

Figure 5.7: P2P Trade Secret Architecture

composed of the following modules: Attribute Selection Module, Control Generator Module, Interaction History Module, Perturbation Module, Service Locator Module, and Interaction Handler Module.

The *Attribute Selection* module selects the attributes that are suitable for perturbation based on the nature of the request. The *Control Generator* module computes the allowed perturbation levels for those attributes. It uses the template information and historical information from past interactions. This information is obtained from the *Interaction History* module that keeps a record of complete past interactions. The exact parameters of invocations, business identities, etc. are stored in the database. The *Perturbation* module is responsible for altering the data in a manner that renders it useless for the competitor. The *Service Locator* allows the discovery of WSDL descriptions by accessing the UDDI registry. It

implements *UDDI Inquiry Client* using WASP UDDI API. Once a service is discovered, its operations are invoked through the *Interaction Handler* using *SOAP Binding Stub*, which is implemented using Apache SOAP API.

Figure 5.8: The WebBIS Customer Interface

The interface of WebBIS is shown in Figure 5.8. It consists of an HTML page which is generated by a servlet. In our implementation of the system, we have two business clients that may play the role of hosts. *Aracron* and *Crux* are two of the businesses that a customer could go to for ordering books. The customer interface at both sites is similar and provides similar functionality. The purpose of having more than one business provide similar functionality is to show the effect that a

single business could perform the actions of both the requester and the satisfier. In the role of the requester, the business would go to other competitors to satisfy a customer demand. Similarly, as a satisfier, the business would be responsible for honoring other business' demands, if the need arises. In Figure 5.9, the HTML page is shown that is generated upon the customer inquiry. Note that the customer is oblivious to the fact that his request may be fulfilled by a competitor or any other business.

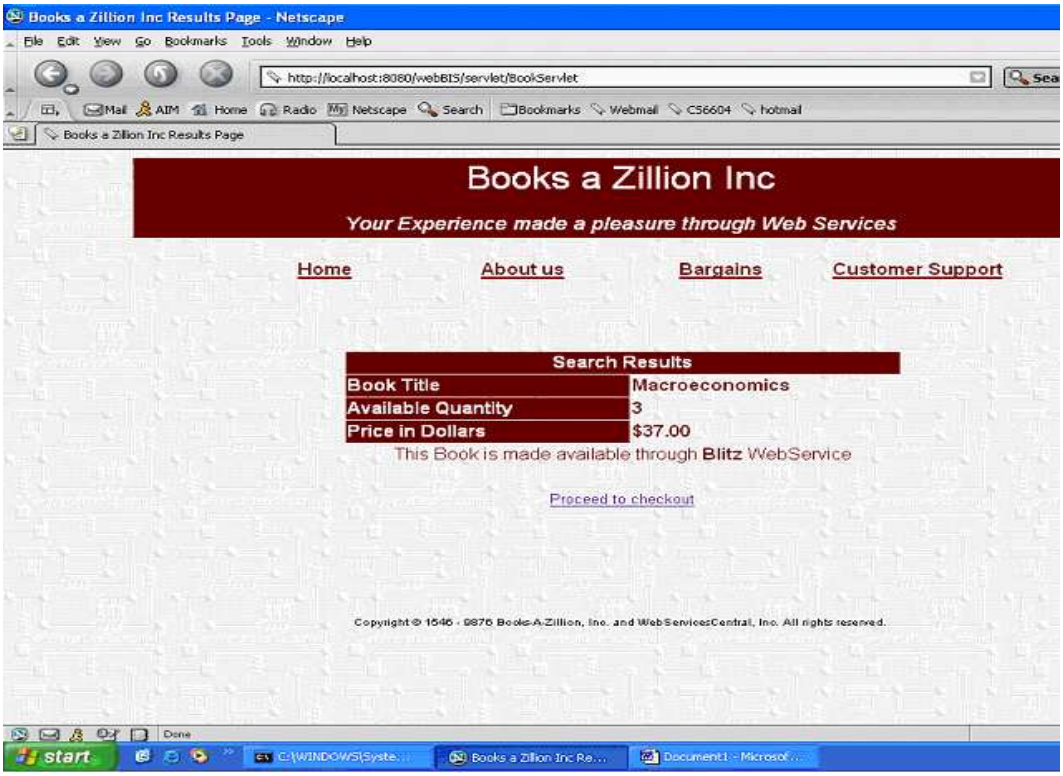

Figure 5.9: The Book Details Page

Figure 5.10, shows the WebBIS page which is presented to the customer to fill in his 'vital details'. This information, along with the customer request may be

shared with a competitive business. Thus, this information needs to be perturbed.

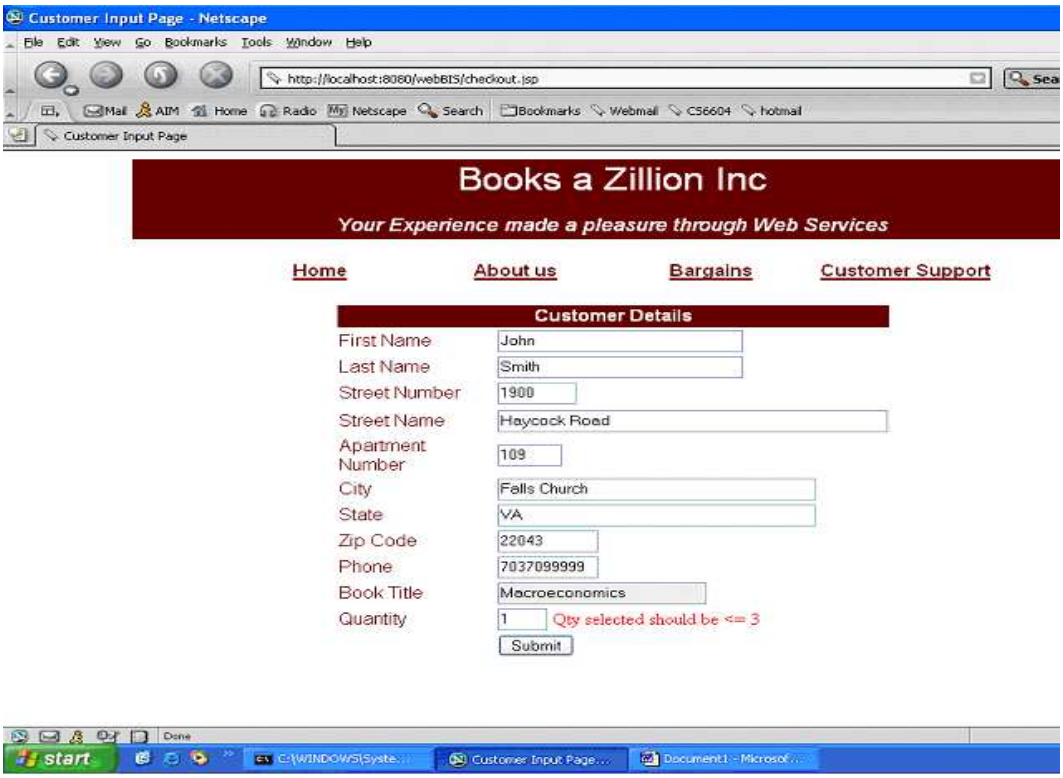

Figure 5.10: Information Taken from the Customer to Perturb

In Figure 5.11, the perturbed information is shown. Note that this information is not shared with the customer. Only a confirmation page is displayed. However, we have included the following figure for illustrative purposes in this thesis.

We have populated the different WebBIS business databases with more than 100,000 customer records. The motive behind having a large number of records is to provide a reasonable data bank for the data mining application used. We have employed the *StarProbe* [78] data mining software to check the applicability of our proposed approach. "StarProbe is a star schema based cross platform data mining

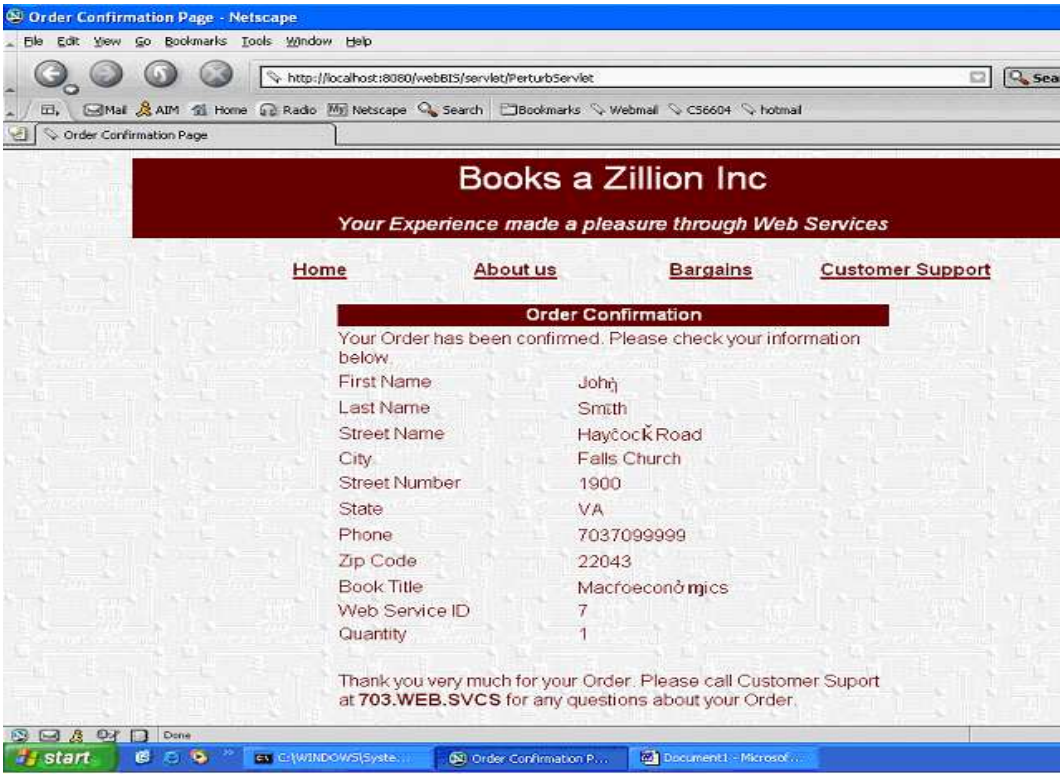

Figure 5.11: Perturbed Customer Information

software system that works smoothly with most common database systems and incorporates statistics, machine learning, data warehousing, OLAP and business intelligence, providing hotspot analysis, predictive modeling (decision tree, neural network, rule induction), neural clustering, visualization, statistics, regression and correlation analysis [78]. In our case, neural clustering is the technique that can be used to reveal the associations between customers and relate between the *same* (perturbed) customer records.

Figure 5.12 shows the results of employing clustering techniques on the perturbed data set. It can be seen that there are 27 clusters obtained from the

customer ordering data. This number is far less than the 1000 clusters that should have been obtained (the original number is obtained using StarProbe on unperturbed data). Note that we employed the data filtering techniques available in StarProbe to cleanse the data of any anomalies. Still, not many clusters were found. This backs our earlier assumption that the proposed data perturbation scheme is effective for more than 90% of the time. However, note that it is not a proof that our scheme is fail-safe. In Figure 5.12, each square represents a data cluster. The different colors represent the number of records in each cluster. We have labeled some squares for illustration while others are left blank.

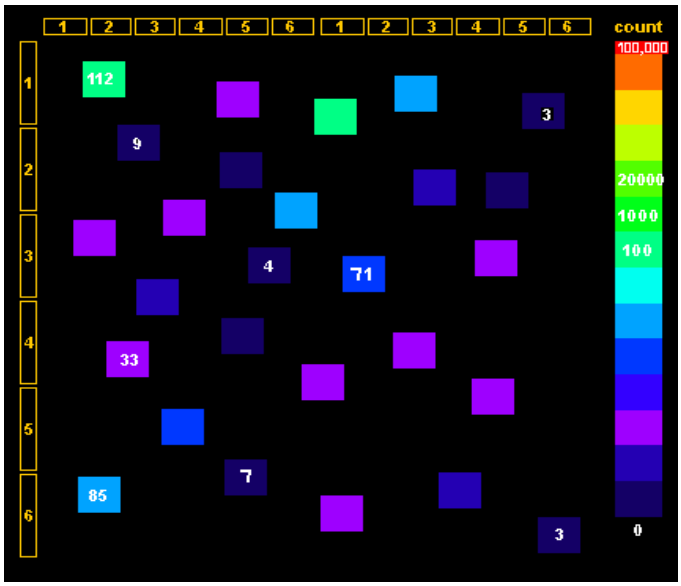

Figure 5.12: Clusters Obtained through StarProbe

Chapter 6

Related Work

In this chapter we present a brief overview of the various B2B interaction practices. We start the chapter by listing some research prototypes representing different technologies supporting B2B interactions. In the latter half of the chapter, we provide an analysis of the different works on privacy related to trade secrets of businesses.

6.1 Research Prototypes

As mentioned in the previous chapters, B2B integration has taken a central stage in the research community. due to the inadequacies of the traditional technologies, new systems are consistently emerging. In the following, we provide a brief overview of the efforts aimed at B2B integration.

6.1.1 CMI

CMI (*Collaboration Management Infrastructure*) [81, 36] provides an architecture for inter-enterprise workflows. The main components of CMI engine includes the *CORE*, *coordination* and the *awareness engines*. The *CORE engine* provides basic primitives used by the coordination and awareness engines. These primitives

include constructs for defining resources, roles, and generic state machines. CMI's *coordination model* extends the traditional workflow coordination primitives with advanced primitives such as *placeholder*. The concept of placeholder enables the dynamic establishment of trading relationships. A placeholder activity is replaced at runtime with a concrete activity having the same input and output as those defined as part of the placeholder. A selection policy is specified to indicate the activity that should be executed. If multiple providers offer implementations for an activity interface, the selection policy may use a broker to choose the implementation that offers the "best" quality of service. CMI's trading partners are tightly-coupled. For example, the message format and the communication protocol to be used between partners must be agreed upon before service activity definition. Heterogeneity is addressed through object-oriented proxies which enable access to different information sources such as relational databases, EJBs, and CORBA objects. CMI provides application-specific state machines and operations for modeling services. This allows for the selective monitoring of state changes in external services. CMI addresses security only at the process model level through a role-based process and activity execution.

6.1.2 eFlow

eFlow [19] is a platform that supports the specification, enactment, and management of composite services. A composite service is described as a process schema that combines basic or composite services. A composite service is modeled by a graph, that defines the order among the nodes in the process. It may include *service*, *decision*, and *event* nodes. Service nodes represent the invocation of a basic or composite service. The definition of a service node contains a *search recipe* represented in a query language. When a service node is invoked, a search recipe is executed to select a reference to a specific service. Decision nodes specify the alternatives and rules controlling the execution flow. Event nodes enable service

processes to send and receive several types of events.A *service process instance* is an enactment of a process schema. To support heterogeneity of services, eFlow provides adapters for services that support various B2B interaction protocols such as OBI and RosettaNet.

6.1.3 WISE (Workflow based Internet Services)

WISE [80, 55] aims at providing an infrastructure for the support of cross-organizational business processes in *virtual enterprises*. WISE architecture is organized into four components: process *definition*, *enactment*, *monitoring* and *coordination*. The *process definition* component allows *Virtual Business Process* (VBPs) to be defined using as building blocks the entries of a catalog where companies within a trading community (TC) can post their services. The *process enactment* component compiles the description of the VBP into a representation suitable for enactment and controls the execution of the process by invoking the corresponding services of the TC. The *process monitoring* component keeps track of the progress made in the execution of the VBP. The information produced by this tool is used to create an awareness model used for load balancing, routing, quality of service, and analysis purposes. The *process coordination* component supports multimedia conferencing and cooperative browsing of relevant information between all participants in the TC.

6.1.4 Mentor-Lite

Mentor-Lite [99] addresses the problem of distributing the execution of workflows. The idea is to partition the overall workflow specification into several subworkflows, each encompassing all the activities that are to be executed by a given entity within an organization. The basic building block of Mentor-Lite is an interpreter for workflow based on state charts. Two other modules are integrated with the workflow interpreter defining the workflow engine: *communication man-*

ager and *log manager*. The *communication manager* is responsible for sending and receiving synchronization messages between the engines. It uses the *Transaction Processing* (TP) monitor *Tuxedo* for delivering synchronization messages within queued transactions. The *log manager* provides logging and recovery facilities. A separate workflow log is used at each site where a workflow engine is running.

6.1.5 SELF-SERV

SELF-SERV [9, 82] proposes a process-based language for composing Web services based on *state charts*. It also defines a *peer-to-peer* Web service execution model in which the responsibility of coordinating the execution of a composite service is distributed across several peer components called *coordinators*. The coordinator is a lightweight scheduler which determines when a state within a state chart should be entered and what should be done when the state is entered. It also determines when should a state be exited and what should be done after the state is exited. The knowledge needed by a coordinator to answer these questions at runtime is statically extracted from the state chart describing the composite service operations and represented in the form of routing tables.

6.2 Work on Business Privacy

The trade secret preservation system proposed in our thesis is novel in the sense that none of the previous works has focused on the problem of protecting trade secrets in B2B environments involving competitor collaboration. The proposed system provides an *automatic* trade secret preservation mechanism. It takes advantage of the human perceptual system and exploits the lack of the aforementioned in machines. To the best of our knowledge, none of the previous work has provided a solution for trade secret preservation in P2P Web services collaboration among competitors.

There has been very little work in regard to the privacy of data in B2B activities. Majority of the work focuses on marketplaces, agreements, payment mechanisms or procurement solutions with little or no mention of privacy [52, 18, 39, 87]. The problem of preserving privacy from a competitive business' point of view is discussed in [11] and [34]. However, neither of these work focuses on a no-intermediary P2P model. Also, both attempts fail to address the issue completely. Only the potential problems are discussed in these works. There is no mention of a possible solution to the problems faced in B2B environments that involve trade among competitors.

A notable exception regarding the work on trade secrets flowing from one business to another is available in [77]. The work emphasizes the risk involved in divulging some sensitive information to the business employees with the possibility of them *defecting* to a competitor. The proposed model does not provide a technical solution to the problem of protecting trade secrets. It only provides a theoretical specification which involves employees who may defect to a competitor with some sensitive information. The potential loss and future gains of defection are evaluated in the paper to reach an *ideal* mix that defines when and how much of the sensitive information needs to be divulged to the employees. This model is not directly applicable to the Web as the nature of trade secrets in an online setting is inherently different. The risk of the divulging trade secrets due to the necessary data flow is higher than an employee's defection.

Preserving the disclosure of data to unwanted recipients has motivated the research in the field of databases. The work has centered around the desire to provide the required data without compromising the sensitivity of the data items [2]. The sensitivity of data items means that individual data values are not identifiable. Data perturbation is one of the proposed techniques for protecting the sensitivity of the data. Methods of data perturbation include swapping record values, replacing original distribution with sample values, adding noise to the data and adding random perturbations to the query results among a few other [3, 4]. The *use* of

data is not lost, as the individual data values are not important for knowledge extraction in form of aggregations, averages, etc.

In large data sets, knowledge extraction mainly depends on the ability of the system to effectively *link* various records. It may be the case that due to "errors" two or more similar records appear distinct. These errors can either be accidental, e.g., input of a character not intended, or they could be intentional, e.g., perturbation. In [50], the problem of record linkage is explored. The solution attempts to link two strings that may be "same". The strings are mapped to a Euclidean space and then similarity joins are performed in that space to link various records. The effective linkage depends on the robustness of the distance metrics used. For instance, the Levenshtein distance [56] has been taken as a commonly-used distance metric. We have run the techniques proposed in [50] on a data set obtained by using our proposed techniques. The hybrid perturbation method was found to be more than 90% effective in preventing any linkage among various records. One possible reason could be the various reorderings of the attributes. The other techniques when considered individually also did not show the same amount of accuracy (99%) as shown in the paper.

Other solutions to the problem of record linkage can be found in [6, 40, 41, 46, 54]. An optimal solution to this problem has been proposed in [92]. The proposed solution tackles the record linkage problem by dividing it into two distinct steps: the searching of potentially linkable pairs and the decision whether or not a given pair is correctly matched. The first step requires the *conditioning* of the data and the determination of the required parameters by the decision model. The conditioning may transform all the data values to either lowercase or uppercase and then compute the Soundex code. This is done to remove any anomalies in the data due to the change of case of a character. However, our proposed scheme accomplishes this individuality by defining each Unicode character distinctly. Moreover, manual inspection of records may be needed for correctly labeling the vectors involved in making the comparison decisions. In a situation where millions of records are

present, this scheme is clearly infeasible.

The problem of inferring sensitive data from non-sensitive data using data mining is explored in [21]. Privacy is accomplished by ensuring that the competitor only holds a sample of the original data. Neither the original data nor the distribution is revealed. In [88], the need for developing mining tools that are powerful yet do not violate privacy and security is motivated. The nature of a B2B collaboration is such that *real* data needs to be shared among businesses. We cannot transform the data in a manner that individual values are not identifiable, as a data item may constitute the name of an item, its required quantity, demanded price, etc. In this thesis, we have tried to solve this particular problem by providing a novel solution, using data perturbation.

Chapter 7

Conclusion

In this chapter, we summarize the results of our thesis and discuss future research directions for B2B Web service Interactions.

7.1 Summary

Businesses need to conduct trade with competitors to obtain or maintain their market advantage [15]. In course of a business interaction, *unwanted* information may flow to *undesired* recipients. The recent advances in Web technology mean that inter-enterprise interactions could involve "many participants". Different business functionalities could be out-sourced to various businesses that may include competitors. Web service technology is fast evolving as the enabling technology [60]. However, current techniques for B2B interactions are error prone with respect to the flow of secret trade specific information. In this thesis, we proposed methods for the automatic control of such secret information. This requires *perturbing* the information in a manner that preserves its "use" but inhibits any knowledge extraction. We also implemented our approach in *WebBIS*, a prototype for accessing e-business Web services.

The work in this thesis focuses on e-business cases that involve delivery to the

consumer. The delivery is done by a third party agency whose sole responsibility is to deliver and it does not engage in any other competition. The first step of our research was thereby understanding the delivery process. Various studies have shown that subtle syntactic and semantic changes to the "normal text" does not hinder the human perception to a great extent [74]. This find forms the basis of our research. We have proposed to *transform* the text in a manner that changes it from its original form, but at the same time does not lose its usage. The proposed *perturbation-based* framework is such that would not hinder the delivery process.

We have developed three major perturbation techniques. The fourth perturbation technique is a combination of all and has proved to be more robust. In a business activity that involves the customer list and customer preferences as potential trade secrets, the extraction of such knowledge is of primary significance. Thus, the basic purpose of any trade secret preservation technique is to prevent any form of knowledge mining and extraction. The proposed perturbation techniques transform the data in a manner such that the same record would appear as distinct. Knowledge extraction methods normally work by "aggregating" similar records. Our proposed perturbation techniques transform the records in a manner that do not allow such aggregation and clustering. There have been several attempts to provide *efficient* record linkage techniques. We do not say that our proposed techniques defeat the record linkage in its entirety. However, the proposed techniques are aimed at making the process of knowledge extraction practically intractable. We use the *Unicode* character set in defining and transforming the records.

In summary, we presented a trade secret preservation system for B2B Web services. The proposed system aims at automating the process of trade secret protection in B2B interactions using Web services. The solution is based on the general model for perturbation where several characters in a word are altered to conceal original items. The perturbation technique is employed so that the data maintains its usability and confidentiality throughout the interaction.

7.2 Future Directions

For B2B E-commerce to scale to the Internet, there is a need for efficient integration with all relevant partners, established *a priori* or on demand. The need for interoperability in B2B applications is more pronounced than usual partly because of the way businesses operate, the systems they have, and the difficulties created by systems' autonomy and heterogeneity. Although the current technologies provide the foundation for building B2B integration frameworks, several research issues still need to be addressed. These include *process-based integration of services*, *dependable integration of services*, *support of standardized interactions*, *security*, and *privacy*. In the following, we provide a brief overview of the open issues related to security and privacy as these are directly related to our work.

7.2.1 Security

Security is a critical issue that must be dealt with in B2B E-commerce. Security must be enforced to give businesses the confidence that their transactions are safely handled. A few *de facto* standards are available for transport-level security (e.g., SSL) and message-level security (e.g., SMIME). However, issues such as authentication and authorization still need to be addressed. Businesses generally perform controls over the internal use of their business processes. In B2B E-commerce, there is a need to extend this controlled access to outside companies' boundaries. The concept of access control, traditionally studied in the context of databases, must be thoroughly investigated in the context of B2B applications. Research on specifying, validating, and enforcing access control policies for B2B applications is one where intensive work is needed. In particular, access control should be performed at both the database and application levels. In addition, businesses generally have to deal with various and even contradictory access control policies while transacting with their partners.

7.2.2 Privacy

Privacy refers to the restriction of knowledge about various pieces of business transactions to parties involved in the transactions. It is generally (mis)perceived as an issue whose *natural* solution consists of good security mechanisms. Although security and privacy are two tightly interrelated issues, secure B2B frameworks do not necessarily ensure privacy [59, 75]. The importance of the privacy problem does not seem to have triggered the right level research efforts. In fact, few techniques and standards have addressed the issue of preserving privacy in Web-based applications. One such standard is W3C's *Platform for Privacy Preferences Project* [94] (P3P). However, P3P enables the specification of the privacy of Web sites but *not* B2B applications. Worse, P3P provides no mechanisms that guarantee that Web sites actually implement their stated privacy policy. A major issue in B2B E-commerce is the ability for businesses to understand each others' privacy policy. There is also a need to provide mechanisms to address how the privacy policy of integrated services is derived from the individual policies of the trading partners.

In our future work, we intend to extend our research to other business interaction models and explore the proposed techniques in context of different business interaction scenarios. We would also explore various data mining techniques and devise strategies that could make the proposed methods *fail-safe*. We have experimental evidence that the techniques we have used cannot be easily *cracked* in polynomial time. However, we intend to build on these findings and devise strategies that would make the knowledge extraction procedure even more intractable. In our current work, the presence of humans (for delivery purposes) is a requirement. We would like to extend the research in this direction, where trade secrets could be preserved even without the involvement of humans.

Bibliography

[1] N. Adam, O. Dogramaci, A. Gangopadhyay, and Y. Yesha. *Electronic Commerce: Technical, Business, and Legal Issues*. Prentice Hall (ISBN: 0-13-949082-5), August 1998.

[2] N. R. Adam and J.C. Worthmann. Security-Control Methods for Statistical Databases: A Comparitive Study. *ACM Computing Surveys*, 21(4), December 1989.

[3] D. Agrawal and C. Aggarwal. On the design and quantification of privacy preserving data mining. In *Symposium on Principles of Database Systems*, pages 247–255, 2001.

[4] R. Agrawal and R. Srikant. Privacy Preserving Data Mining. In *Proceedings of the ACM SIGMOD Conference on Management of Data*, May 2000.

[5] G. Alonso, F. Casati, H. Kuno, and V. Machiraju. *Web Services: Concepts, Architectures and Applications*. Springer-Verlag, 2004.

[6] W. Alvey and B. Jamerson. Record Linkage Techniques. In *International Workshop and Exposition, Federal Committee on Statistical Methodology*, 1997.

[7] Amazon. What sales rank mean. In *www.amazon.com*, August 2004.

[8] ATIS. *EDI Guideline Consistency Subcommittee (EGCS).* http://www.atis.org/atis/tcif, August 2004.

[9] B. Benatallah, M. Dumas, M. Sheng, and A. H. H. Ngu. Declarative Composition and Peer-to-Peer Provisioning of Dynamic Web Services. In *ICDE Conference*, February 2002.

[10] B. Benatallah, B. Medjahed, A. Bouguettaya, A. Elmagarmid, and J. Beard. Composing and Maintaining Web-based Virtual Enterprises. In *1st VLDB TES Workshop*, September 2000.

[11] Maria Bengtsson, Susanna Hinttu, and Soren Kock. Relationships of Co-operation and Competition between Competitors. In *19th. Annual AMP Conference*, September 2003.

[12] T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Scientific American*, May 2001.

[13] M. Bichler, A. Segev, and J. L. Zhao. Component-based E-Commerce: Assessment of Current Practices and Future Directions . *ACM SIGMOD Record*, 27(4), December 1998.

[14] A. Bouguettaya, B. Benatallah, and A. K. Elmagarmid. *Interconnecting Heterogeneous Information Systems.* Kluwer Academic Publishers (ISBN 0-7923-8216-1), July 1998.

[15] Adam M. Brandenburger and Barry J. Nalebuff. *Co-opetition: A Revolution Mindset that Combines Competition and Cooperation.* Doubleday Publishers, 1995.

[16] Budden, Micheal C., Jones, Micheal A., and Connie. Supplier Relationships and the Trade Secrets Dilema. *International Journal of Purchasing and Materials Management*, 32(3), Summer 1996.

[17] C. Bussler. B2B Protocol Standards and their Role in Semantic B2B Integration Engines. *Bulletin of the Technical Committee on Data Engineering*, 24(1), March 2001.

[18] Jiannong Cao, Tony Fong, Heng Li, and Xuerong Wang. E-Union: Concept and framework of Open B2B e-Trading Marketplaces. In *IEEE IPDPS*, 2002.

[19] F. Casati, S. Ilnicki, L.-J. Jin, V. Krishnamoorthy, and M.-C. Shan. eFlow: a Platform for Developing and Managing Composite e-Services. Technical Report HPL-2000-36, HP Laboratoris Palo Alto, 2000.

[20] F. Casati and M.-C. Shan. Process Automation as the Foundation for E-Business. In *VLDB Conference*, September 2000.

[21] C. Clifton. Using sample size to limit exposure to data mining. *Journal of computer security*, 8(4), 2000.

[22] C. Clifton and D. Marks. Security and Privacy Implications of Data Mining. In *ACM SIGMOD Workshop on Research Issues on Data Mining*, May 1997.

[23] E. Cobb. The Evolution of Distributed Component Architectures. In *CoopIS Conference*, September 2001.

[24] cXML. http://www.cxml.org.

[25] Zip-Codes Database. Zip Facts and Figures. In *http://www.zip-codes.com/zips*, August 2004.

[26] U. Dayal, M. Hsu, and R. Ladin. Business Process Coordination: State of the Art, Trends, and Open Issues. In *VLDB Conference*, September 2001.

[27] Dorothy E. Denning. *Cryptography and Data Security*. Addison-Wesley, 1982.

[28] ebXML. http://www.ebxml.org.

[29] eCO. http://eco.commerce.net.

[30] M. Elfeky, V. Verykios, and A. Elmagarmid. Tailor: A Record Linkage Toolbox. In *18th. International Conference on Data Engineering ICDE'02*, 2002.

[31] eMarketer. E-Commerce Trade and B2B Exchanges. http://www.emarketer.com. April 2003.

[32] Fastwater. http://www.fastwater.com.

[33] M. Fisher. Introduction to Web services. Java Web services tutorial. *http://java.sun.com/webservices/docs/1.0/tutorial/*, August 2002.

[34] Joshua Gans and Stephen King. Competition Issues Associated with B2B E-Commerce. In *Core Reserch, Competition and Regulatory Economics*, September 2001.

[35] D. Georgakopoulos, editor. *Information Technology for Virtual Enterprises*, the 9th International Workshop on Research Issues on Data Engineering, March 1999.

[36] D. Georgakopoulos, H. Schuster, A. Cichocki, and D. Baker. Managing Process and Service Fusion in Virtual Enterprises. *Information Systems*, 24(6), 1999.

[37] Elaine Hardcastle. Dixons UK half year sales flat, silent on Christmas. In *Reuters*, November 2003.

[38] C. Heilman, K. Nakamoto, and A. Rao. Pleasant Surprises. *Journal of Marketing Research*, 39(2), May 2002.

[39] Paul Hempel and Ying Ki Kwong. B2B e-Commerce in emerging economies. *Journal of Strategic Information Systems*, 10(1), 2001.

[40] M. A. Hernandez. A Generalization of Band Joins and the Merge/Purge Problem. In *Ph.D. Thesis. Department of Computer Science, Columbia University*, 1996.

[41] M. A. Hernandez and S. J. Stolfo. Real-World Data id Dirty: Data Cleansing and the Merge/Purge Problem. *Data Mining Knowledge Discovery*, 2(1), 1998.

[42] D. Hollinsworth. *The Workflow Reference Model*. Brussels, Belgium, November 1994. TC00-1003, http://www.aiai.ed.ac.uk/WfMC/DOCS/refmodel/rmv1-16.html.

[43] Mike Honor, Fraser Pearce, Paul O'Connell, and Ioanna Stagia. Customers Are For Life, Not Just For Christmas. In *TechStrategy-Forrester Research*, January 2001.

[44] J. Hopkins. Component Primer. *Communications of the ACM*, 43(10), October 2000.

[45] HP. *NetAction*. http://www.hp.com.

[46] J. A. Hylton. Identifying and Merging Related Bibliographic Records. In *Master's Thesis, department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology*, 1996.

[47] IBM. *WebSphere*. http://www.ibm.com.

[48] IETF. http://www.ietf.org, August 2004.

[49] IETF. *Electronic Data Interchange - Internet Integration (EDIINT)*. http://www.ietf.org, August 2004.

[50] Liang Jin, Chen Li, and Sharad Mehrotra. Efficient Record Linkage in Large Data Sets. In *Eighth International Conference on Database Systems for Advanced Applications (DASFAA'03)*, 2003.

[51] R. Kalakota and A. B. Whinston. *Frontiers of Electronic Commerce*. Addison Wesley (ISBN: 0-201-84520-2), February 2000.

[52] Robert Kauffman and Hamid Mohtadi. Information Technology in B2B E-Procurement: Open Vs. Proprietary Systems. In *IEEE HICSS-35*, 2002.

[53] J. Kim. A Method for Limiting Disclosure of Microdata Based on Random Noise and Transformation. In *ASA Proceedings Survey Research Methods*, 1986.

[54] B. Kliss and W. Alvey. Record Linkage Techniques. In *Workshop on Exact Matching Methodologies, Department of the Treasury, Statistics Income Division*, 1985.

[55] A. Lazcano, G. Alonso, H. Schuldt, and C. Schuler. The WISE approach to Electronic Commerce. *International Journal of Computer Systems Science and Engineering*, 15(5), September 2000.

[56] V. I. Levenshtein. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. In *Soviet Physics Doklady*, volume 10, 1966.

[57] S. M. Lewandowski. Frameworks for Component-based client/server Computing. *ACM Computing Survey*, 30(1), March 1998.

[58] Jane Lin. Christmas: Consumer's season. In *Analysis in Brief - Statistics Canada - No.11-621-MIE2003007*, December 2003.

[59] B. Medjahed, A. Rezgui, A. Bouguettaya, and M. Ouzzani. Infrastructure for E-Government Web Services. *IEEE Internet Computing*, 7(1), January 2003.

[60] Brahim Medjahed. Semantic Web Enabled Composition of Web Services. In *PhD Thesis, Computer Science Department, Virginia Tech. University*, 2004.

[61] B. Meyer. On To Components. *IEEE Computer*, 32(1), January 1999.

[62] Microsoft. *Distributed Component Object Model (DCOM).* http://www.microsoft.com.

[63] K. Muralidhar, D. Batra, and P. Kris. Accessibility, security and accuracy in statistical databases. In *Management Science*, volume 41, 1995.

[64] Krishnamurty Muralidhar and Rathindra Sarathy. A theoretical basis for perturbation methods. *Statistics and Computing*, 13, 2003.

[65] P. Muth, D. Wodtke, J. Weissenfels, A. K. Dittrich, and G. Weikum. From Centralized Workflow Specification to Distributed Workflow Execution. *Journal of Intelligent Information Systems*, 10(2), March 1998.

[66] Nasa. *Scientific and Engineering Workstation Procurement (SEWP).* http://www.sewp.nasa.gov.

[67] United Nations. *United Nations Directories for Electronic Data Interchange for Administration, Commerce and Transport (UN/EDIFACT).* http://www.unece.org/trade/untdid/welcome.htm, August 2003.

[68] Netscape. *Secure Socket Layer (SSL) 3.0 Specification.* http://wp.netscape.com/eng/ssl3/.

[69] OASIS. http://www.oasis-open.org/cover, September 2002.

[70] OBI. *OpenBuy.* http://www.openbuy.org.

[71] Hellen K. Omwando, Jaap Favier, and Tim van Tongeren. Europe's Online Christmas sales bring Good Tidings. In *TechStrategy-Forrester Research*, November 2003.

[72] R. Orfali and D. Harkey. *Client/Server Programming With Java and CORBA*. Wiley Computer Publishing (ISBN: 047124578X), March 1998.

[73] Sunil Prabhakar Radu Sion, Mikhail Atallah. Rights Protection for Relational Data. In *Proceedings of the ACM SIGMOD*, June 2003.

[74] Graham E. Rawlinson. The significance of letter position in word recognition. In *PhD Thesis, Psychology Department, University of Nottingham*, 1976.

[75] A. Rezgui, M. Ouzzani, A. Bouguettaya, and B. Medjahed. Preserving Privacy in Web Services. In *the 4th International ACM Workshop on Web Information and Data Management*, November 2002.

[76] E. Roman, S. W. Ambler, and T. Jewell. *Mastering Enterprise JavaBeans*. Wiley Computer Publishing (ISBN: 0471417114), December 2001.

[77] Thomas Ronde. Trade Secrets and Information Sharing. *Journal of Economics and Management Strategy*, 10(3), Fall 2001.

[78] Rosella. *StarProbe. http://www.roselladb.com/starprobe.htm*, August 2004.

[79] RosettaNet. http://www.rosettanet.org.

[80] C. Schuler, H. Schuldt, G. Alonso, and H.-J. Schek. Workflows over Workflows: Practical Experiences with the Integration of SAP R/3 Business Workflows in WISE. In *Proceedings of the Informatik'99 Workshop "Enterprise-wide and Cross-enterprise Workflow Management: Concepts, Systems, Applications"*, October 1999.

[81] H. Schuster, D. Baker, A. Cichocki, D. Georgakopoulos, and M. Rusinkiewicz. The Collaboration Management Infrastructure. In *ICDE Conference*, March 2000.

[82] M. Shen, B. Benatallah, M. Dumas, and E. O.-Y. Mak. SELF-SERV: A Platform for Rapid Composition of Web Services in a Peer-to-Peer Environment. In *VLDB Conference*, August 2002.

[83] The Unicode Standard. www.unicode.org. August 2004.

[84] Sun. *Java RMI (Remote Method Invocation)*. http://java.sun.com/products/jdk/rmi.

[85] United States Postal System. Addressing Tips and Tools. In *http://zip4.usps.com/send/preparemailandpackages*, 2004.

[86] C. Szyperski. *Component Software - Beyond Object-Oriented Programming*. Addison-Wesley (ISBN: 0-201-74572-0), November 2002.

[87] L. G. Telser. A Theory of Self-Enforcing Agreements. *The Journal of Business*, 53(1), Jan. 1980.

[88] B. Thuraisingham and M. Ceruti. Understanding Data Mining and Applying it to Command, Control, Communications and Intelligence Environments. *IEEE*, 2000.

[89] Department Of Trade. Uniform Trade Secrets Act 1 (4).

[90] Department Of Trade. Restatement of Torts Sec.757. 1939.

[91] S. D. Urban, S. W. Dietrich, A. Saxena, and A. Sundermier. Interconnection of Distributed Components: An Overview of Current Middleware Solutions. *Journal of Computer and Information Sciences and Engineering*, 1(1), March 2001.

[92] V. S. Verykios, G. V. Moustakides, and M. G. Elfeky. A Bayesian Decision Model for Cost Optimal Record Matching. *The VLDB Journal*, December 2003.

[93] W3C. *Extensible Markup Language (XML)*. http://www.w3.org/XML.

[94] W3C. *The Platform for Privacy Preferences Specification (P3P)*. http://www.w3.org/TR/P3P.

[95] W3C. Web Services Architecture. *W3C Working Draft*, August 2003.

[96] W3C. *Simple Object Access Protocol (SOAP). http://www.w3.org/TR/soap*, August 2004.

[97] W3C. *Universal Description, Discovery, and Integration (UDDI). http://www.uddi.org*, August 2004.

[98] W3C. *Web Services Description Language (WSDL). http://www.w3.org/TR/wsdl*, August 2004.

[99] J. Weissenfels, M. Gillmann, O. Roth, G. Shegalov, and W. Wonner. The Mentor-Lite Prototype: A Light-Weight Workflow Management System. In *ICDE Conference*, February 2000.

[100] X12. *EDI (Electronic Data Interchange) ANSI X12*. http://www.x12.org.

[101] XML/EDI. http://www.xmledi-group.org, August 2004.