A Transaction Assurance Framework For Web Services

A dissertation submitted to the Department of Computer Science,
Faculty of Science at the University of Cape Town
in fulfilment of the requirements for the degree of

Master of Science

in
Computer Science

— Reinhardt van Rooyen —

Supervisor
Adjunct Professor Andrew Hutchison

June, 2007

ABSTRACT

Trust assurances for customers of online transactions is an important, but not well implemented concept for the growth of confidence in electronic transactions. In an online world where customers do not personally know the companies they seek to do business with, there is real risk involved in providing an unknown service with personal information and payment details. The risks faced by a customer are compounded when multiple services are involved in a single transaction.

This dissertation provides mechanisms that can be used to reduce the risks faced by a client involved in online transactions by allowing the him/her access to information about the services involved and control or prescribe how the transaction uses the services. The dissertation uses electronic transactions legislation to ground a trust assurance protocol and minimize the assumptions that have to be made. By basing the protocol on legislation, no information that isn't already required by law is used in the protocol.

A trust assurance protocol is presented so that the client can establish which services are involved in a transaction so that the he/she can begin to determine whether or not he/she is willing to conduct business with the services. A trust model that calculates an assurance measure for services is developed so that the client can automatically establish a measure of trust for a service based on the external perceptions of a service, and his/her own personal experience. A simulation environment was created and used to monitor the services involved in a transaction to evaluate the trust assurance protocol and gain experience with the trust calculation that the client computes.

Vocabularies that simplify and standardize descriptions of personal information, business types and the legal structure imposed on Web services offering goods or services online are presented to reduce the ambiguity involved in gathering information from different online sources. The vocabularies also provide a cornerstone of the trust assurance protocol by providing information that is necessary to compute the trust value of a Web service.

Results of the trust assurance protocol are obtained and evaluated against the qualitative requirements of providing assurances to clients, and confirms that the protocol is feasible to be deployed, in terms of the overhead placed on a transaction. This dissertation finds that a trust assurance protocol is necessary to provide the client with information that he/she legally has

access to and that the trust model can provide a calculable measure of trust that the client can use to compare Web services.

ACKNOWLEDGEMENTS

Thanks are due
to my supervisor, for his patience, understanding and
impeccable guidance throughout this research,
to my family, for their continuous love, support and encouragement and
to my friends who have stood by me for the duration of this research.

Thanks is given to
Telkom SA Limited Center of Excellence (CoE)
for the funding they provided for this research.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

A customer in online transactions locates a service that offers the goods or services he/she wants and pays for the goods or services with his/her credit card. The client provides personal information to the service regarding his/her name, address, telephone numbers and even identity numbers. A few days or weeks later, the client receives his goods or services as advertised by the service. The convenience of online shopping is obvious. However, the scenario above is the ideal, best possible outcome. There are many more sinister chains of events that could cause the customer to have a bad experience conducting online shopping: A few weeks after the successful transaction, the client receives spam email; the goods arrive and are not in the condition advertised; the goods do not arrive. These outcomes to the transaction will prohibit customers from entering the online market place for fear that they will be exploited, that they will not be able to control what happens to their personal information, or that they have no guarantee that the goods that they paid for will arrive.

Assurances address these concerns, putting the customer at ease despite these risks existing. Being able to establish trust in a transaction is crucial to having a positive experience online and provides a safety net to customers who shop online. This dissertation focuses on providing the end user with assurances about the transaction and the usage of his/her personal information during and after a transaction in a Web services world. This chapter introduces online transactions, personal information, trust, Web services and highlights how the rest of the dissertation is structured.

1.0.1 Online transactions

Buying goods and procuring services over the Internet has become a viable alternative to buying goods and services in the physical world. Electronic commerce offers several advantages that shopping in the real world cannot. With E-commerce, it is possible to buy goods from different countries and from vendors that cannot or do not supply local stores. E-commerce makes it easy to be more informed about the products or services before paying for them [29], by allowing the user to search and browse through vast amounts of information about specific and related information quickly and easily. Potential customers have at their disposal technical information, user reviews, complaints and opinions, which are not all readily available in the real world.

In the physical world, it is easier to create differences in price of a product due to the lack of

knowledge that a customer generally has regarding the pricing of the identical product elsewhere. This price difference might be a result of legitimate expenses, such as the transportation of the goods to the store or it could be a result of a store owner's knowledge that no other store in the area supplies the same product. In an online world, it is much easier for users to compare prices of identical products from different online stores, and there are many search engines, such as froogle[1] and pricegrabber[2], that allow users to easily search for the store selling a particular product at the best price.

However, E-commerce, while promising an enhanced shopping experience, also has disadvantages when compared to conducting transactions in the real world. Customers who purchase goods online cannot physically inspect the quality of the items before purchasing them and the customer has to pay for the goods before he/she receives them [36]. The customer accepts a great deal of the risk of the transaction in order to facilitate the transaction. In the physical world, these risks are mitigated by the customer's ability to inspect the goods before purchase and being able to walk away with the goods as soon as the items are paid for. Even when the transaction is anonymous to both parties, i.e. the customer has never purchased items from the seller before, the risk involved in the transaction does not become as great as a comparable transaction online, as the customer can inspect the goods, and can identify the company that sold the goods to him/her, allowing the customer to complain should the goods or services be deficient in any way.

1.1 PERSONAL INFORMATION

Another threat that online shopping and transactions expose customers to is the safety of the customer's personal information. In the physical world, the collection of personal information, and personally identifying information (PII) is not as easy as it is in the online world. It is costly to capture and analyze the data and requires significant data-capturing and other time-consuming human involvement. As an illustration, consider a book-store in the physical world compared to the online world. In the online world, the bookstore can collect information on the books a user browsed before ultimately making a purchase. It can then use this information to provide a better service to potential customers by telling them that other people who browsed the same book, ultimately bought another book. This information would be very difficult to collect in a physical book-store. An ethnomethodological study would have to be conducted by trained personnel that could be affect the shopping experience and could be considered an invasion of a customer's privacy.

In the online world, however, the collection, storage and dissemination of personal information and usage information add virtually no cost to the service provider [17]. The infrastructure that provides the service is already able to handle the additional data collection and analysis. The service provider then has the ability to leverage the information that has been collected as a business asset. It is much easier to collect more data from more people than it is possible to do in the physical world. By disseminating and analyzing the data collected from customers and potential customers, the company can tailor the shopping experience that it offers, but it can just as easily sell or exploit the data for commercial gain[17]. The data collected can be about

[1] froogle : http://froogle.google.com
[2] pricegrabber : http://www.pricegrabber.com

the browsing behaviors of customers, geographical information, demographical information and can extend to personally identifiable information, such as information collected in most registration processes.

1.2 TRUST

The ease of establishing an electronic store or service makes the risk of fraud much more likely in the electronic environment than in the physical world [4]. In addition, the lack of sufficient laws governing electronic transactions makes it easier for an unscrupulous service provider to get away with offering sub-standard service.

Due to the greater risks that customers face in Internet transactions, customers should be entitled to greater assurances to offset the increased risks. Basic assurances of the seller or service provider's identity have launched the Internet as a place where business could conducted in a trusting environment. Certificate authorities such as Verisign and Thwate provide assurances to customers on a website that the company is legitimate, enabling customers to confidently enter into a transaction where personal and financial details are disclosed [28]. However, the certificate provided by certificate authorities provides very few details about a company, and is used mainly as a way to pass the public key of a company to a customer to facilitate the setting up of a secure session between the two parties. The certificate does not provide the customer with any assurances other than the basic authentication about the service provider.

Certificates issued by a certificate authority, used by current web browsers, conform to the X.509 v3 certificate standard. This standard is outline in RFC 3280 [21] and is commonly referred to as Public Key Infrastructure X.509 group (PKIX). The certificate contains information on the issuer of the certificate, the time period for which the certificate is valid, the name of the subject of the certificate, its public key and a signature of the certificate that can be validated by re-signing the certificate using the certificate authority's public key.

The information in the certificate gives the customer no more information about a service provider other than its name and domain name, and the public key to which the company has the corresponding private key. The information does not provide the customer with sufficient assurances that his/her transaction will proceed as expected and provides no means by which the customer can relay his/her preferences to the service provider.

This dissertation introduces a mechanism to allow for finer-grained assurances that will assure the client about more than just the identity of a company. It provides the customer with assurances about the proper use of their financial, personal and transactional information. Specifically, this dissertation focuses on the Web services environment, where transactions can be made up of connections between modular software components from various vendors. This dissertation investigates and establishes a measure of trust that can be built to provide the client with the necessary trust in the company that he/she is dealing with.

1.3 WEB SERVICES

Web services is an emerging message oriented middleware architecture that enables companies with disparate computing architectures to communicate effectively. Web Services is based on

platform independent, well formed text messages that enable two companies to convey information in a way that another company can automatically disseminate. Companies can expose their internal systems to external users. These services provide functionality that can be used by other services and a fee may be charged for its use. The modularity of the composition of Web Services offers an unique environment in which different Web services can be grouped together to perform a single business transaction.

In this dissertation, Web services are viewed as modular services, such that many companies may offer the same service to clients or other services. An example of this would be different travel agencies looking up available flight times at a client's request, or many different companies each offering a currency convertion service. In a world where many different services can perform the same functionality, assurances and benefits become an alluring prospect for clients. Clients can choose a particular service because it is less expensive to use, or they can use a service because it offers a better privacy protection policy than any other service that provides similar functionality.

Whilst the ability to compose a single transaction from multiple companies' services in a dynamic environment introduces great opportunities for both service providers and customers, the risks involved in transactions also become more prevalent. It only takes one service with access to the client's personal information to undermine the entire transaction and misuse the information received. No one party involved in the transaction is responsible for the proper use of information and it is therefore up to every service to act in an honest manner and adhere to the customer's request regarding the proper use of his/her personal information.

Web services has received significant attention from both industry and academia. As a result, a plethora of specifications and standards have emerged to address a multitude of scenarios at different levels of the Web services stack. The standards and specifications deal more with the functional aspects of Web services. Specifications regarding the atomicity of distributed transactions exist to control a transaction involving multiple parties at a functional level, but they do not address the privacy concerns that users should have towards the misuse of their personal information. The relevant standards and specifications to this dissertation will be covered in greater detail in Chapter 2. This dissertation provides a specification for the Web Services environment, in which aspects beyond the functional requirements of Web Services can be negotiated so that the customer can be assured that the transaction and usage of his/her personal information proceeds only as agreed upon.

1.4 COMPOSITE WEB SERVICES TRANSACTION CHAINS

Web services can be connected to create a virtual chain of services where each service in the chain adds value or provides a service to enable the successful completion of a business transaction. The service provider with which the client is directly in contact may use other service providers to find information on a particular product, find the cheapest price for a product, purchase a part of machinery or anything that the service provider needs to complete his/her transaction with the client. These services may or may not require remuneration, and payment Web services may be involved in the transaction to facilitate the transaction. These services form part of the chain of services created to process the transaction. Each service has its own

conditions of operation and might require information about the client of the transaction in order to fulfil its service. This service might really need to use this information or might just wish to collect data about users for its own purposes.

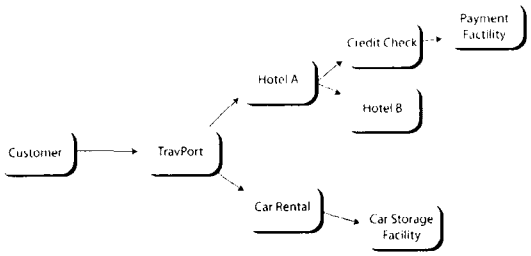

Figure 1.1: A Web Services Transaction Chain example

The problem with the scenario depicted in Figure 1.1 is that the client has little control in what happens in the transaction beyond the service provider that the customer is directly dealing with. In this scenario, the service provider may require to contact and deal with two additional service providers, with each one charging a fee for their part in the transaction. Furthermore, each of the two service providers include credit-checking companies to validate that they will be paid for their services. Provider 1 can deal with either 123 Credit or ABC Credit and Provider 2 only has the ability to deal with ABC Credit. If is was known that ABC Credit was more trustworthy than 123 Credit, it might be in the client's best interests to pay a premium so that Provider 1 would deal with ABC Credit rather than 123 Credit.

The client's preference for a certain company to be involved in the transaction over another spans many different requirements and conditions. The client may want to deal only with companies certified by a certificate authority. The client may prefer to enter into a transaction only with companies that operate within countries that have amicable internet transaction legislation or trading agreements with the customer's country of residence. The reasons for a customer's preference lie beyond the functional requirements of a service and need to be addressed at a higher level than functional requirements.

The client has no say about what companies are involved in the transaction, has no control over what companies have access to some or all of its personal information and has no agreement in place with the third-party service providers regarding the consumption, storage and ownership of his/her personal information. This presents a challenging stumbling block to using Web services for transactions that involve any risk. If a customer knew that other companies were involved in the transaction and that they used his/her personal information in any way, he/she might want assurances that the information would not be sold to an information-harvesting company. In the scenario shown above, the client has no control and cannot specify his/her wishes to all the parties involved in the transaction.

A customer concerned with the protection of his/her personal information may not be willing to engage in a transaction in which the protection and proper use of his/her personal information can be assured.

A protocol to provide the necessary assurances for the customer is required so that both the

customer and the service providers can engage in transactions with the least amount of risk possible. With a reduced risk, a customer will be more willing to engage in a business transaction with the service provider and thus provide the service provider with remuneration for services rendered.

The goals of this dissertation are to investigate the trust assurances that customer's desire in order to confidently enter into online transactions with unknown service providers. The dissertation also presents a protocol designed to streamline the assurance negotiation and agreement processes. This dissertation investigates the trust measures that are available to clients in online transactions, establishes a trust model to use in a trust assurance protocol and creates a simulation environment in which the protocol, and trust calculation can be executed and monitored.

1.5 POTENTIAL APPLICATIONS

Transactions involving multiple partners can be confusing. Add the fact that Web services are modular and complex Web services may be composed of multiple Web services, there is a need to provide the customer in these transaction with the opportunity to become involved in the setup of a transaction. The customer needs to be aware of all the risks facing him/her and should be able to choose the transactions involved in the transaction if two services can offer the same service in the overall transaction. The trust assurance protocol can complement business protocols by first evaluating the entire transaction in terms of the safety issues involved from the end customer's perspective, and then proceed onto the actual business protocol or transaction.

Furthermore, the vocabularies, definitions and documents, including the trust model, could be used to standardise the messages passed between transactions to minimize the problem of ambiguity and to provide a standard base for future trust models and protocols from which to start.

1.6 THESIS OUTLINE

This dissertation investigates the need for trust assurances and establishes the requirements for providing a trust assurance protocol. The Background chapter investigates relevant technologies and academic literature concerning trust in an online environment and looks at the legislation surrounding electronic transactions. A trust model, a protocol to establish trust assurances and a trust measure is developed and presented in Chapter 3. Chapter 4 builds on the trust assurance chapter by implementing the protocols and trust model to create a trust assurance protocol. A transaction execution framework is also presented to execute the protocol so that the protocol may be evaluated. The vocabularies used in this dissertation are also discussed in this chapter. Chapter 5 discusses the evaluation of the trust assurance protocol and trust model and presents the results of this dissertation. Lastly, the conclusion provides a summary of the work presented in this dissertation.

CHAPTER 2

BACKGROUND

2.1 INTRODUCTION

Web services is an emerging extensible messaging framework in which messages can be sent between two network interfaces across any network. In contrast to previous middle-ware messaging systems, Web services provides a light-weight framework that uses well-formed text messages to transmit data and instructions across network boundaries and between disparate computer systems. Web services is based on existing standards to lower the barriers to adoption and increase the rate and ease of adoption. Web services messages are written in SOAP which is itself described in the eXtended Markup Language (XML). These and other base standards are discussed later in this chapter. In a similar manner, many other standards are based on the core Web services standards to provide specific functionality to Web services. This allows functionality to be added to Web services in an evolutionary manner, and allows new interchange methods to be developed as they become desirable.

The ease of using Web services has allowed companies, sole proprietors and even individuals to create and publish Web services which can be found and used over the Internet. However, the ease of creating a Web service means that unscrupulous individuals can also easily create Web services that look no different from any other Web service. Transactions taking place between two computer agents now carry real risks for the owners of companies and involve real money [4].

The capabilities created by the flexibility of the Web services technology provide both opportunities and problems. Companies are able to create and publish small autonomous services that interact directly with customers and with other Web services. These services could be small and modular, allowing customers and other services to make use of the service for a fee or without charge. If a charge is levied for the use of a service, the service could in turn use the Web service of a merchant bank to process the payment. The modularity of the Web services environment allows for exciting new business models, where hundreds of services from different companies around the world could offer the same service to a customer anywhere in the world [46]. The problems that arise from this scenario is the risk of choosing the right services for the transaction. If there are hundreds of services offering the same service, it is likely that some services could be less trustworthy than others. Because there is little to no barrier to entry to create a

Web service and it is difficult or nearly impossible to distinguish a good service from a bad one just by appearance as every Web service message will look exactly the same, regardless of the integrity of the company or individual providing the service. How can a customer of these services be confident that the transaction will proceed as advertised, and that his/her personal information will not be misused?

In studying the problem being addressed in this dissertation, it is clear that there are many different fields of knowledge involved in the problem. Not only are there technical issues involved, but sociological and legislative issues are also tightly integrated into the problem. This chapter serves as a primer for the work presented in this dissertation, from the foundations of XML to trust models and current legislation about electronic transactions.

2.2 EXTENSIBLE MARKUP LANGUAGE (XML)

The eXtensible Markup Language (XML) [54] is a text format that uses tags to syntactically qualify text in a document. It is derived from the Standard Generalized Markup Language (SGML) [25]. XML was submitted to a standards body, the World Wide Web Consortium (W3C), in November 1996 at the SGML Conference held in Boston. XML is a language that allows any hierarchical structure to be easily stored in a text format that consists of using tags to give structure to the text. It is generally human-readable because tags usually have descriptive names and the document is stored and transmitted in plain text. XML does not provide a dictionary of set tags that must be used, such as the restrictions of the HyperText Markup Language (HTML). Instead, XML allows users to define their own tags, but users are then responsible for creating their own applications that are capable of consuming the XML messages. A simple XML document is shown below:

```
<Fruit_Basket>
  <fruit>
    <name>Orange</name>
    <colour>Orange</colour>
  </fruit>
  <fruit>
    <name>Banana</name>
  </fruit>
  <fruit>
    <name>Apple</name>
    <colour>Green</colour>
  </fruit>
  <fruit>
    <name>Apple</name>
    <colour>Red</colour>
  </fruit>
</Fruit_Basket>
```

XML is used as a convenient standard to extend and create structured messages and documents. It has become a defacto standard in emerging standards and specifications regarding the World

Wide Web [11]. Standards and specifications built upon XML utilize the language's expressive capabilities to easily form new document types that can be created and consumed on any computing platform. XML parsers allow for efficient and consistent creation and dissemination of XML documents and can be included in other applications, allowing developers to use existing parsers and libraries to speed up the development and increase the quality of their own applications.

Because XML allows users to create their own tags, ambiguity can creep into XML documents, as the meaning of tags is not always clear. As an example, the tag <desktop> could both mean a physical top of a desk, or the virtual desktop of a graphical user interface. XML namespaces were created to deal with this ambiguity.

2.2.1 XML namespaces

XML namespaces is a standard created by the W3C to provide XML with the ability to create unique identifiers and to associate tags with these identifiers. If a tag is given a namespace, then the tag can be distinguished from other tags with the same name. A namespace is a reference to a Uniform Resource Identifier (URI). This URI is usually not read by a parser, it is merely used as a string to identify a particular XML element in an XML document. URI's have been used due their uniqueness, as opposed to a descriptive name, such as "myOwnNamespace". It is common practice to make the namespace URI more useful than just a unique string identifier. The URI could point to a DTD or XML Schema which defines the data structure governing the use of the XML elements in the specific namespace.

2.2.2 Document Type Definitions and XML Schema

A Document Type Definition (DTD) is a file containing rules that define how an XML document has to be structured in order to be a valid XML document for that particular DTD. It is defined in the HTML 4.01 specification[38]. The following is a very simple example of a DTD document:

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT Fruit_Basket (fruit)*>
<!ELEMENT fruit (name, colour?)>
<!ELEMENT name(#PCDATA) >
<!ELEMENT colour (#PCDATA) >
```

The DTD defines Fruit_Basket as the root element of the DTD. This means that any XML document based on the DTD must begin with a Fruit_Basket element. It states that the only element a Fruit_Basket can contain is fruit element and indicates that any number of fruit elements can be contained by the Fruit_Basket element, by virtue of the "*" operator. The fruit element is defined as a element consisting of the name and colour elements. The colour element is optional due to the "?" operator. Both the name element and the colour element is defined to be of the type "#PCDATA", which stands for printable character data. This means that the name and colour elements contain text and not more elements.

An example XML document that is based on the DTD described above is shown below:

```
<Fruit_Basket>
  <fruit>
    <name>Orange</name>
    <colour>Orange</colour>
  </fruit>
  <fruit>
    <name>Banana</name>
  </fruit>
  <fruit>
    <name>Apple</name>
    <colour>Green</colour>
  </fruit>
  <fruit>
    <name>Apple</name>
    <colour>Red</colour>
  </fruit>
</Fruit_Basket>
```

DTDs allows users of XML data to ensure that their XML messages are well formed and that they will be compatible with applications using the DTD. DTDs, however, have become more or less obsolete due to limitations in the description language. DTDs cannot express namespaces, and as described above, have become an integral part of the XML language. DTDs have also been ecplised by a new definition language, namely XML schema [53]. The definition of XML Schemas by the W3C is that XML Schemas "... express shared vocabularies and allow machines to carry out rules made by people. They provide a means for defining the structure, content and semantics of XML documents." XML schemas are valid XML documents themselves so that they can be parsed by XML parsers without any additional work requiring to be done to specify how an XML document has to be formed in order to comply with the schema definition.


```
<?xml version="1.0" encoding="UTF-8"?> <xs:schema
elementFormDefault="qualified" attributeFormDefault="unqualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="Fruit_Basket">
        <xs:annotation>
            <xs:documentation>Comment describing your root element</xs:dc
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element name="Fruit" minOccurs="0" maxOccurs="unbounc
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="name" type="xs:string"/>
                            <xs:element name="colour" type="xs:string" mi
                        </xs:sequence>
                    </xs:complexType>
```

```
            </xs:element>
        </xs:sequence>
    </xs:complexType>
</xs:element>
</xs:schema>
```

This XML file described earlier will also be valid for the schema presented above.

2.2.3 **XML in this dissertation**

XML is currently the *defacto* standard of formatting complex information so that it can be transported between two endpoints over the Internet. In this dissertation, XML will be used to mark up all messages passed between two communicating parties. In part, this is a requirement of the underlying communication mechanism, SOAP, but it is also a convenient way of displaying and storing information.

2.3 SOAP

SOAP[19] is a lightweight XML specification to specify messages independently of transport mechanisms and protocols. SOAP is independent of any programming architecture or platform, allowing different platforms the capability to create and consume SOAP messages created in disparate environments.

SOAP, originally an acronym for "Simple Object Access Protocol" was developed by Dave Winer, Don Box, Bob Atkinson and Mohsen Al-Ghosein in 1998. The purpose of SOAP was to allow access to objects on disparate connected computing platforms. The current version of the SOAP protocol, version 1.2, has dropped the meaning of the original acronym due to the belief that the acronym is confusing[19].

The SOAP protocol is a structured message format in which messages can be sent from one computing service to another. SOAP provides the mechanisms to format a message and attach header information containing information about the message itself. SOAP does not place restrictions on the protocols or the message contents that it transports, making it a light-weight, highly agile and protocol-neutral messaging system. Messaging paradigms, such as asynchronous one-way messages and synchronous request/response type messaging, can easily be created by passing single messages in a predefined way between parties who both know the manner in which multiple SOAP messages can create more complex messaging paradigms in order to achieve a higher level transport protocol. SOAP itself does not provide different messaging mechanisms, but rather only formats an individual message. SOAP is designed to allow other protocols or protocol extensions to support different messaging mechanisms. As an illustration, consider two popular messaging systems, the send-and-forget mechanism, where a message will be sent and the sender will not wait for a response. The synchronous send-and-receive mechanism forces the sender to halt its computations to wait for a return message from the responder. The SOAP message in these two instances could be identical. There is nothing inherent in a SOAP message to state how a messaging system should work. By avoiding complex details about the messaging frameworks and transport mechanisms out of SOAP, the standard is simple

to use, but can be extended to accommodate more complex interactions[19].

SOAP Message Security

The SOAP protocol is independent of any transport mechanism, leaving it up to a transport protocol to define the intricacies of transporting SOAP messages. This is why transport level security is not necessarily enough to protect SOAP messages. Currently, it is common to transport SOAP messages using the Transmission Control Protocol (TCP) over the internet, but SOAP can just as easily be sent over the Simple Mail Transfer Protocol (SMTP). The elegance of SOAP is that nothing in the SOAP message has to be changed to be sent over different transport protocols, and can thus switch between different protocols seamlessly, allowing greater proliferation of SOAP messages. By using security capabilities of the transport protocol, SOAP becomes tied to a particular transport protocol, and has to undergo a message transformation at a network boundary in order to be transmitted over a different transport mechanism or protocol. This may compromise the secured message or might not even be possible if the network boundary interface does not support message level processing or the encryption techniques required to transform the message.

SOAP Messages

SOAP messages consists of a SOAP envelope, an optional header and a body.

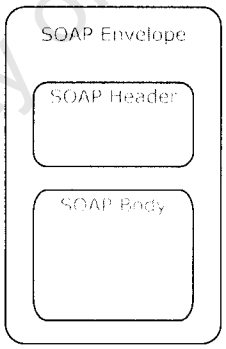

Figure 2.1: A graphical representation of a SOAP message

```
<?xml version='1.0' ?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
        <env:Header>
        </env:Header>
        <env:Body>
        </env:Body>
</env:Envelope >
```

Figure 2.2: A skeleton XML representation of a SOAP message

Figures 2.1 and 2.2 above are both representations of a SOAP message. Figure 2.1 shows conceptually how a SOAP message's header and body elements are contained within the SOAP envelope. Figure 2.2 shows the nested elements within the SOAP envelope defined by the envelope namespace.

2.4 RELATED WEB SERVICES STANDARDS

In addition to the fundamental building blocks provided by XML and SOAP, more standards are needed to make Web services accessible and discoverable. The Web Services Description Language (WSDL) [50] and the Universal Description, Discovery, and Integration (UDDI) standard [8].

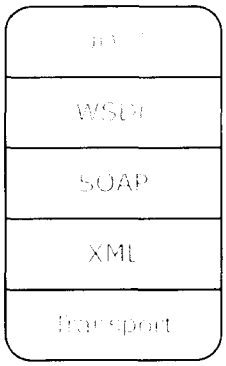

Figure 2.3: The standards forming the basis of Web Services

In Figure 2.3 the basic stack of standards and protocols is shown. Each level of the stack builds onto the level below it to provide additional functionality to that provided in the previous level. The transport level consists of the protocol that enables messages to be transported over physical network protocols, e.g. HTTP, FTP, SMTP, and any other protocol that allows the transmission of text over the communication channel. XML provides the format for all the SOAP messages that enable the message to be split into its body information and its headers so that the message can be effectively routed and delivered to the correct party. WSDL describes a Web service and how to use it. A UDDI registry can then be used to locate the Web service that meets the service requirements. WSDL and UDDI are described below.

2.4.1 WSDL

The Web Services Description Language (WSDL) is a standard that allows the functional parameters of a Web service to be described so that users may know how to invoke the service[13]. According to the standard, "WSDL is an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information." The WSDL document thus lets clients know how they can engage in a Web services transaction with a Web service defined in the WSDL document and also which parameters the Web service endpoint requires. WSDL is only concerned with providing details about the functional requirements of a Web service, and cannot express more complex business policies or requirements. WS-PolicyAttachment [23] is a specification that describes how a WS-Policy document can be accessed from a WSDL document.

2.4.2 UDDI

The Universal Description, Discovery, and Integration (UDDI) standard was created to create the formation of a registry where services could be registered so that users could find a particular

Security	WS-Security WS-SecureConversation WS-Trust WS-Policy WS-Privacy
Transactions	WS-Transaction WS-BPEL WS-AtomicTransaction WS-Coordination WS-Discovery
Choreography	WS-CDL

Table 2.1: Table of Web Services standards and specifications

service based on the functionality advertised by a Web Service. The UDDI standard is endorsed by the Organization for the Advancement of Structured Information Standards (OASIS).

UDDI registries can be accessed by SOAP messages to find suitable services to interact with. Without a registry where service providers could list their services and what they do, it would be practically impossible to locate a previously unknown service. The main function of UDDI registries is to provide the location of a service as well the location of a particular service's WSDL document to allow SOAP messages to be created to that the user may interact with the Web Service.

2.5 WEB SERVICES STANDARDS

The extensible nature of the Web Service's architecture has caused an explosion of Web Services standards and specifications, as standards bodies and companies attempt to leverage Web Services. Specifications are created to allow new functionality for Web Services, and these specifications are then submitted to standardization bodies, such as the W3C or OASIS, to be standardized. Standardization is necessary so that different implementations of Web services engines can engage each other seamlessly and thus fulfil the objective that Web services were created to achieve.

The recent hive of activity has created a mass of standards and specifications in many different spheres of Web Services. Whilst there are many standards and even more specifications, only some are relevant to this thesis. The main Web services standards and specifications are listed below and the most relevant standard are investigated in greater detail. Table 2.1 shows the notable and relevant standards and specifications:

2.5.1 WS-Security

The WS-Security standard [35], adopted by OASIS and first released as a standard in 2004, represents a major component for almost any other Web services standard or specification dealing with security. Currently at version 1.1 as of February 2006, the specification continues to form the foundation of security standards around Web services. The WS-Security standard does not

propose any security protocols, and does not enforce a particular encryption protocol or signature hashing algorithm, but rather provides a framework within which messages or parts of messages can be secured with any security mechanisms. As such, WS-Security does not provide a all-encompassing security product, but rather allows other protocols and even application logic to allow the creation of more advanced security features.

The WS-Security standard is itself a collection of standards:

- WS-Security Core Specification
- Username Token Profile
- X.509 Token Profile
- SAML Token Profile
- Kerberos Token Profile
- Rights Expression Language (REL) Token Profile
- SOAP with Attachments Profile

The WS-Security Core Specification is a standard that introduces extensions to SOAP 1, 1.1 and 1.2 to allow for message content integrity and also provide mechanisms for confidentiality. The core standard allows multiple types of token to be attached to a SOAP message, but doesn't elaborate on what these tokens are. The profiles that form part of the WS-Security standard provide extensions to the core specification and allows the use of X.509 certificates, a Security Assertion Markup Language (SAML) token, Kerberos Tokens and REL tokens to be attached to SOAP messages via the WS-Security standard[43]. The aim of the core specification is to provide three enhancements to SOAP messages: To provide message integrity mechanisms, ensure message confidentiality and the ability to attach security tokens to a message.

The Username token profile extends the core specification by providing mechanisms to allow consumers to authenticate themselves to a Web service producer by a username and a password. The X.509 certificate token profile allows an X.509 certificate to be added to a Web service message so that a Web services producer or consumer can be authenticated. X.509 certificate is explained later in this chapter. Similarly, the other token profiles extends the ability to attach the relevant tokens to a SOAP message. The SOAP with Attachments (SWA) profile extends goals of the WS-Security Core specification to non-XML data that is sent in a SOAP message so that its integrity is maintained and it can be confidentially transmitted between two parties.

2.5.2 **WS-Trust**

WS-Trust is a specification that extends the functionality provided by WS-Security[22]. The specification extends the type of security tokens that can be used with SOAP messages secured with WS-Security. Additionally, WS-Trust provides a mechanism to exchange security tokens between two parties. In particular, WS-Trust is concerned with the following:

- methods of issuing security tokens
- methods of security renewing tokens
- methods of validating security tokens
- ways to establish trust relationships items methods to assess the presence of trust between two parties
- methods to broker trust relationships.

According to the specification, "The goal of WS-Trust is to enable applications to construct trusted [SOAP] message exchanges." In other words, if a Web service receives a message purporting to be from a business partner, how can the two parties authenticate themselves and establish a trust relationship?

As with other building-block specifications, WS-Trust does not provide concrete protocols and thus does not restrict Web services to conforming to a particular mechanism. However, all this abstraction still leaves the entire transaction landscape without a definite manner in how trust relationships should be brokered. WS-Trust explicitly does not provide support for password authentication, token revocation, management trust policies and other goals such as key derivation between the different parties.

The specification addresses three key issues:

- requesting and obtaining security tokens
- managing trusts and establishing trust relationships
- establishing and assessing trust relationships

2.5.3 **WS-Policy**

According to the WS-Policy specification document, "WS-Policy provides a flexible and extensible grammar for expressing the capabilities, requirements, and general characteristics of entities in an XML Web services-based system. WS-Policy defines a framework and a model for the expression of these properties as policies." [44]

WS-Policy presents a framework in which statements forming a policy can be stated and attached to a SOAP message. Primarily developed to deal with security issues of SOAP messages, WS-Policy can be used as a general framework in which to describe the policies of a particular Web service or Web services client.

Similar to the notion of modular and open Web services standards, WS-Policy consists of a number of specifications, each adding functionality to the general framework of the policy specification.

The WS-Policy framework places a great deal of emphasis on the need for the ability to state the alternatives for a choice, so that the other party involved can choose the most preferred option.

2.5.4 **WS-Transaction**

The WS-Transaction specification specifies two coordination mechanisms that extends the capability of the WS-Coordination specification [24]. The two mechanisms that are defined are Atomic Transaction and Business Activity. These coordination schemes allow complex business tasks to be conducted in a manner which all parties involved understand how the transaction will occur. Atomic transaction represents the entire transaction as an atomic event - either the entire transaction succeeds or nothing does. If the last step in a transaction fails, then the entire transaction fails and nothing is committed. If the transaction succeeds, then and only then is the transaction committed and stored as a successful transaction.

According to the WS-Transaction specification, "The Business Activity protocols handle long-lived activities and the desire to apply business logic to handle business exceptions." [24]

Business Activity represents transactions that need more flexibility and complexity than Atomic transactions can represent. Business activity allows business processes to be modelled and conveyed to all parties involved in the transaction. Business Activity coordinations can include atomic transactions. If a business activity fails, it is important to use business logic to determine the failure and to store the transaction for reasons beyond the scope of the individual business activity. Financial, regulatory and customer relationship reasons all need to be addressed when long-running and resource-intensive business transactions fail.

2.5.5 **WS-CDL**

WS-CDL, also known as the Web Services Choreography Description Language, is a specification that "describes peer-to-peer collarborations of participants by defining, from a global viewpoint, their common and complementary observable behavior; where ordered message exchanges result in accomplishing a common business goal." [51]

WS-CDL is a specification for specifying how different parties interact in a particular transaction, looking at the complete transaction between all the parties from a global viewpoint. The WS-CDL specification does not specify how those parties find each other before the choreography can be specified. The WS-CDL specification is designed so that there is not a single point of control for the transaction. All the services work autonomously from one another and only pass messages between each other. WS-CDL is considered with ordering rules for the messages thus providing a sequence of messages that have to be passed in order that the WS-CDL is adhered to. WS-CDL is not a executable langue and as such cannot be run to automatically run the choreography defined in the document. A choreography is rather a description of the processes that have to take place in a particular order so that all the parties involved in the transaction are aware of their role in the transaction [51].

2.6 TRUST

Trust plays an important part in almost every activity that people do. They have trust in the people they are dealing with, or the object they interact with. At a subconscious level, trust fulfils a vital role in society and personal safety. The trust required to use an elevator is real, although it is not easy to define and is difficult, even impossible, to calculate. The roots of

trust lie in biological and sociological trust and permeates every situation that people encounter. We trust the elevator not to plunge down the elevator shaft uncontrollably, we trust that the button we push will take us to the desired level in the building and we trust that the door will open when the elevator stops. Without implicit trust in the ability and intentions of people or objects, society would not function because no-one would put themselves in a vulnerable position required to make use of someone else's services[17].

When a level of abstraction, such as computers, is introduced, the meaning and derivation of trust becomes more problematic. This is because there is the trust between two computers or computing domains, and the trust between people interacting with each other via the computer or a person dealing directly with a computer. It also removes the experience and instincts that people have about a situation or a person. In a world where only well-formed text messages are sent and received, it becomes very difficult to collect any implicit trust indicators.

In the physical world, people can intrinsically evaluate a person, situation, or company based on previous experiences and instincts that are difficult to enumerate. In a computing world using Web services, the messages created and transmitted from a malicious company or person do not provide any intrinsic warning signs; it is just transmitted as messages that look exactly like any other message. Whilst this is not true for the semantic meaning of the message, it holds for the syntactic message format and therefore does not allow computers to consume the data and evaluate the trustworthiness of a message.

Researchers from all different fields of academia have tried to define trust so that the concept of trust can be elegantly modelled and included into scenarios. The concept of trust is non-trivial because it is not possible to provide one unique definition of trust [33]. The reasons for this is that trust is a concept integral to almost every sphere where two or more people interact, or where a person interacts with an inanimate object. Trust is so deeply rooted in our sociological and biological being that we intrinsically know when to trust or not to trust. With the advent of a technology where people can engage in faceless transactions across borders of control, it is desirable to develop a well-formed definition of trust and how it can be computed. The following section briefly looks at different definitions of trust that have appeared in literature.

2.6.1 Definition of trust

In academic literature, especially in computer science, trust has been synonymous with terms such as "confidence", "reputation" and "benevolence". Whilst in certain conditions that may hold, it is clear that trust is a complex term to grasp. In this dissertation, a definition of trust is required which is capable of being measured and calculated for a specific purpose. Although any trust definition with these characteristics has to be a simplified, restricted view of trust, a computational model must exist so that it can be determined if a transaction should proceed or not.

Diego Gambetta's definition[18] has been used and cited in numerous papers regarding trust, and is prudent to examine here. Gambetta's states that, "trust (or, symmetrically, distrust) is a particular level of the subjective probability with which an agent assesses that another agent or group of agents will perform a particular action, both before he can monitor such action (or independently of his capacity ever to be able to monitor it) and in a context in which it affects

his own action." Gambetta asserts that being "trustworthy" implies that the *probability* that the trustee will honour his claims towards us, or in the absence of explicit claims, will act in a manner that is not detrimental towards the trustor, so that cooperation between the parties can take place. Conversely, being "untrustworthy" refers a low probability of the trustee acting in a favourable manner towards the trustor.

Gambetta defines the measure of trust as a threshold point rather than an absolute measure of trust of distrust. In Gambetta's research, a trust level of 0 is considered to to be compete distrust, and a value of 1 to be complete trust. Logically it follows that the mid-point, 0.5, refers to uncertainty in the scale. Other researchers have developed definitions that contain discrete values for different levels of trust and thus do not add the complexity of a continuum to their trust models. In [3], Abdul-Rahman and Hailes introduce the notion of trust degrees as opposed to a trust continuum. They introduce four distinct degrees of trustworthiness, namely:

- very trustworthy
- trustworthy
- untrustworthy
- very untrustworthy

Adbul-Rahman and Hailes use Gambetta's definition of trust but utilize the discrete values for trust in order to simplify the determination of trust in their definition.

Apart from the initial trust determination, previous interaction is vitally important to a trust rating. Gambetta calls this a subjective expectation[18] that the trustor has about the trustee's future behavior based on the transaction that has taken place between the two parties. Historical information, personal experience as well as recommendations from other trusted parties or even unknown parties affect many different trust schemes. eBay's feedback system ultimately determines a seller's trustworthiness by showing potential buyers the comments and ratings made by other buyers who have dealt with the seller previously. In [5], Bajari and Hortacsu showed how a seller's reputation in the online auction site, eBay, influences the price received on a bid item quite significantly. A seller perceived to be more trustworthy gets paid a premium for being trustworthy.

Bhattacharya *et al* [9] states that "Trust is an expectancy of positive (or nonnegative) outcomes that one can receive based on the expected action of another party in an interaction characterized by uncertainty." Uncertainty refers to the risk inherent in depending on someone else to cooperate towards the desired goal without knowing what their motives are or if the other party is able to produce the desired result. Risk is inherent in any situation where trust occurs.

Mayer, Davis and Schoorman [30] define trust as the "willingness of a party to be vulnerable to the actions of another party based on the expectation that the other party will perform a particular action important to the trustor, irrespective of the ability to monitor or control that other party." The definition effectively associates a level of risk to any situation requiring trust

because of the position of vulnerability that the trustor places himself in to allow the trustee to perform the desired action.

2.6.2 **Risk**

In [4], Jøsang *et al* state that it is the consumer who is forced to accept the risk in a transaction conducted in an online environment, by paying for a product before seeing or testing it. The consumer has to rely on the trustworthiness of the service provider and thus there exists a significant advantage if a service provider is perceived to be more trustworthy than another service provider.

It is therefore advantageous for both the customer and the (trustworthy) service provider to be able to establish some evidence of trustworthiness to facilitate the transaction.

Mayer *et al* [30] extends the definition of risk and how it affects trust. They claim that there is no risk in trusting a person, but there is risk in engaging in a trusting action with that person. The act of trusting a person or situation does not involve assuming risk upon oneself; it is when you trust the trusted party to assume a position which may have negative repercussions to yourself, that the risk is taken. Trust is a necessary component of being willing to assume the risk of relying on another party. In this dissertation, the customer places him/herself in a vulnerable position by allowing many service providers to act on his/her behalf to complete a transaction. The aim of the dissertation is to minimize the risk of such a transaction by providing the customer with sufficient information about the transaction and the services involved in the transaction to make an informed choice about the risks that he/she faces in the transaction.

2.6.3 **Trust models**

In the literature surrounding trust, many different types of trust have been mentioned. Each type of trust defines trust for a specific context. The list of trust definitions below is not exhaustive, but covers the major trust definitions appearing in trust literature, ranging from sociological trust to organizational trust to trust between computer systems:

- reliability trust
- decision trust
- direct trust
- recommender trust
- weak-form trust
- semi-strong trust
- strong trust
- strategic trust
- moralistic trust

- transitive trust
- social control trust
- reciprocity as trust
- decentralized trust
- personal trust.

Reliability trust

Reliability trust is simply a term used to describe the reliance of one party on another to perform an action on his/her behalf. It is a common definition of trust and is defined by Gambetta as the "subjective probability" that one party will perform an action on another's behalf. [18]

Decision trust

Another common type of trust is called decision trust. Decision trust is defined as the as the extent to which a trustor will trust an unspecified trustee because the possible positive outcomes outweigh the risks involved in the transaction [18]. Effectively, if the possible benefits or necessity of a transaction outweigh the risks involved in the transaction, then the trustor will initiate the transaction with the unknown trustee and will willingly assume the risk.

Direct trust

Direct trust refers to trust between two parties where the trustor does not need to rely on any external information to calculate the trustworthiness of the trustee. In this case, the trustor knows everything necessary about the trustee in order to evaluate his/her suitability for the task to be completed. In [3], Abdul-Rahman and Hailes use their discrete values of trust to assign levels of direct trust.

Figure 2.4: Direct trust between client and service

In Figure 2.4, direct trust is shown. In this scenario, the trustor has previous knowledge of the trustee and has enough information at his/her disposal to make an informed choice about whether or not he can trust the trustee to perform the desired task in a trustworthy manner.

Recommender trust

Recommender trust refers to trust between a trustor and trustee where the trustor uses external parties to determine whether or not a trustee is trustworthy for a particular transaction. Recommender trust is quite a broad topic, as the external resources can be trusted third parties, such as certificate authorities, or can consist of a society of electronic agents that propagate previous interaction information throughout the society so that a measure of trustworthiness can be

determined. Abdul-Rahman and Hailes define recommender trust as trust derived from word-of-mouth. They also caution that trusted authorities are not good enough by themselves because a trusted authority's authority over a community diminishes as more agents join the community [2].

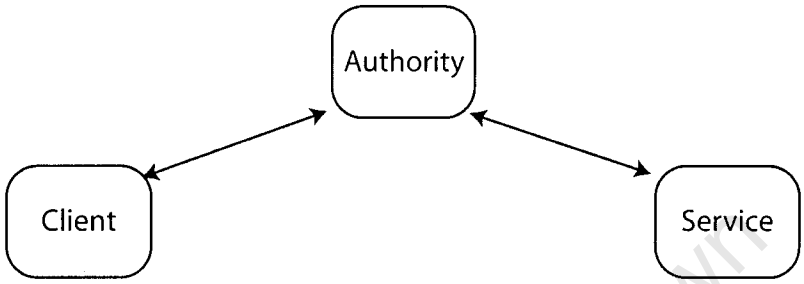

Figure 2.5: Indirect trust between client and service

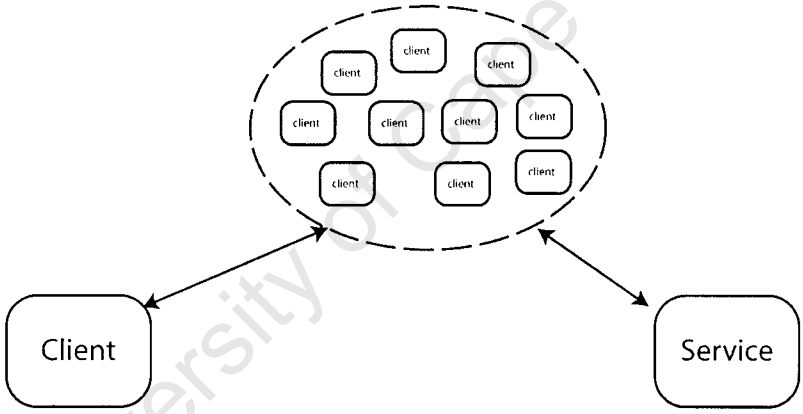

Figure 2.6: Reputation feedback mechanism that reports historical performance of a service

Figures 2.5 and 2.6 show the different recommender trusts that can be utilized to determine the trustee's trustworthiness.

Strategic trust

In [49], Uslaner defines strategic trust to be a trust that is developed through experience with different people engaged in a specific transaction or act. Strategic trust is specifically defined for the online environment. Uslaner states that strategic trust is the concern about our safety online: whether the site is secure, whether our personal information is stored in a correct manner and a myriad of other conscious decisions about the current situation against previous experiences of online transactions and knowledge about the potential dangers of conducting transactions in an online environment.

Moralistic trust

In [49], Uslaner defines moralistic trust as trust that is developed individually at an early age and is independent of experiences of the Internet and online e-commerce. Uslaner claims it's

generally an optimistic trust that enables people to take the initial risks online in the first place, before they develop any strategic trust over time by gaining experience in the online world.

Although his paper emphasizes the importance of moralistic trust, such a trust cannot be measured or calculated with respect to individual online business transactions. In the trust model elaborated on in this chapter, moralistic trust effectively becomes the parameter *propensity to trust* to indicate its global influence on an individual's ability to trust a Web service or Internet company in the absence of previous knowledge or experience.

Transitive trust

Transitive trust can be considered a type of recommender trust where the trustor takes the trust value that the recommending party has of the trustee as his own[4]. As an illustration, if the trustor completely trusts the recommending party, Charlie, and Charlie completely trusts (distrusts) the trustee, then the trustor will completely trust (distrust) the trustee. The transitivity of the trust value enables quick calculations of the trust value. In [27], Jøsang disagrees that trust can be modelled via transitivity, which led to a more structured view of transitivity in trust, namely, conditional transitivity of trust[2]. Conditional transitivity states four conditions for the transitivity of trust. The conditions are listed below:

1. The trustor receives a trust value from a recommender as a *recommendation* about the trustee.
2. The trustor must trust the recommender to recommend only suitable parties.
3. The trustor is allowed to make judgements over the quality of the recommendation received from the recommender.
4. Trust is not absolute. The trustor does not have to adopt the level of trust that the recommender has in the trustee.

So, if a trustor trusts Charlie and Charlie trusts the trustee completely, then the trustor will trust the trustee, but not necessarily as much as Charlie does. Once again, the trust level is dependent upon the risks involved in the transaction and how much the trustor trusts Charlie to provide an accurate recommendation for the context of the transaction.

Social control trust

Social control refers to a feedback mechanism that is used to propagate information about a transaction to the rest of the potential users of a service in a community [3, 45]. It is an essential component of the trust mechanism that is shown in Figure 2.6. Social control is known as a soft security mechanism, and as such eliminates the requirements for a global authority [39]. Social control induces cooperation because any transgression of the part of a service provider or trustee will be propagated throughout the entire domain of the social control. Thus, social control can be a self-fulfilling prophecy as any behavior that deviates from the norm in the social control domain, will be viewed as undermining the social control and will reflect negatively on the perpetrator of a transgression.

In [34], Mui and Halberstadt present a reputation model that includes permeating a reputation through a community of cooperating agents. Reporting a company's or person's bad reputation in a system proves to be pivotal to all the agents in the community wishing to conduct legitimate business transactions.[12]

Reciprocity as trust

Reciprocity is defined as "a mutual exchange of privileges" in the Mirriam-Webster English dictionary. Reciprocal actions will create a greater sense of trust between two parties. A transgression or lack of reciprocal action will reduce trust between two parties. This level of trust based on reciprocal actions can be propagated through a community using a social control mechanism and can be used as a trust rating. Reciprocity has been studied by evolutionary biologists[34], and explain reciprocity as a possible explanation for cooperation.

In [34], Mui *et al* uses reciprocity to expand on the concept of strategic trust. They use reciprocity between two parties as a major component in their trust and reputation model. Benevolence is a similar concept to reciprocity when used in a trust calculation, as a high reciprocity will make it more appealing for a customer to engage in an transaction with a particular company. The difference between the definition of reciprocity and benevolence is that the factors that make up reciprocity are split between the benevolence and integrity factors in the trust model used in this dissertation. Whilst a reciprocity value is permeated through the entire system and is used to calculate a possible trust or reputation value, benevolence is an individual factor that is not distributed throughout the network in which the transaction takes place. If a customer is unhappy with a transaction in which it was involved, he/she will reduce his/her benevolence rating towards the company in question and will also, if deemed necessary, permeate information about the failed transaction or the company's transgression so that the perpetrator's integrity factor may be reduced.

This separation from an individual's dislike of a company therefore has less of an impact on the company, whilst still reducing the company's total rating for the individual involved in the original transaction such that he will not engage in further transactions, or be more cautious in doing so. The separation also makes it more unlikely that an unfair attack on a company can influence the company's reputation too much. If a company makes an error or a mistake in a transaction involving an unforgiving customer, the customer might never use that service again. However, the negative rating the customer reports will not influence the overall integrity rating greatly and thus will not skew everyone's perception of the company.

Decentralized trust

Decentralized trust refers to a trust mechanism where each trustor makes his or her own decisions[2, 10] independently of any control authority or community. It was introduced to reduce ambiguity and complexity of having to propagate a trustor's policies throughout a community. This enables simplicity in the trust model, because it allows the trustor to make use of any information he/she can obtain, including ratings provided by trust authorities and community recommendations, and then formulate its own decision about the trustworthiness of a trustee to suit his/her personal preferences that are applicable to the specific context and risk in which a transaction takes place.

Weak-form trust

Barney and Hansen [6] defined weak-form trust as trust that relates "relationships and interactions where the parties involved are not vulnerable to the actions of the other parties, and thus no significant precautions are required."

Weak form trust thus requires less assurances and a smaller level of trust in order for a trustor to engage in the relationship with the trustee as there is little risk in the scenario.

Semi-strong trust

Barney and Hansen [6] went on to define semi-strong trust as trust where one party or both parties are vulnerable to each other's actions. This introduces more risk into a situation and therefore requires more trust assurance between the parties. The authors then proceed to state that governing mechanisms are needed to facilitate the transaction.

Strong trust

Barney and Hansen [6] proceeded to define a third type of trust, namely strong trust. Strong trust refers to the trust required in a situation where one party is "significantly" vulnerable to the other party, because malicious behavior on the part of the trustee would be advantageous to himself, despite the negative repercussions of the trustee's actions. The authors state that governing mechanisms and social and economic controls do not influence the trustee because the incentive of cheating is greater than cooperating.

Personal trust

Williamson [52] defined personal trust as trust that "applies to emotional and personal interactions such as love relationships where mutual performance is not always monitored and where failures are forgiven rather than sanctioned."

Williamson believed that modelling trust was not suitable for economic interaction and that more well-defined concepts, such as reliability, utility and risk, can be used instead of trust in such interactions[4]. In this dissertation, personal trust in not suitable as a trust model due to the nature of the composition of transactions. Multiple service providers can be used for a single transaction, where many, if not all, service providers are unknown to the trustor, which eliminates the forgiving and loving relationship that defines "personal trust."

2.6.4 **Trust in this dissertation**

In the previous sections, models of trust in academia was discussed to to show the diversity of what trust can mean. It is important to note that these trust models aren't necessarily exclusive of one another. Many are complementary and can be combined to create new specialized models. To establish a suitable trust model for use in this dissertation, the authors propose using various models and allowing the consumer, or trustor, the freedom to use the model that best suits him for a particular scenario. By allowing for a trust suitable to the situation, and allowing the trustor to establish the correct type of trust to provide just enough assurance, so as to avoid unnecessarily costly calculation and investigation if it is not required or desired. In the online

Proposition	
1	The higher the trustor's propensity to trust, the higher the trust for a trustee prior to availability of information about the trustee.
2	Trust for a trustee will be a function of the trustee's perceived ability, benevolence and integrity and of the trustor's propensity to trust.
3	The effect of integrity will be most salient early in the relationship prior to the development of meaningful benevolence data.
4	The effect of perceived benevolence on trust will increase over time as the relationship between the parties develops.
5	*Risk-taking Relationships* (RTR) is a function of trust and the perceived risk of the trusting behavior.
6	Outcomes of trusting behaviors, e.g. RTR, will lead to updating of prior perceptions of the ability, benevolence, and integrity of the trustee.

Table 2.2: Six propositions for trust

world, where it is common to interact with someone completely unknown in a once-off transaction, a complete analysis of the trustworthiness of the unknown entity might not be required if the transaction is not important or does not expose the trustor to any real risks. However, in situations where the trustor is subjected to real risks due to the transaction, being able to establish whether the services involved are trustworthy or not would be advantageous not just to the trustor, but also to the provider of the service. If a service is deemed to be trustworthy, a trustor would be more willing to interact with the service and pay a premium for the assurance that the service provides. This concept lies at the core of services such as eBay's reputation system[42].

In [40], Ratnasingham *et al* state that a lack of trust between business partners in Internet transactions creates a barrier to the uptake of Internet technologies for business transactions. They claim that uncertainties that exist are caused by a lack of awareness on the part of smaller companies primarily about "universally accepted business standards and policies" and thus their inability to confidently conduct online transactions. There is thus a need for a trust model that is capable of providing the desired assurances to provide the participating parties the safety of conducting online transactions, without introducing expensive and time consuming "one size fits all" trust models.

In this paper, the definition of trust is based on the definition by Mayer *et al*[30]. The definition provided is similar to the definition of decision trust described above, but includes the fact that the trusted relationship exists in spite of an inability to monitor or control the trustee. The paper proposes six propositions that contribute to the measure of trust between two parties - a *trustor*, the party that trusts another party, and a *trustee*, the party being trusted. The six propositions are listed in Table 2.2.

The parameters mentioned in proposition 2 in table 2.2 are described in the following sections.

Ability refers to the Web service's expertise in which the service is offered. One is much more likely to accept information regarding the current interest rate from a bank than a plumbing

company offering the service, as the calculation of the interest rate lies much more in the expertise domain of a bank. Ability alone does not convey trust, for a financial services group that has the expertise to calculate the correct interest rate might have no motive to do so, and may rather calculate one that provides them with more profit.

Benevolence refers to the good will of the company towards the trustor. Benevolence is a measure of the altruistic intents of a Web service in that the service will strive to produce the most effective results for the customer, regardless of the negative costs it may incur, such as processing expenses and post-sale customer support.

Integrity relates to the external perception of a service held by the public in general. Consistency in past actions is indicative of integrity. Integrity of a service is also conveyed by correlating previous claims and actions of a service, and a strong sense of fairness displayed by the service in previous engagements. Unlike *benevolence*, the *Integrity* rating applies to the collective view that a community has of a service. The integrity rating also extends to the view that authorities have of the service, so that potential customers can establish the integrity rating of a service provider and then decide whether or not to proceed with a transaction.

These three values, taken into account with a customer's *propensity to trust*, can be used to determine a level of trust the customer will have towards a Web service and the customer can then consider the necessary trust assurances required for a transaction. A customer's *propensity to trust* relates to a trust value that a client will assign to a service that the client knows nothing about. A trusting customer will place more trust in an unknown service to do the tasks it says it will than an untrusting customer who will need trust assurances before he will engage in the same transaction with the same amount of information. Some people are just more willing to trust than others, and should influence any trust model so that a level of assurance can be calculated that is suitable for the people involved and for the transaction at hand.

Time also plays a part in the determination of a trust factor. Time is represented by previous encounters and the public perception of a service. When a customer knows nothing about a service other than its own statements about its functionality, a customer has no way to determine a *benevolence* factor towards that service. Instead, the client will place more importance on the other two factors, namely *ability* and *integrity*. Over time, however, *benevolence* will increasingly become more important because it encompasses the previous transactions the customer has had with the service and the customer should be able to provide a *benevolence* factor from his previous experiences. The benevolence factor becomes more important as more transactions are concluded and eclipses the *integrity* rating because the integrity rating is a rating provided by external parties. An individual who has a good relationship with a service provider will stick with the service provider even if other individuals have had a bad experience. This is not necessarily true in situations where a service has committed a serious offence which affects its *integrity* rating significantly. This is why *integrity* and *benevolence* both play a part in the trust determination rather than being phased out over time.

An simplistic trust calculation formula is given below.

$$T = (aA.bB.iI)P_t$$

where *a, b and i* represent the importance factor of *ability (A)*, *benevolence (B)* and *integrity (I)* respectively, and where

$$a + b + i = 1$$

P_t represents the customers *propensity to trust* and affects the entire trust calculation. P_t is a multiplier such that

$$x \leq P_t \leq y$$

where x and y are lower and upper limits of the importance factor of a trustor's *propensity to trust*. The *propensity to trust* multiplies the trust calculation to alter the trust equation to impact on the amount of assurance needed to engage in a transaction to reflect the trustor's *propensity to trust*.

For a successful trust assurance calculation, T should be greater than the perceived risk involved in the transaction. As risk is an antecedent to trust, both must be computed in order to be compared. The risk of any transaction is both subjective and objective. Transactions involving large sums of money might be perceived to contain more risk simply due to the amount of money involved, but the risk could also be established by the accuracy of the information required, the controls in place to monitor the service and other factors that influence the risks involved in using a service.

The last factor involved in the calculation is the relative weighting of the three parameters, namely *ability*, *benevolence* and *integrity*. The weighting of benevolence becomes more important over time and its weighting therefore increases and reduces the importance of the integrity factor.

The trust defined here is dyadic, and suits the purpose for trust assurance between a trustor and trustee. The trustor could indicate his willingness to trust a third party by using the trustee's trust rating of other services, chaining the trust values together between multiple services. This introduces trust transitivity into a transaction if it is desired without adding complexity to the general trust calculation. The trustor could then easily obtain trust values for an entire transaction chain, and the transaction can then be evaluated based on the weakest link, or the average trust rating of the the entire transaction. By utilizing the concept of decentralized trust in the trust calculation, a customer has access to different trust models without changing the trust calculation protocol. A customer may choose to accept the rating of a trusted 3rd party or may choose to use a social control community to determine the trust rating. A trustor could even use both ratings in an attempt to obtain a more accurate rating. The trust model used in this dissertation is elaborated on further in Chapter 3.

2.6.5 **Data collection**

Information acquired about a customer in an online transaction may be collected explicitly or implicitly. Explicit data is any data supplied directly by a customer to the Web service such as the information provided in the registration process. Implicit data is data that concerns the

customer, but which has not been supplied directly by the customer. Examples of implicit data include transaction histories or search queries and can indicate preferences the customer may have but has not specified.[14] In a Web services environment, both types of data can be collected when the customer interacts with the Web service. All explicit data collection should be made clear to the user regarding the collection and use policies of the Web service. Implicit data collection may be beneficial to the Web service, and the customer might see the collection of such data as an intrusion of his privacy. Thus a customer learning of such practices might alter his benevolent perception of a Web service. Implicit data collection can provide a more personalized service to customers and can thus help in providing a better environment in which a transaction can take place. As long as the customer knows before the transaction whether or not his actions will be monitored and collected, his expectations will not change when he finds out that the service provider has collected data. Explicit data collection is discussed more thoroughly in Section 2.7.

2.7 LEGAL FRAMEWORK

In view of the trust and reputation systems that can be applied to online transactions and trust relationships, and the different trust models, there is a need to create a realistic base for trust models and calculations. Assumptions demanding too much from service providers without offering any incentives will not succeed, because the the burden of restricting the service provider's actions lies with the service providers themselves. There is no incentive for creating a more restricted environment in which to operate. In the physical world, people have been using many different cues and instincts to determine whether they can trust a stranger or not. In the online world, these mechanisms do not exist; a good service will send out the exact same messages that a malicious service will. As described earlier, there have been many trust models and computational trust models that use different techniques in order to entrench similar intrinsic trust evaluation techniques into computer applications. Whilst these models do provide novel ways of determining trust, the trust is still baseless in terms of the repercussions an offending party will face if he/she becomes opportunistic to the point where the end user is exploited for profit or some other gain, apart from a reduced reputation rating, if his/her transgression is reported. With the minimal cost of setting up the malicious service under a different name, if there are no "real world" repercussions for transgressions, malicious services will continue to operate if their transgressions are profitable.

It is for this reason that legislation can provide a sound grounding for trust evaluation and trust models. By using existing and upcoming laws, the establishment of requirements for a trust model becomes easier because parties have to adhere to the laws that apply to them. In the physical world, it is common to base rational decisions in part on the law - not explicitly, but rather as an implicit consideration that the law enforces cooperation under the penalty of punishment and compensation to the victim of a transgression. This is true for business sectors such as financial services, where there are strict rules regarding business operations that intrinsically put customers at ease when dealing with these companies. The value placed in legislation in determining trust is only evident in the absence of laws such that a person's confidence in a sector or a company is undermined by the lack of control in the sector.

Unfortunately, traditional laws and regulations concerning commercial transactions and busi-

nesses have not proven to be suitable for similar transactions and entities in the online environment. Legislation confined to a particular country or physical geographical location is no longer sufficient to an environment that has no physical space. The issues over control of electronic transactions, the validity of electronic messages and the authority of resolving conflicts that may arise in the online sphere have sparked the need for clearer legislation in the electronic transactions sector.

In [26], Johnson and Post illustrate the deficiencies of traditional laws by means of an example. They show how traditional trademark laws are dependent of a country and that different countries have different trademark laws. When the borders created by physical landmarks and geographical authority are removed, the issues regarding trademark enforcement become blurred. Legislation has to take cognizance of the nature of the Internet, electronic communication and electronic transactions.

Johnson and Post speculate that a local government's control is diminished in electronic transactions because the physical borders no longer contain the transactions and the way in which they are conducted. Whilst both countries might have claimed jurisdiction over the transaction or trademark, the courts many not have the personal jurisdiction over the foreign entity to order them to appear before the court. An entity can willingly subject itself to the authority of a foreign courts in contract law, but a company does not have to submit to a foreign jurisdiction. A recent example of incompatible jurisdiction involved a company based in Illinois that sued Spamhaus, a British-based non-profit organization.[1] The Illinois court claimed jurisdiction over Spamhaus. Full details of the legal battle can be read on the Spamhaus website, http://www.spamhaus.org/

The closest resemblance electronic transactions have to traditional transactions is that there is a offer and the option to accept or reject the offer. The contract entered into between the parties involved is still binding, and new legislation has created the foundations in which the contract can be proved or disproved. International trade law has created model laws which allows different countries to start from a common understanding of terminology, the requirements for data message storage and evidentiary value. The United Nations Commission on International Trade Law (UNCITRAL) [47] has created model laws regarding electronic commerce, electronic signatures, legal validity of computer records and the use of electronic communications in international contracts[2].

2.7.1 Data privacy laws

Data privacy has become a paramount concern in electronic transactions. Studies of websites have shown that the number of websites displaying privacy policy information has increased from 14% in 1998 to over 88% in 200 [20]. This has mostly been a self-regulatory step as commercial websites have addressed the concerns of customers who make use of commercial websites. In the United States of America, with the exception of the state of California, there are no laws enforcing the use of privacy policies[7]. However, the Federal Trade Commission may investigate companies who breach their own privacy policies. In contrast, the European Union

[1] Spamhaus Legal Challenge : http://www.spamhaus.org/legal/answer.lasso?ref=3

[2] UNCITRAL Model Laws : http://www.uncitral.org/uncitral/en/uncitral_texts/electronic_commerce.html

Directive on Privacy and Electronic Communications and the Electronic Communications and Transactions Act of 2002 both require service providers to display privacy policies and conform to the data privacy regulations stated in the respective laws. The laws are described in greater detail below.

The need for legal acknowledgement of electronic transactions has spurred a great deal of activity in legislative sectors to recognize electronic data as legal equivalents of paper-based transactions. The UNCITRAL model law describes which aspects of electronic commerce should be enforced by legislative measures [47]. This model law has influenced the creation of legislation in over 25 countries, including the United States of America, England, France, South Africa and China. While these laws give credence to the validity of electronic transactions, the laws do not necessarily include concepts of a consumer's or client's right to data privacy. The problem of ensuring data privacy protection can be achieved in two different ways. Firstly, there are legislative measures which enshrine a user's right into law, and secondly, there are self-regulatory standards that apply to specific sectors. In particular, America has decided to regulate certain sectors in terms of personal privacy protection. The most recognised American regulations concerning protection of personally identifiable information collected electronically are:

- The Health Insurance Portability and Accountability Act of 1996 (HIPAA)
- The Children's Online Privacy Protection Act of 1998 (COPPA)
- Financial Modernization Act of 1999, also known as the "Gramm-Leach-Bliley Act."

The laws above all define how companies should treat personal information. These laws have caused as much turmoil to the operations of websites as they have provided benefits to consumers. As an example, the number of complaints to hotmail, a web-based email service, increased by 1,150% in April and May 2000 after it implemented safeguards listed in COPPA [20]. This example shows that legislation cannot provide a panacea for all privacy concerns, and in fact, under the COPPA Act, in some cases customers had to reveal more information about him/herself to prove his/her age. However, legislation is crucial to the wider adoption of electronic commerce. Until a customer can be given the guarantees that other forms of commerce can, it will never grow into the market it is envisaged to be. A customer must be confident that he/she can use his/her country's legal system to resolve disputes.

The European council and member states recognize this fact and member states are proceeding to roll out legislation derived from the UNCITRAL model law and the European Council's directives on Privacy and Electronic Communications [16]. The directive, known as Directive 2002/58/EC of the European Parliament and Council of 12 July 2002, extends the privacy afforded to persons and derives its meaning of "personal information" from the European Council's directive on the Protection of individuals with regard to the processing of personal data and on the free movement of such data of 1995, also known as Directive 95/46/EC. Directive 2002/58/EC was created for the electronic communications sector and extends the more general Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 [15]. The directive defines personal data as "any information relating to an identified or identifiable

natural person (data subject); an identifiable person is one who can be identified, directly or indirectly, in particular by reference to an identification number or to one or more factors specific to his physical, psychological, mental, economic, cultural or social identity..." Directive 2002/58/EC also defines who holds the rights to personal information and what rights are given to the person who the personal information identifies.

Legislation, such as South Africa's Electronic Communications and Transaction Act 25 of 2002 (ECTA) [37] defines several roles that service providers must fulfil, sets out the requirements of electronically binding contracts, rules for companies selling goods or providing services online and defines what information is considered personally identifiable information (PII) and specifies how companies should handle such information. Legislation like the ECTA gives consumers more right to contest business transactions and unlawful use of their PII. The ECT act of 2002 is derived from the UNCITRAL Model law for electronic commerce. It therefore stands to reason that Web services based in South Africa should be able to negotiate appropriate terms under which to exchange personal information with other countries that provide similar laws. The ECT act defines personal information as:

(a) information relating to the race, gender, sex, pregnancy, marital status, national, ethnic or social origin, colour, sexual orientation, age, physical or mental health, well-being, disability, religion, conscience, belief, culture, language and birth of the individual;

(b) information relating to the education or the medical, criminal or employment history of the individual or information relating to financial transactions in which the individual has been involved;

(c) any identifying number, symbol, or other particular assigned to the individual;

(d) the address, fingerprints or blood type of the individual;

(e) the personal opinions, views or preferences of the individual, except where they are about another individual or about a proposal for a grant, an award or a prize to be made to another individual;

(f) correspondence sent by the individual that is implicitly or explicitly of a private or confidential nature or further correspondence that would reveal the contents of the original correspondence;

(g) the views or opinions of another individual about the individual;

(h) the views or opinions of another individual about a proposal for a grant, an award or a prize to be made to the individual, but excluding the name of the other individual where it appears with the views or opinions of the other individual; and

(i) the name of the individual where it appears with other personal information relating to the individual or where the disclosure of the name itself would reveal information about the individual, but excludes information about an individual who has been dead for more than 20 years.

The description defines restriction for information that is personally identifiable information, such as a first name and last name, but also places restrictions on private information, such as race and religion. The similarity between the definitions provided in the ECTA and the European Council Directive 2002/58/EC allows for a compatible trust assurance protocol. In this dissertation, focus is given to the ECTA, but the protocol developed will be compatible with the European data privacy laws, and should be compatible on the rest of the electronic communication requirements as both are derived from the UNCITRAL model law on electronic commerce.

2.7.2 **ECTA**

The aim of ECTA is "to provide for the facilitation and regulation of electronic communications and transactions; to provide for the development of a national e-strategy for the Republic; to promote universal access to electronic communications and transactions and the use of electronic transactions by SMMEs; to provide for human resource development in electronic transactions; to prevent abuse of information systems; to encourage the use of e-government services; and to provide for matters connected therewith." In this research, only the sections of interest in the act are used. These sections deal primarily with the requirements for the providers of online services, the privacy rights provided for consumers in the online environment, and the legal weight of electronic data messages. The ECTA provides numerous requirements that have to be met by companies or individuals conducting business transactions in an electronic environment. An Example of this includes Chapter VII, dealing with comsumer protection. The chapter is comprised of eight sections. Section 42, entitled "Scope of application," states that the chapter of consumer protection only concerns electronic transactions. This means that there are legislative differences between transactions conducted in offline environments, and transactions conducted using electronic messages.

Section 43 is of great interest and potential application because it stipulates exactly what information suppliers must make available to consumers. The section, entitled "Information to be provided", lists 18 information categories that a supplier must make available as a provider of goods or services in an electronic manner.

1 Its full name and legal status

2 its physical address and telephone number

3 its website address and e-mail address

4 membership of any self-regulatory or accreditation bodies to which that supplier belongs or subscribes and the contact details of that body

5 any code of conduct to which that supplier subscribes and how that code of conduct may be accessed electronically by the consumer

6 in the case of a legal person, its registration number, the names of its office bearers and its place of registration

7 the physical address where that supplier will receive legal service of documents

8 a sufficient description of the main characteristics of the goods or services offered by that supplier to enable a consumer to make an informed decision on the proposed electronic transaction

9 the full price of the goods or services, including transport costs, taxes and any other fees or costs

10 the manner of payment

11 any terms of agreement, including guarantees, that will apply to the transaction and how those terms may be accessed, stored and reproduced electronically by consumers

12 the time within which the goods will be dispatched or delivered or within which the services will be rendered

13 the manner and period within which consumers can access and maintain a full record of the transaction

14 the return, exchange and refund policy of that supplier

15 any alternative dispute resolution code to which that supplier subscribes and how the wording of that code may be accessed electronically by the consumer

16 the security procedures and privacy policy of that supplier in respect of payment, payment information and personal information

17 where appropriate, the minimum duration of the agreement in the case of agreements for the supply of products or services to be performed on an ongoing basis or recurrently

18 the rights of consumers in terms of Section 44, where applicable.

In addition to the information to be made available, the supplier or service provider must allow a consumer to review the entire electronic transaction, correct any mistakes that the consumer may have made during the process of procuring the goods or services, and to withdraw from a electronic transaction before finally placing the order. Subsection three of Section 43 entitles the consumer to cancel the transaction within 14 days of receiving the goods if the supplier failed to provide the necessary information or didn't allow the consumer the required mechanisms to review the transaction, fix mistakes made or the ability to withdraw from the transaction before the order was placed. In addition to these requirements, the supplier is liable for damages suffered by a consumer resulting from inappropriate payment mechanisms being used to facilitate the transaction.

Section 43 of the ECTA provides an idyllic basis for a trust assurance protocol. Every service provider that falls under the jurisdiction of South Africa or a country that has implemented 2002/58/EC has to provide all the information presented to customers. A trust assurance protocol based on the information that has to be supplied is applicable to the "real world," because the assumption that the service provider has to present all information for the trust assurance protocol is not actually an assumption, but rather a legislative requirement.

Chapter VIII of the ECTA deals with the protection of personal information collected through electronic means. In ECTA, the section that stipulates the personal information protection procedures, Section 51, is voluntary and it is up to the supplier of the electronic service to subscribe to the entire section or not at all. If a supplier or service provider subscribes to the principles outlined in the section, he must make provision for all the following:

1 A data controller must have the express written permission of the data subject for the collection, collation, processing or disclosure of any personal information on that data subject unless he or she is permitted or required to do so by law.

2 A data controller may not electronically request, collect, collate, process or store personal information on a data subject which is not necessary for the lawful purpose for which the personal information is required.

3 The data controller must disclose in writing to the data subject the specific purpose for which any personal information is being requested, collected, collated, processed or stored.

4 The data controller may not use the personal information for any other purpose than the disclosed purpose without the express written permission of the data subject, unless he or she is permitted or required to do so by law.

5 The data controller must, for as long as the personal information is used and for a period of at least one year thereafter, keep a record of the personal information and the specific purpose for which the personal information was collected.

6 A data controller may not disclose any of the personal information held by it to a third party, unless required or permitted by law or specifically authorized to do so in writing by the data subject.

7 The data controller must, for as long as the personal information is used and for a period of at least one year thereafter, keep a record of any third party to whom the personal information was disclosed and of the date on which and the purpose for which it was disclosed.

8 The data controller must delete or destroy all personal information that has become obsolete.

9 A party controlling personal information may use that personal information to compile profiles for statistical purposes and may freely trade with such profiles and statistical data, as long as the profiles or statistical data cannot be linked to any specific data subject by a third party.

In ECTA, a data controller is defined as "any person who electronically requests, collects, collates, processes or stores personal information from or in respect of a data subject."

Web services must work within this legislative environment and provide users with the confidence that the entire Web services transaction can be logged so that an authoritative body can

determine who was involved in a business transaction and what role they played in that particular transaction. The following section introduces the need for a framework in which such an auditing trail can be created.

2.8 ONTOLOGIES

An ontology, in a computer science context, is a markup language designed so that software applications can process the data included in the ontology, instead of just presenting it to humans. Ontologies represent data and the links between data so that a computer can infer and create new information from the information represented in the ontology. Ontologies store information in different elements, allowing a computer to infer information in which data is stored.

Ontologies describe information as classes, individuals, attributes and relations. Classes contain individuals and other classes and can be seen as collection or a set. Individuals represent the basic object that are represented in the ontology. Attributes expand the knowledge represented in classes and individuals by elaborating on the properties of the classes or individuals. An important aspect of an ontology are the relations between classes so that new classes can be inferred from the existing classes according to their rules and properties.

Ontologies can also describe vocabularies for different domains. Ontologies can act as taxonomies that classify different elements into an hierarchial structure. A vocabulary can be shared and extended between different parties by defining a vocabulary in such a manner.

2.8.1 OWL

According to the World Wide Web Consortium (W3C), "The OWL Web Ontology Language is a language for defining and instantiating Web ontologies." Ontologies are core to the evolution of a new Internet. There is a great deal of data on the World Wide Web, and much of it is linked so that one can navigate one's way through the mass of information relatively easily and smartly. Computers are not capable of distinguishing between different types of data and cannot inherently produce useful results to a query. The aim of the semantic web is to tag information with structured information that computers can follow and thus produce better results themselves. Ontologies create vocabularies that contain relational information between different types of information, so that computers can follow the relationships and infer information from existing relationships and produce results that are not currently possible with text marked up primarily to be display to humans [32].

OWL, a vocabulary extension of RDF, the Resource Description Framework[31], is becoming known as a building block for the semantic web[31]. Instead of just presenting information in a manner that can easily be interpreted by people, OWL's goal is to represent data in a manner that is more accessible to computers. The OWL language provides for greater machine interpretability of a document and allows a computer to gather or infer information about data items contained in an OWL document. OWL supersedes the Resource Description Framework (RDF), which was created by the W3C as a formal data model, written in XML, to represent and describe online resources such as websites and the content of the web resources. The core of the framework revolves around machine readable metadata, enhancing a computer's ability to consume the web resource and its content.

Ontologies also provide the functionality provided by taxonomies, and as such is capable of providing vocabularies so that different computer agents are able to operate on the same semantically tagged information in a consistent and reliable manner. Vocabularies of common information and their meaning is crucial in the abstract interpretation mechanisms that computer interpretation can provide.

Ontologies are different from taxonomies, which is just a hierarchial structure. An ontology can represent a taxonomy quite easily, and the OWL-Lite sub-language of OWL is designed to conform to this need. The advantage of ontologies over taxonomies is that ontologies contain properties as well as classes and individuals. Properties allow the designer of the ontology to state general facts about a class (and therefore its members) and specific facts about individuals.

In OWL, great care is taken to ensure the differences between classes and individuals. A class consists only of an identifier and the properties regarding that class with respect to other classes. Classes represents sets of individuals. Individuals are entities that belong to classes. Classes describe individuals, whilst individuals comprise the actual entities in classes. The granularity of the design of the ontology regarding the classes and individuals is quite important, as an entity could validly be either a class or an individual. The choice made in the design has a large effect on the sub-language of OWL that the ontology will fall under. Below is a brief description of the three sub-languages of the OWL language.

OWL is separated into three distinct variations:

- OWL-lite
- OWL-DL
- OWL-Full.

OWL-Lite is a subset of OWL-DL which in turn is a subset of OWL-Full. OWL-Lite is suitable for classification hierarchies and simple constraints between different classes. It was created so that people currently using taxonomies could easily migrate their vocabulary to be OWL compliant. OWL-DL is a subset of OWL specifically designed to be as expressive as possible, but still retain computational completeness and decidability[31]. This means that any inference drawn from the ontology will finish in finite time. OWL-DL has more expressive capabilities then OWL-Lite, but forgos some complexity due to the goals for completeness and decidability. The DL in OWL-DL stands for Description Logics, which is the logics on which the OWL language is based. OWL-Full is the most expressive of the three subsets, but complete computational reasoning is unlikely as there are no guarantees that a computation will complete [31].

For the semantic web, only OWL-Lite and OWL-DL is suitable as an ontology that cannot be guaranteed to return a result when queried will cause too many problems in the semantic web.

2.9 SUMMARY

The background chapter provides the base from which a trust assurance protocol can be developed. By looking at the underlying technologies, trust systems used in literature, and legislation concerning electronic transactions, a protocol can be developed that utilizes the best available technologies, trust models and conforms to legislation. All the different topics discussed above provide a foundation for a protocol and model that spans across different domains to ensure that the customer is provided with all the necessary assurances that the technology, legislation and trust models can provide.

CHAPTER 3

TRUST ASSURANCE

3.1 INTRODUCTION

Trust is an ambiguous term with different meanings when used in different circumstances. Academic research in psychology, sociology and computer science has defined trust over and over, and its definition has changed in every field. In the background chapter, various definitions of trust was described and a model for trust that will be used in this dissertation was introduced. While the model is conceptually intuitive and elegant, it is still too ambiguous. There still needs to be a common set of parameters so that the trust model can be used without requiring intensive manual labour. The terms *ability*, *integrity* and *benevolence* are intuitive in their meaning to humans, but in order to be able to provide a consistent trust evaluation, these terms need to be defined with specific parameters, so that they can be evaluated consistently.

This Chapter first extends the trust model described in Section 2.6.4 and then establishes parameters with which to populate the trust model.

3.2 TRUST MODEL

In order to use the definition provided by Mayer *et al* [30] to calculate a trust level, we need to establish parameters for each of the three characteristics of the trust relationship, namely *ability*, *integrity* and *benevolence*.

There is a need to add semantic metadata to a company to know how we should evaluate a company that we intend to interact with. A trustor should know that a bank operating in South Africa should at the very least adhere to the Banks Act, along with the ECTA if the transaction is being conducted online. The trustor should also know that the South African Reserve Bank is responsible for bank regulation and supervision in South Africa. The Reserve Bank issues banking licences to banks and thus acts as an authority and is capable of verifying a service's claims about being a bank. Without knowledge of such an authority existing for a particular industry, a malicious service provider could conceal this information from the trustor. There is therefore a need to be able to stipulate the authorities, and legislation that a particular company has to adhere to, without requiring such information from a service itself.

The first requirement of this is being able to establish in what industry sector the company op-

erates. Without this information, finding the right accreditation and verification bodies would certainly be impossible. The information about the industry sector, combined with the country in which the company operates, will allow a trustor to find suitable information about the authorities in the sector.

In order to establish the type of company that is being dealt with, we propose using the International Classification Benchmark to categorize companies into different sectors, where industry and sectoral legislation, codes of conduct and authorities can easily be determined, without blindly accepting the word of the company that is being evaluated.

In order to facilitate the automatic dissemination of the classification information, the authors have created an ontology to classify companies according to their primary function. The ontology is based on the Industry Classification Benchmark (ICB) [1], developed by the Dow Jones Indexes and FTSE in 2004 to classify different types of industries that are represented in stock exchange markets worldwide. Whilst this classification designed for the stock markets, the sectors and industries cover all the sectors that can be used to describe a company. This classification places a company into exactly one category, and can be extended to placing the company in different categories for each domain of expertise that the company is involved in. This ontology can be combined with information regarding legislation that the company has to adhere to and information about self-regulatory bodies, accreditation bodies, and industry associations so that a trustor can establish exactly what requirements a service must meet to conform to the requirements set out for it by all the appropriate authorities. Without this information, the trustor will have no starting point in determining whether the service is a lawful and verified service or a malicious person masquerading as a legitimate company.

The ontology will be introduced and examined in detail in section 4.4. In Figure 3.1, the ontological representation of the ICB is shown to illustrate the granularity of the classification benchmark. By associating information about accreditation bodies, codes of conduct, legislation, and verification bodies for a particular industry sector for a particular country, a trustor can easily determine whether a company belongs to all the necessary accreditation bodies for its sector. In Figure 3.1, a sample class is shown from the entire ontology. The enlarged class shows the Travel_Leisure_SuperSector, and contains the various sub-sectors for airlines, travel and tourism, hotels, restaurants and bars, the gambling sector and the recreational service sector. The ontology allows information about legislation, accreditation bodies, self-regulatory bodies, industry watchdogs and other information to be added to each class of the ontology. Any authority body listed for the Travel_Leisure_SuperSector will also apply to the sub-sectors, so that general laws that apply to the entire super sector also apply to the sub-sectors.

Ability

To determine the ability of a trustee, the trustor must evaluate the trustee against the particular context of the transaction they are entering in to. Ability is domain specific and a trustee must be evaluated for each expertise domain that the trustee offers. For example, the perceived ability of the South African Weather Service (SAWS) to forecast the weather would lead to a higher trust rating of their ability than the trust perceived from them manufacturing equipment un-related

[1] Industry Classification Benchmark : http://www.icbenchmark.com/

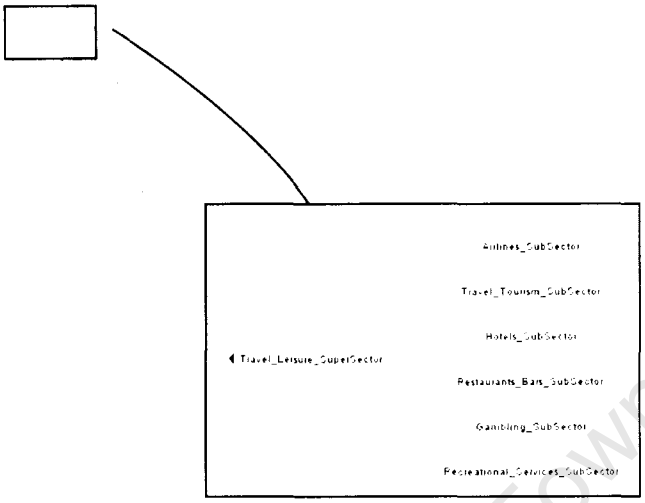

Figure 3.1: Graphical representation of the Industry Classification Benchmark ontology

to weather. Whilst this seems logical to humans, computers can not automatically determine that the South African Weather Service should be more capable at predicting the weather than producing manufacturing equipment.

To determine the ability of a company in a particular domain, several parameters can be evaluated to produce a rating of the overall ability of a company:

- Accreditation bodies
- self-regulatory bodies
- public knowledge of company's expertise in domain
- an ability rating from a trusted third party
- the company type, according to the international classification benchmark
- the company's public certificate
- an evaluation of search engine rating
- past history of expertise in domain provided by an authority in the sector.

A trustor's knowledge about relevant accreditation bodies and self-regulatory bodies prevents trustees from claiming that they meet all the criteria specified for them when they do not. A company can try to fool a trustor by omitting the presence of a verification body in a particular sector, but with the information available to the trustor from the ICB ontology, any such misdirection will be found out. Without such an information repository, the trustor would have no way to determine whether there are additional verification bodies or authorities in a sector. The

values for accreditation bodies and self-regulatory bodies come from querying those bodies to verify the existence of a company in their records. Public knowledge of a company's expertise is another source of information about a company. Here, the value of shares listed in stock markets can give an indication about a public company's performance in a given sector. Private companies might find it beneficial to advertise their expertise so that public knowledge of their expertise increases and combined with previous history, enhance any trust derived in this manner. The value generated for the public knowledge of expertise is received by querying a community endpoint which contains sufficient information gained from the online community. The trustor can also query the community itself, and search public forums for information about a particular company.

Independent trusted third parties also have an effect on the perception of a company's ability. In particular, if products or services are endorsed by the South African Bureau of Standards (SABS), then that should create a positive effect for a company's products or services. Similarly, a company that adheres to the ISO9001 quality standard should have that achievement impact its perceived ability rating. A third party entity with knowledge of an industry and an incentive to produce reliable and accurate information could be used to receive a rating for a company's ability rating.

A company that provides a service outside its primary business market should be treated more cautiously than a company entrenched in a particular market. Following an example used earlier, a consumer should trust weather information more if it came from the South African Weather Service that from a reputable cosmetics company. If a service offered by a company exists in the same classification as the company, then a high rating will be recorded for the company on this requirement. If the domains differ slightly, then a lower rating will be afforded and if the expertise domain of a service is completely different to the expertise domain of the company, a low rating will be awarded. A company's public certificate can also serve as a guide that the company is legitimate and if the trustor only needs minimal assurance to a company's authenticity, then the trustor may decide to only validate a trustee's public certificate to prove authentication. A company without a public certificate could indicate that the company is not able to deal with secure transactions and will receive a rating of -1. If a company's certificate appears on a certificate revocation list, it is up to the trustor to decide whether or not to continue with the transaction.

Search engine results can point to a company's popularity and status in a particular field and give the trustor an indication about whether a company is a well-known establishment. Referring back to the example of the South African Weather Service, a internet search for "South African Weather Service" produces 71,500 results, a search for "South African Weather Service +forecast" produces 15,700 results and a search for "South African Weather Service +manufacturing" produces 604 results. Whilst these results do not prove anything conclusively, they do suggest that the public record shows that SAWS is more recognized with forecasting than manufacturing. Previous history of successful transaction and referees can also vouch for the ability of a company in a certain field, A third party, community or both can provide a rating for the history of the company's ability.

Integrity

Integrity consists of the external rating of a trustee by a community for all factors other than those that contribute to its ability rating. The integrity of a trustee is a statement about their conduct in general, over and above their abilities in the specific context in which the transaction is taking place. The following parameters describe the integrity of a service:

- Third party evaluation
  - public reputation
  - historical legal challenges
  - current legal challenges
  - consistency in actions
  - operational openness
- Directly observed
  - acceptance of relevant codes of practice
  - adherence to voluntary legislation (such as Section 51 of the ECTA)
  - adherence to mandatory legislation
  - perceived sense of justice
  - observed consistency in actions
  - perceived operational openness
  - the jurisdiction in which the company operates.

Third parties refer to established companies and entities who solely evaluate and authenticate other companies. The need for assurance about the authentication of a website or company online created a new business model and started the boom in the certificate authority market. Third party evaluations also refer to a new breed of authorities, such as trust authorities and other authorities that attempt to provide more assurances than basic authentication and communication of a company's public key. These third parties provide ratings to consumers about companies so that the consumer has some indication about a company before engaging in a transaction with the company. An example of a third party would be the recommendations of an industry watchdog, which the industry has come to respect and is seen as a fair and unbiased in order to give an accurate reflection of the companies in the particular sector. A public reputation is the integrity level that a community holds of a company or service. This could be a reputation system, such as the reputation system used in EBay[41], or it could be a more general rating built into a social community that has dealt with a certain company. Historical legal challenges refer to previous lawsuits filed against a company. Of particular interest are lawsuits of negligence and malice regarding the services being evaluated. Not all lawsuits directed against a company will affect a company or service, but particular lawsuits directed against the company should be noted when evaluating the company's integrity. Similarly, the current legal challenges faced by a company

will influence its integrity rating and should thus be taken into account. Ratings about the legal challenges can sourced from appropriate authorities or from a community. The rating rates the company relative to the rest of the sector to determine the weight of the legal issues faced by a company. Consistency in actions influences the integrity rating either positively or negatively because of the confidence a trustor will have in a trustee if he is deemed consistently good. A consistently bad trustee should be avoided if possible and an inconsistent trustee should be approached with caution. The consistency of a company can be measured by a community of agents, a third party, or even investigating previous financial records to ensure that the company is consistent in its business practices. Operational openness refers to the amount of information disclosed to a trustor about the transaction and the progress thereof. Openness in the transaction instils a greater sense of assurance and therefore open business practices, where a trustor can check on the progress of his transaction, will increase his confidence in the transaction and the company he is dealing with. The openness rating can be sourced from a community of agents and provides a rating relative to the rest of the industry.

Directly observed ratings about integrity are the ratings about the integrity that a consumer can gather about a company without the need for third party assistance. Third parties could also provide ratings for the conditions that can be directly observed, and the trustor could prefer to delegate the entire integrity check to third parties, either centralized authorities or decentralized communities.

If a company publicly states that it accepts codes of practice which an industry has set, any violation of the codes of practice will leave the company or service liable and affords the trustor with the ability to challenge the company in a legal institution for compensation. A company who refuses to agree to a code of practice or declines to advertise it should be treated with more caution, because any violation of the code of practice does not afford the client the chance to seek for recompensation based on those particular grounds. ECTA contains a voluntary section, Section 51, which sets out rules regarding the electronic collection of personal information. A company that subscribes to a voluntary section of legislation should be deemed to have a strong sense of justice, which would influence its integrity. Similarly, a company that adheres to the mandatory sections of legislation should be seen to have more integrity than companies who have been found contravening the law. A trustor's perception of a trustee's sense of justice will provide assurance to the trustor if the trustee is perceived to have a strong sense of justice. This means that the trustee will abide by the contract between the two parties and will provide assurances to the trustor. Consistency in actions are exactly the same as described previously, but takes into account that directly observed behavior has a greater influence than a third party's recommendations. Lastly, the jurisdiction in which a company operates is an important factor in choosing a trading partner. If there are two services offering the same service, the service operating in the same jurisdiction as the trustor will be more appropriate because any dispute can be resolved without the need to cross jurisdictions.

Benevolence

Benevolence is the most personal criterion used to evaluate a trustee. A service might have a good ability and integrity rating according to past history and third party recommendations, but previous experience is a significant factor when deciding how trustworthy a trustee is. In the real

world, once a person has had an experience with a company or service, it becomes a large part of his decision to use the service again, despite the fact that the ability and integrity rating of the company stays the same. If a person experiences bad service at a restaurant or receives faulty goods from a company, he will take that into account before returning to the same restaurant or company, despite the external view of both business being good. The following parameters describe benevolence:

- Previous interaction experience
  - normal business operations
    - timely delivery of goods or services
    - full compliance to business contract agreement
  - extraordinary service
    - customer service
    - refunds of exchanges
- perceived altruism.

Benevolence relies on direct previous interaction with a company or services. A rating of previous business experiences with a particular company will influence the overall trust value that a trustor holds over a trustee. If a transaction runs smoothly and completes as per the agreement of the business contract, then the personal trust rating of the trustor towards the trustee will increase. Similarly, if something went wrong in a business transaction, a trustor might decrease his trust rating towards a trustee. Extraordinary service, whether it is good customer support services or efforts to rectify mistakes made by the trustee, will increase the level of benevolence that the trustor holds of the trustee. Refunds and exchanges that occur without hassle will also increase a customer's benevolence towards a company and make them more willing to transact with the trustee again. The ratings for the benevolence factors involved are supplied by the trustor himself, as he is the only one with information that can populate the benevolence ratings. The range of the ratings are -1 to 1 and specify the worst possible rating to the best possible rating respectively.

3.2.1 Parameters for trust calculations

The parameters of the trust equation listed above provides a trustor to use different trust derivation techniques and form them into a trust value that suits his needs for a particular transaction. The trustor has to look at which parameters are feasible to obtain, the risk involved in the transaction, the assurances he would like to have before entering the transaction, which of the parameters are the most important and then evaluate the trustee with all the information that the trustee possesses. It would make sense that a trustee would ensure that an online car dealership has a good reputation, and a good rating from third parties and other previous customers. It would however, not be suitable to go through the effort of establishing the same information and calculate the trust level the trustor has in a company or service that provides a directory lookup or provides a free currency convertor calculator. The client must be able to state his

general preferences and be able to rate the previous parameters in a manner that is suitable for the transaction at hand and then calculate a trust rating that is relevant to the transaction.

We propose a subjective rating scale to indicate the relative importance of each of the parameters described above:

1 Not applicable

2 Not important at all (0.25)

3 Not very important (0.75)

4 Neutral (1)

5 Important (1.25)

6 Very important (1.75)

7 Deal breaker.

The numbers shown in the brackets represent the multiplication factor with which the value will be multiplied. The last item has no multiplication factor, as any transaction that fails a condition labelled as "deal breaker" will cease immediately. Thus if the transaction matches the conditions labelled as deal breaker, it is effectively ignored for the rest of the calculation.

An example of the importance factor ratings is shown in Table 3.1. The importance rating above simply describe the multiplication factor with which each rating will be multiplied, except that any rating of 1, i.e a factor of no importance will be removed from the trust calculation equation. A rating of 5 means that the condition must be true or else the transaction will fail. That is the reason why there is a threshold column in table 3.1 to describe the minimum rating required before the transaction can proceed. It is important to note that some of the values that will be obtained from evaluating the parameters will be discrete values, such as "Public Certificate" and that other parameters will produce a value within a range. For discrete values, a value of -1 signifies failure or the parameter test and 1 signifies a pass. The other factors can also represent subjective trust ratings, as a parameter of "Jurisdiction" can only be evaluated once a list of suitable or unsuitable jurisdictions is created. The trustor can either assign a value to a jurisdiction or group jurisdictions into groups so that a jurisdiction can be evaluated against the trustor's preferences.

3.2.2 **Trust calculation**

The trust calculation is dependent on several factors. Firstly, any calculation needs to account for the relative weights for the three parameters, namely, Ability , Integrity and Benevolence.

In figure 3.2, the graph for ability is defined by

$$P(x) = a$$

	Name	Rating	Threshold
Ability	Accreditation bodies	4	
	Self-regulatory bodies	4	
	Public rating of expertise in domain	5	
	Third party ability rating	5	
	Company type	4	
	Public certificate	7	
	Search engine company rating	4	
	Historical ability in sector	4	
Integrity	Third party integrity rating	4	
	Public reputation	4	
	Historical legal challenges	4	
	Current legal challenges	4	
	Consistency	4	
	Operational openness	4	
	Acceptance of codes of practice	4	
	Voluntary legislation	4	
	Adherence to legislation	4	
	Sense of justice	4	
	Observed consistency	4	
	Perceived operational openness	4	
	Jurisdiction	5	
Benevolence	Timely delivery of goods or services	2	
	Adherence to business contract	2	
	Customer service	2	
	Refunds or exchanges	2	

Table 3.1: Table showing the trust evaluation parameters along with the importance of each.

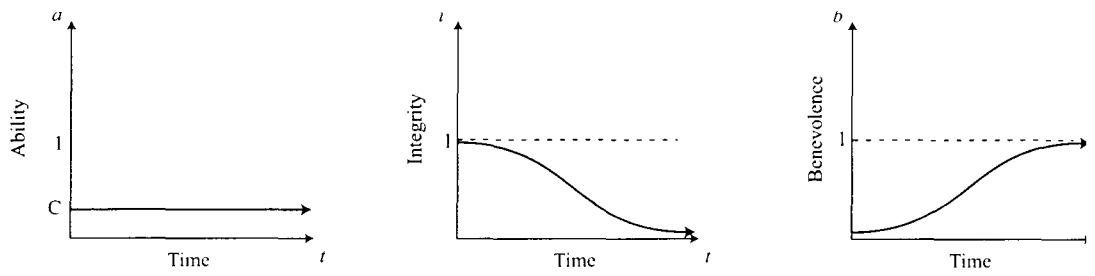

Figure 3.2: Relative parameter weights over time

This indicates that the relative importance of ability does not change as the number of interactions between two parties increase.

In Figure 3.2, the graph for Integrity is defined by

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2}$$

Initially, integrity accounts for much of the decision about whether to use a new and unknown service. Because a benevolence rating doesn't exist yet, it is very difficult to know how a service will act towards a trustor, and thus the integrity rating becomes a good indicator of the level of service to expect. Over time, as more interactions occur between the two parties, the integrity rating will become less important because the trustor will have personal knowledge of the service which will influence his decision whether or not to use the service again.

In Figure 3.2, the graph for benevolence is defined by

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2}$$

where $\sigma = 2\mu$ to shift the curve to the right so that the curve is in the range $0 \leq t \geq 2\mu$.

Initially, a trustor will have no knowledge of a service and thus will not know personally how that service will act towards him. For this reason, the weighting of the benevolence factor will not matter because the trustor has no information on which to base his/her decision. As more transactions take place between the two parties, the trustor will be able to work out how benevolent the trustee is towards him and it will influence his/her decision whether or not to use the service more than the integrity rating received from other sources. The integrity rating will become less important as personal experience is more important than knowledge about the service from other people.

An example parameter evaluation Table is shown in Table 3.2.

This example is between a trustor and a trustee where no previous transactions have occurred between the two. The trust evaluation rating can then be worked out as:

$$T_v = \sum_{i=0}^{n} \frac{f_i(V_i)}{f_i} \tag{3.1}$$

The sum does not include any items without values, or items with a value rating of "Not important at all" or "Deal breaker." Each parameter category, namely ability, integrity and benevolence, is summed up individually so that the relative weight factor can be applied.

Ability has a constant weighting factor of 0.33 so $A = 0.33(0.624) = 0.206$

	Name	Rating	Threshold	Value
Ability	Accreditation bodies	4		0.75
	Self-regulatory bodies	4		1
	Public rating of expertise in domain	5		
	Third party ability rating	5		0.5
	Company type	7	1	1
	Public certificate	7	1	1
	Search engine company rating	1		0.14
	Historical ability in sector	4		0.4
Integrity	Third party integrity rating	4		
	Public reputation	4		0.6
	Historical legal challenges	3		-0.2
	Current Legal challenges	3		0
	Consistency	4		0.7
	Operational openness	4		0.4
	Acceptance of codes of practice	4		1
	Voluntary legislation	4		0
	Adherence to legislation	4		1
	Sense of justice	4		0.34
	Observed consistency	4		0.3
	Perceived operational openness	4		0.24
	Jurisdiction	4		0.5
Benevolence	Timely delivery of goods or services	1		0
	Adherence to business contract	1		0
	Customer service	1		0
	Refunds or exchanges	1		0

Table 3.2: Table showing the trust evaluation parameters along with their values.

Integrity has a weighting of

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(0-\mu)^2/2\sigma^2} = 0.9974$$

which means that $I = 0.9974(0.394) = 0.393$

The trust value is calculated by combining ability, which accounts for one third of the value, with integrity and benevolence, which accounts for two thirds of the trust value.

Benevolence is not calculated for this transaction, as it has no applicable values. If the trustor interacts with the same trustee that is in this transaction, the integrity weighting would drop, the benevolence weighting would rise and the trustor would have values for the benevolence factors.

Therefore, the entire trust calculation deems the trustee to be trustworthy with a rating of $0.206 + \frac{2}{3}0.393 = 0.468$ If the customer is happy with a trust evaluation of 46.8%, then the trustor can proceed with the transaction.

3.2.3 Trust assurance protocol

In order for customers to engage in online transactions, it is necessary to determine the risks involved in the transactions. There are several factors that have to be considered before a customer will be able to determine whether or not to initiate the transaction.

- Importance of the transaction
  - If the customer needs the transaction to progress, and has no other method of achieving the required result, he might have to initiate the transaction regardless of the risks involved.
- Protection of personal information
  - If the customer is concerned about the potential abuse of his personal information, he might delay starting the transaction until he has the necessary information to assure himself that his information will not be abused by the service providers involved in the transaction.
- Large risk
  - If the transaction involves a large amount of money or risk, the customer may be unwilling to initiate the transaction until he has enough confidence that the transaction is legal and that the transaction is completed under an acceptable legal jurisdiction and there are procedures in place to contest the transaction if the transaction does not proceed successfully.
- Service providers involved in the transaction

- The customer may wish to know exactly who is involved in the transaction to make sure that there aren't any unscrupulous companies involved in the transaction.

The factors mentioned above have to be addressed by an assurance protocol to address customers' concerns that assurances be provided that the transaction will not have a negative impact on their expectation of the transaction. However, the level of protection sought by a customer does not alter the trust assurance protocol, but rather adjusts the requests for information that the customer creates. The client is responsible for determining a level of assurance that he feels is suitable and can then request the appropriate information from potential service providers, and any authorities or communities that can influence his decision.

3.2.4 Level of assurance

The level of assurance that the customer would require for a particular transaction is dependent on the risk involved in the transaction, the value or worth of the transaction, whether the customer trusts the primary service provider and the information the client has to disclose in the course of the transaction. In a transaction involving many service providers, third party service providers might require personal information about the client that the primary service provider does not need. For example, a car hire service might require the unique licence number of a customer or his exact age to complete a car insurance quote through a general insurance portal. If the primary service provider does not need the information that the customer provides, the customer should know who has access to the information and what they intend to do with it. The importance of full disclosure of information usage become more evident with the consumption of more sensitive data, such as a person's identity number or his credit card details. In a world where companies share information between each other to complete a transaction on behalf of a customer, the privacy policies of the primary service provider does not necessarily apply to any third party. It is up to the customer to investigate the privacy policies of all the services involved in the transaction.

Once the client has determined who is involved in the transaction, the client must be able to define a level of assurance at which he wants the transaction to proceed. The level of assurance refers to the assurances the client wants or needs before he will engage in a transaction. This level of assurance is subjective to the customer's concern for the privacy of his personal information and the risk involved in the transaction. The protocol presented in this dissertation allows a customer to receive a list of possible transaction parties, eliminate the services with privacy policies orthogonal to his preferences, and then evaluate the remaining companies to determine the best possible transaction path for the transaction. For example, a customer based in England might prefer to use a merchant bank for payment that is also based in England, because the client knows that merchant banks in England have strict rules regarding personal information. If the merchant bank does violate these conditions, the customer can use the local legal system as the merchant bank falls under the jurisdiction of the local courts. When a choice of a Nigerian or UK merchant exists for the customer to choose from, it is desirable to evaluate the two services on policies that suit the customer the best.

3.3 TRUST ASSURANCE PROTOCOL

A trust assurance level can only be determined once sufficient information has been collected from various parties about the subject of the evaluation. Information from the subject itself along with information that other people have of the subject and personal knowledge about the subject is required to properly evaluate the subject. Therefore, information collection is key to a protocol regarding the establishment of trust assurances. In a world where the subjects are companies offering goods or services for sale, the companies are bound by laws and industrial regulations to supply information about themselves and their activities. As seen in Section 2.7, ECTA stipulates clearly what information a company or individual providing goods or services in an electronic environment must provide. This legislative requirement provides a convenient base for the information required by a trust calculation service. In order to facilitate the transfer of information between parties, we have created a policy document containing the necessary information about a entity offering goods or services online. The next section describes the policy document created from the legal obligations that exist in legislature. The next section then introduces the protocol that allows a customer to request policies, disseminate them and find a suitable transaction path.

3.3.1 Policy document

The policy created essentially allows a company to concisely present information regarding its operation, and its use of a client's personal information in a standard way, so that the policy document can be distributed to other companies and potential customers. Creating a policy, rather than just querying a service provider on individual items, allows a client to both get all information and also query individual aspects of the company's operation as specified in the policy. The policy document also acts as a guard to ensure that a company presents all the information that it is legally obligated to provide and thus places no extra burden of the company. The policy allows a company to present and provide the information required from it in a way that is now globally accessible.

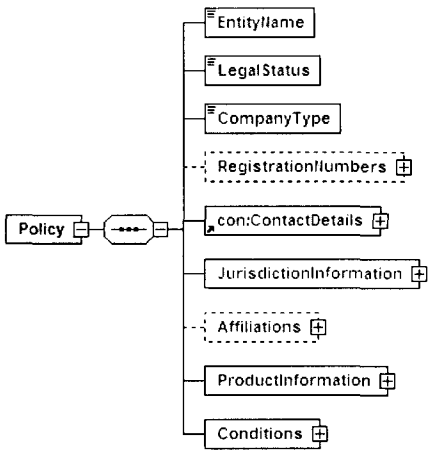

Figure 3.3: Overview of the policy XML schema

In Figure 3.3, the top-level structures for the policy document is shown. Below is a description of all the element presented in this figure.

Policy
The eveloping XML tag containing a single policy document.

EntityName
A text element representing the entity's name in plain English.

Legal Status
A text element describing the legal status of the company or indivdual, described by a vocabulary. The legal status of an entity can be one of the following:

- Individual
- Sole proprietor
- Partnership
- Close corporation
- Private limited company
- Public limited company
- Guarantee company.

The list of company types above represent the legal company types that are legally recognized (apart from "individual," which signifies no company type), and provides necessary information so that a customer can then check the requirements on that type of company to see if it complies with the regulations. The customer can also use this information to validate the company by querying various registries to see if the entity really exists as the type of company stated. The list of the different company types is fairly straightfoward, and does not require explanation of the different company types, except that a "Guarantee company" is the equivalent to a Section 21 company in South Africa. Companies registered under Section 21 are defined as "associations not for gain" and are non-profit organisations dealing with "promoting religion, arts, science, educaion, charity, recreation, or any other cultural or social activity or communal group interests." [1] This is the reason why a vocabulary is neccessary so that equivalent company structures can be identified. "Guarantee company" is the legal classification in different countries around the world and Section 21 company would have little meaning beyond the borders of South Africa.

CompanyType
CompanyType allows the entity to describe itself in terms of the International Classification Benchmark discussed in Section 3. The benchmark, combined with the company type and jurisdiction information under which the company operates, allows the customer to establish the exact requirements that the company operates under so he/she can then evaluate the company accordingly. By establishing the type of company by the sector of operation, the customer can

establish what self-regulatory bodies exist for the specific industry and can then evaluate the company further by querying the appropriate bodies involved in the specific sector. The ICB benchmark was dissussed previously along with the ontology created to represent the benchmark.

RegistrationNumbers
RegistrationNumbers is a complex XML element allowing a company to state its registration numbers, such as its VAT number and numbers assigned to it by other regulatory and authoritative bodies. It is shown in greater detail in Figure 3.4.

ContactDetails
ContactDetails is a complex XML element containing other XML elements to describe all the contact details for a particular company. Contact details is shown in depth in Figure 3.5 and discussed in detail later.

JurisdictionInformation
JurisdictionInformation is a complex XML element containing information about the jurisdictions under which a company operates. It is shown in greater detail in Figure 3.6.

Affiliations
Affiliations is an optional complex XML element allowing the company to state any affiliations that it has with other companies or with accreditation, regulatory or self-regulatory bodies. It is shown in greater detail in Figure 3.7.

ProductInformation
ProductInformation is a complex XML element for describing the goods or services that the company is offering for a particular transaction. The element is shown in greater detail in Figure 3.8.

Conditions
Lastly, Conditions is a complex element allowing the company or entity to state the conditions of using the service, their privacy policy and other information about the condition and terms of use. It is shown in greater detail in Figure 3.9.

By using the legal requirements for the information supplied in the policy, the policy is not adding any additional burden on the company involved. Rather, it is just formatting the information the company has to suppply to conform to legislation in a specific manner. In the following sections, the policy is described in greater detail to describe the complete usage of the policy document.

3.3.2 RegistrationNumbers

The RegistrationNumbers element shown in Figure 3.4 allows a company to list its registration numbers. An anbounded number of RegistrationDetails elements can be created so that an unlimited number of registration numbers can be listed. A description of a registration number can be provided, along the the number itself, and optionally, any authority associated with a

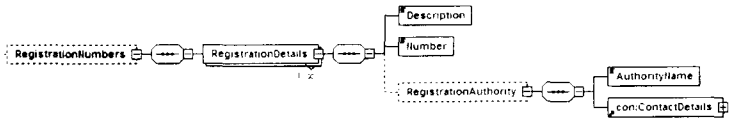

Figure 3.4: The RegistrationNumbers element of a policy

particular registration number can be listed.

3.3.3 **ContactDetails**

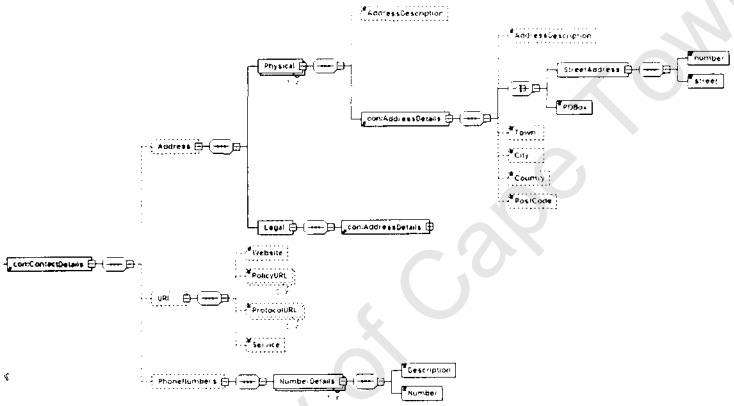

Figure 3.5: The ContactDetails element of a policy

The ContactDetails element of the policy element is a general XML structure developed to contain all the information needed to present the contact information of a service. The ContactDetails element is defined in a separate XML schema document and is included in the policy element schema.

The ContactDetails element contains elements for physical addresses, internet addresses or URIs and phone numbers. All the elements are optional because the ContactDetails element is reused in other sections of the policy schema definition and in other XML schemas. It is thus necessary to generalise the element to allow it to be reused without having to redefine the element for every situation.

The Address element contains two elements, namely physical and legal. The Physical address element is unbounded and multiple addresses can be defined. A physical address element contains an optional description of the address and an Address Details Element. Like the ContactDetails element, the AddressDetails element is also defined in a separate XML schema and is included into the policy schema for reuse purposes. The AddressDetails element contains all the elements necessary to fully describe a physical mail or PO BOX address. The legal address element also contains an AddressDetails element and is no different from the physical address element except that the address listed in the Legal element is the legal mailing address for the company or individual offering the goods or services. According to ECTA, any entity providing

goods or services in an electronic environment must explicitly state the address where the entity can receive legal documents. It is explicitly stated in the policy document.

The URI element lists the various electronic resources that are made available by the entity providing the service. The service can optionally list the website URI, a list of Policy endpoints that are applicable to the service and the transaction, endpoints to any protocols being used and the address for the service itself. Once again, the URI are all optional due to ContactDetails being defined externally, but a service must provide the URIs applicable to the service as stipulated in Section 43 of the ECTA.

The PhoneNumbers element allows a company to list several numbers so that the service may be contacted. Multiple numberDetails elements can be created in the PhoneNumbers element. The NumberDetails element can store numbers of any type, and thus the description field is necessary to notify the consumer what the number is for. The number could be an office telephone number, customer support number, fax number, pager number or any other telecommunications number.

3.3.4 **JurisdictionInformation**

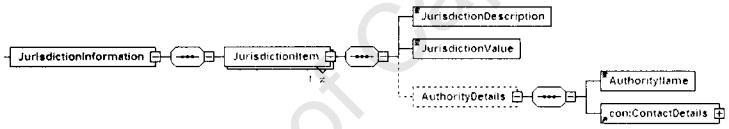

Figure 3.6: The JurisdictionInformation element of a policy

Jurisdiction information asserts in which jurisdiction the company is resident. As mentioned in Chapter 3, territorial jurisdiction could prove to be important in the choice of trading partner. If two parties reside in the same jurisdiction or in countries or states that have common jurisdiction rules, it might prove more convenient to choose such a service over a service where jurisdiction is a non-trivial issue, and where the client's jurisdiction has no personal jurisdiction over the company or individual to force them to appear before the local court. By choosing a compatible and suitable jurisdiction more assurance can be offered by the local governance structure.

The policy documents therefore expects a company to at least state what territorial jurisdiction it resides in, but also allows the entity to stipulate any other jurisdiction information that may be appropriate to disclose for the transaction.

The JurisdictionInformaiton element contains one or more JurisdictionItem elements. The JurisdictionItem contains three elements, namely JurisdictionDescription, JurisdictionValue and an optional element named AuthorityDetails. JurisdictionDescription is an element whose value has to be part of the jurisdiction vocabulary list defined in Chapter 3. This allows a common understanding of the type of jurisdiction being described. As mentioned before, the entity must at least provide one Jurisdiction item and it must contain a JurisdictionDescription for "territorial jurisdiction". Territorial description refers to the country under whose authority the company operates. The Value for the JurisdictionDescription is contained in the JurisdictrionValue element.

If the company willingly submits to the jurisdiction of an arbitration authority or self-regulatory body, details about the authority can be listed. There are two elements in AuthorityDetails, namely AuthorityName and ContactDetails. AuthorityName simply lists the name of the authority and ContactDetails contains the necessary contact information of the authority as described in Section 3.3.3.

3.3.5 **Affiliations**

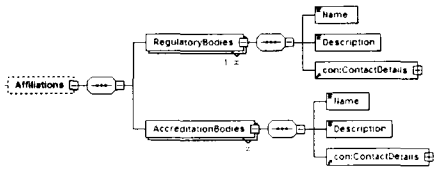

Figure 3.7: The Affiliations element of a policy

Affiliations is an optional element where an entity can list any regulatory bodies or accreditation bodies to which it subscribes. This is a requirement unber ECTA and if the company or individual subscribes to any such bodies, they must be listed here. The RegulatoryBodies and the AccreditationBodies elements share the same structre. They both contain a Name element, a Descritpion element, and a ContactDetails element. The name element simply provides the relevant body's name and the description provides a short textual description of the body. The ContactDetails element contains the necessary contact information of the body as described in Section 3.3.3.

3.3.6 **ProductInformation**

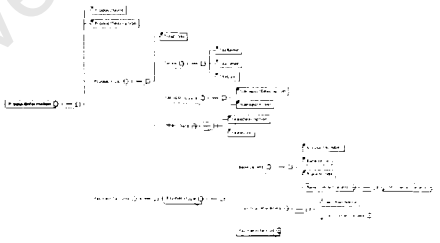

Figure 3.8: The ProductInformation element of a policy

The ProductInformation element contains details regarding the goods on sale or the service being offered. There are four elements within the ProductInformation element, namely, ProductName, ProductDescription, ProductCost, and PaymentOptions. Product name is a simple text element containing the name of the product or service. Similary, ProductDescription is a text element containing a description of the goods or services. Product cost outlines the product's total cost and also breaks the costs down into the applicable taxes, transport costs and any other costs as stipulated in ECTA. The TotalCost element simply contains the total cost of the goods or service. The Taxes element can be used multiple times to describe different taxes and contains elements to describe the name of the tax, the value of the taxation and the actual cost

of the tax. Similary, transport coses can be used multiple times to describe different transport costs and contains elements to describe the transport and the cost of the transport. Other costs is an element that allows the company or individual to describe any miscellaneous costs, by allowing the company to describe the cost and to attach the cost of the extra fee, by using the element FeesDescription and FeesCost respectively. The OtherCosts element is optional and may be used any number of times.

3.3.7 **Conditions**

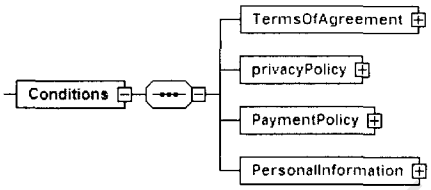

Figure 3.9: The Conditions element of a policy

The Conditions element outlines the service's policies, guarantees, warrantees and many other information items. The Conditions element also outlines what a service does with personal information that it collects and states it in a clear manner so that the customer in the transaction knows what the service claims it will do with his personal information. The Condition contains four Elements: TermsofAgreement, PrivacyPolicy, PaymentPolicy and PersonalInformation. The TermsOfAgreement element is shown in Figures 3.10 and 3.11. The PrivacyPolicy and PaymentPolicy Elements are shown in Figure 3.12 and lastly, the PersonalInformation element is shown in Figures 3.13 and 3.14.

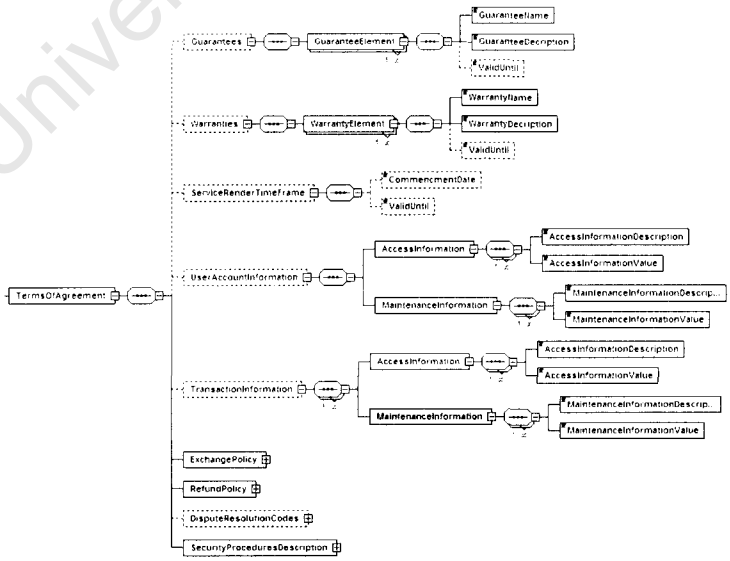

Figure 3.10: The top half of the TermsOfAgreement element of the conditions element

The TermsOfAgreement outlines different terms offered to the customer in the transaction.

These include guarantees, warranties, exchange policies, refund policies, the time in which the goods will be dispatched after the sale or the commencement date of the services, action the customer can take if a dispute arises, the security procedures in place to protect the transaction and the parties involved in the transaction.

The Guarantees element can contain an unlimited number of GauranteeElements which consist of the name of the guarantee element, a general description of the guarantee offered, and optionally a ValidUntil element to indicate a time frame in respect of the guarantee element. The Warranties element is structured in the same manner as the Guarantees element, and does not require additional explanation.

The ServiceRenderTimeFrame is an optional element describing the time when the goods will be dispatched or the services will begin. It provides two optional sub-elements, namely, CommencementDate and ValidUntil. CommenementDate refers to a date when the service will start or when the goods will be dispatched. The ValidUntil element is a date element that states when a service will terminate. The company can use either or both elements to specify the duration of the service or only one to specify when a service will dispatch the goods.

The UserAccountInformation element specifies how a customer can access and maintain the details of his account details. The AccessInformation element contains elements that allows the entity to describe the access information element, along with the AccesInformationValue element, which contains a URI to the appropriate service endpoint so that the customer may access his information. This could be a second Web service, or it could be a website where a customer can easily access his user account information.

Similary, the MaintenanceInformation element describes the manner in which a customer may modify his user account details and functions exactly like the AccessInformation element, except that the elements contained in the MaintenanceInformaiton element are called MaintenanceInformationDescription and MaintenanceInformationValue.

The Transaction element specifies how a customer can access and maintain the details of the transactions. The AccessInformation element contains elements that allow the entity to describe the access information element, along with the AccesInformationValue element, which contains a URI to the appropriate service endpoint so that the customer may access his information. This could either be a Web service, or it could be a website where a customer can easily access the transaction information.

Similary, the MaintenanceInformation element describes the manner in which a customer may modify the appropriate transaction details and functions exactly like the AccessInformation element, excepts that the elements contained in the MaintenanceInformaiton element are called MaintenanceInformationDescription and MaintenanceINformationValue.

Access to and modification of information sent in an electronic transaction are required by ECTA and thus the service provider must provide the necessary controls for such activities.

The ExchangePolicy Element allows the company or service provider to list the different policies regarding product exchanges. An unbounded number of ExchangePolicyElements can be

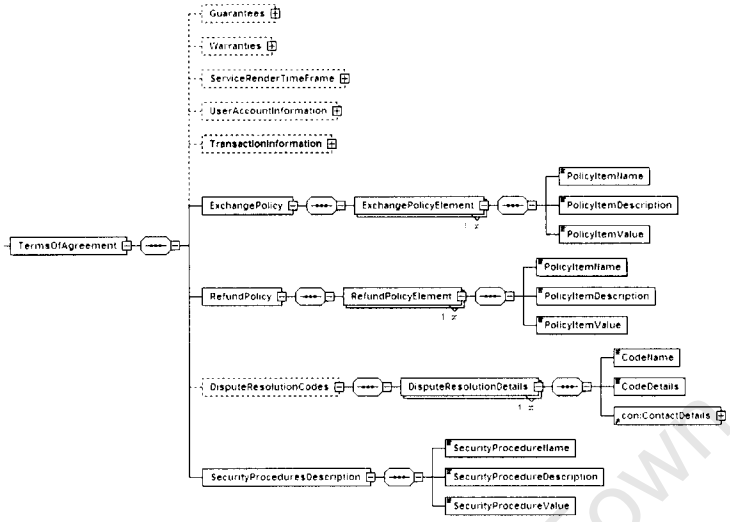

Figure 3.11: The bottom half of the TermsOfAgreement element of the conditions element

created to specify the exchange policy. The ExchangePolicyElement element contains the elements for naming the policy, describing the policy and assigning a value to the policy. The RefundPolicy element similarly allows the service provider to state the policies regarding refunds and is structured in exactly the same manner as the ExchangePolicy element.

The DisputeResolutionCodes element allows the service provider to stipulate any additional dispute resolution codes that are avalable to the consumer if the need arises from the transaction. Details of additional resolution codes are provided in a DisputeResolutionDetails element which contains elements for the name of the dispute resolution codes, an element describing the details of the code, and contact details for the code, as explained in Section 3.3.3.

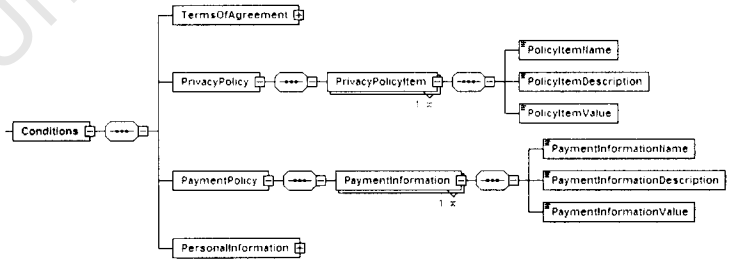

Figure 3.12: The PrivacyPolicy and the PaymentPolicy elements of the Conditions element

The PrivacyPolicy element allows a service provider to assert the company's privacy policy. Privacy policy conditions are stored in PrivacyPolicyItems and contain the name of the privacy policy item, a description and a value. These are kept in the PolicyItemName, PolicyItemDescription and PolicyItemValue respectively.

The PaymentPolicy element silimarly describes the policies of the service provider by allowing

it to specify the policies regarding policies. Like the PrivacyPolicy element, the PaymentInformation item is contained in three elements, namely PaymentInformationName, PaymentInformationDescription and PaymentInformationValue.

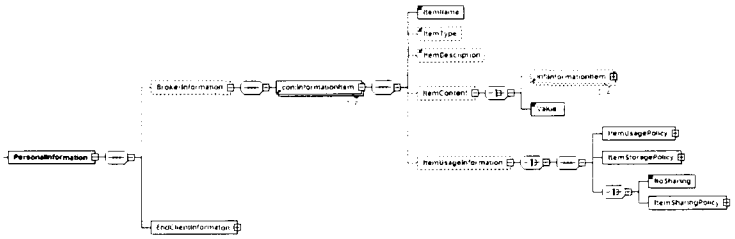

Figure 3.13: The PersonalInformation element of the conditions element

The personalInformation element is a crucial part of the policy for the purposes of determining whether or not a service is suitable for a customer. This element describes the information that the service requires and allows the service provider to state explicitly what the information will be used for. This is a requirement of ECTA. The PersonalInformation element contains two elements, namely BrokerInformation and EndClientInformation. BrokerInformation is an optional element that is used if a service is invoked to perform a service on behalf of a customer via another service provider. Thus the information required by the service provider consists of the information applicable to the broker service. The service might also need information from the end user, such as their credit card details to perform a credit check if it is what a service is required to do.

The BrokerInformation element consists of one or more InformationItem elements. The InformationItem element contains elements to describe name, type, description and optionally, the content of the information item and and usage information.

The ItemContent element consists of either a value or one or more InformationItem elements. Thus complex information items can be built up if necessary.

The ItemUsageInformation element describes a services intention regarding a particular information item. It contains the ItemUsagePolicy element, ItemStoragePolicy element and either a choice of explicitly stating that no sharing of the information item will take place or explicitly stating the sharing policy.

The ItemUsagePolicy can contain one or more PolicyItems, which contains three elements, namely, PolicyItemName, PolicyItemDescription and PolicyItemValue. The PolicyItem structure is identical to the PrivacyPolicyItem structure described above.

The ItemStoragePolicy element is identical to the ItemUsagePolicy in structure, the only difference being the element name.

The mutually exclusive choice of no sharing or the ItemSharingPolicy allows the service provider to state his actions. If the NoSharing element is used, then the ItemSharingPolicy cannot be used and vice versa. The ItemSharingPolicy element contains two unbounded and optional elements,

OtherCompany and PolicyItem. The OtherCompany element describes the entity with whom the information will be shared. It contains the CompanyName element for the company's name, the CompanyType element to describe the type of the company, the ContactDetails element to describe the contact details of the company or entity as described in Section 3.3.3 and the CompanyAffiliation element to describe the affiliation or relationship between the service provider and the company with whom the information would be shared.

All the information described in the BrokerInformation element is only information concerning a service that invokes a service on behalf of a end user. If the service requires end user information, then the EndClientInformation element would have to be populated with the appropriate information.

The EndClientInformation element describes the policies concerning the collection, storage, dissemination, and sharing of a end user's personally identifiable information. The element contains two children elements, InformationItems and PII. The InformationItems element simply contains a list of ItemName elements and merely lists the personal information that the service provider requires. The values for the ItemName element are contained in a vocabulary of personal information so that the names can be standardized and the information items can be listed easily. The vocabulary is listed in Section **********.

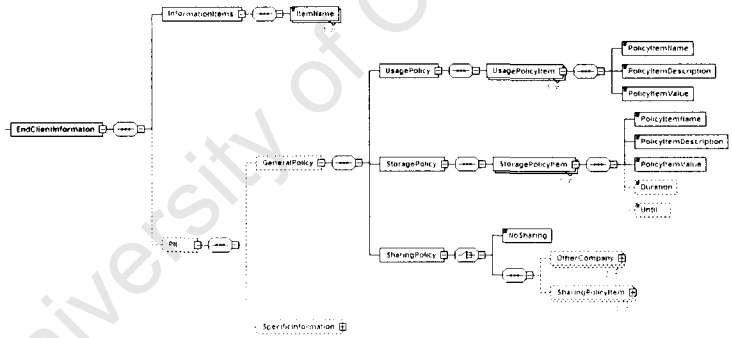

Figure 3.14: The top half of EndClientInformation element of the PersonalInformation element

The second element in EndClientInformation is the personally identifying information or PII element. The PII element contains two optional elements, GeneralPolicy and SpecificInformation. The GeneralPolicy element describes the usage, storage and sharing policy of items not specified in the SpecificInformation element.

The UsagePolicy allows the service provider to list an unbounded number of UsagePolicyItem elements to describe various policies regarding the use of personally identifying information. The UsagePolicyItem contains three elements, PolicyItemName, PolicyItemDescription and PolicyItemValue. These policy elements have been discussed previously.

The StoragePolicy element contains StoragePolicyItem elements and describes the policies regarding storage of personal information. It has the same structure as the UsagePolicyItem element, with the addition of two optional elements, Duration and Until. These element are of a date type and can store a date stating how long the information will be kept. This is in

compliance with ECTA's provision for the protection of PII.

The SharingPolicy outlines the sharing policies of the service provider as it applies to personal information collected via electronic means. The element has the same structure as the sharing element structure described in the ItemUsageInformation element.

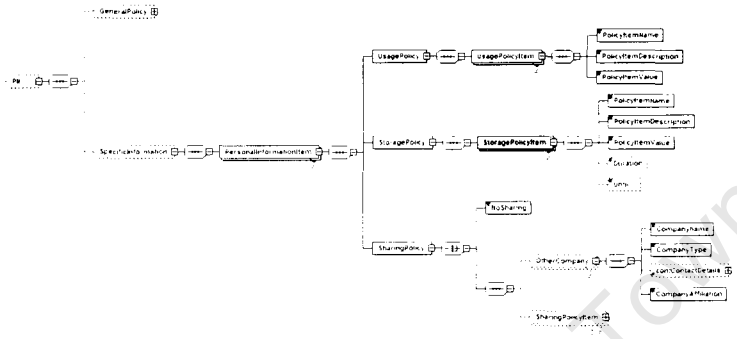

Figure 3.15: The bottom half of EndClientInformation element of the PersonalInformation element

The SpecificInformation element contains information about individual pieces of PII collected that are handled in a different manner to the general policies of PII. If an information item is defined in the specific information, the general policies regarding PII are discarded and replaced by the information supplied by the PersonalInformationItem element. The structure of the PersonalInformationItem element is identical to the structure of the GeneralPolicy element. A PersonalInformationItem element can be created for each information item that deviates from the general policy for PII.

3.3.8 **PolicyDocument**

In the previous section, we describe the schema for creating a policy. Here is a valid policy document based on the schema defined above.

```
<?xml version="1.0" encoding="UTF-8"?> <PartnerRequest
xmlns:inf="http://dna.cs.uct.ac.za/TrustAssurance"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="C:\masters\masters\schema\PartnersRequest.xsd">
<ServicePartners>
<ServicesRequestType>AllServices</ServicesRequestType>
<Restrictions> <InformationItemRestrictions> <All>
    <ExactlyOne>
        <RestrictionCondition>
            <ItemName>CreditCardDetails</ItemName>
            <RestrictionName>Storage</RestrictionName>
            <RequiredValue>EncryptedStorage</RequiredValue>
        </RestrictionCondition>
        <RestrictionCondition>
            <ItemName>CreditCardDetails</ItemName>
            <RestrictionName>Storage</RestrictionName>
            <RequiredValue>NOStorage</RequiredValue>
```

```
        </RestrictionCondition>
        <RestrictionCondition>
            <ItemName>CreditCardDetails</ItemName>
            <RestrictionName>Storage</RestrictionName>
            <RequiredValue>LegalComplianceStorage</RequiredValue>
        </RestrictionCondition>
    </ExactlyOne>
    <RestrictionCondition>
    <ItemName>FullName</ItemName>
        <RestrictionName>Sharing</RestrictionName>
        <RequiredValue>NoSharing</RequiredValue>
    </RestrictionCondition>
    </All>
</InformationItemRestrictions> <ServiceRestrictions>
    <All>
            <RestrictionCondition>
            <RestrictionName>Jurisdiction</RestrictionName>
            <RequiredValue>South Africa</RequiredValue>
        </RestrictionCondition>
    </All>
</ServiceRestrictions> </Restrictions> </ServicePartners>
</PartnerRequest>
```


3.4 TRUST CALCULATION

3.4.1 Composite web services transaction chains

Web services transaction chains are links between different service providers to create an environment in which a customer's request to conduct a transaction can be dealt with. The transaction is conducted between the customer and the primary service provider, but also other services that participate in the transaction to add value-added services or provide some of the services that the primary service provider offers but cannot perform itself. The primary service provider is still the primary contact for the customer and responsible for the success of the transaction, but if the customer's personal information is passed to the participating service providers, it poses a threat to the customer by exposing potentially sensitive information to companies and services that the customer may not even be aware of. These third party services do not have to comply with the privacy policies that are used by the primary service provider and the customer is left vulnerable because he cannot control what happens with his information.

Value-added services or services that the primary service provider outsources, may require some of the customer's personal information in order to conduct the operation. For example, a travel agency portal, acting as the primary service, may enlist the use of a car-hire company to provide a seamless car-hire facility in their portal. The car-hire company may require personal information from the customer in order to provide pricing information. The car-hire company might have a policy of sharing or selling the information it collects to information harvesters. A customer should be able to determine what will happen with his PII before the transaction is started or at least know the intentions of all the parties involved.

As an illustration, suppose that service 2 and service 4 in Figure 3.16 are both car hire com-

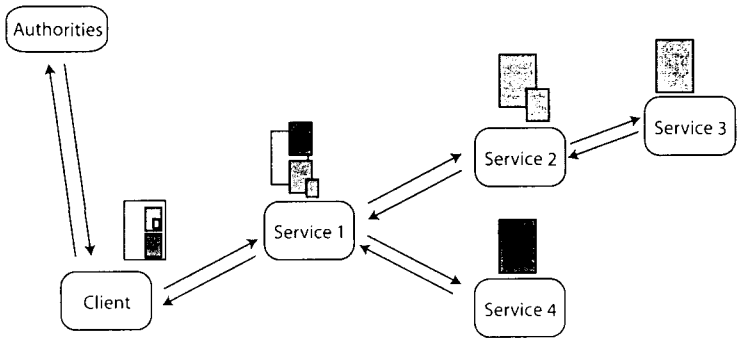

Figure 3.16: Composite Web service chain

panies that the primary service provider deals with. Both companies provide the same service, but service 2 uses service 3 to complete his service offering. The customer should be able to decide whether to use service 2 and service 3, or use service 4 to conduct the transaction. The customer might be willing to pay a premium to protect his PII and if service 3 does not have an adequate privacy policy, the customer might prefer to use service 4 as opposed to the transaction path that includes service 2 and 3, even though service 4 might be more expensive. This concept of chaining services together to provide the end user will all the information concerning a transaction is the cornerstone of the trust assurance protocol. By allowing the client access to the transaction path and to information relating to the services involved in a transaction, the client has the ability to become involved in the transaction. Scenarios created in Chapter 5 will use this model as a base from which to evaluate whether or not assurances can be presented to the client by using the trust assurance protocol.

The scenario depicted in Figure 3.16 presents a challenging stumbling block to the adoption of Web services for transactions that involve any amount of risk. If customers could choose a particular transaction path over a more risky path, it would reduce the entire risk in the transaction. The full disclosure of the services involved in the transaction might provide enough assurances to the customer, but in transactions that present a lot of risk to the customer, the customer may require greater assurances. A privacy conscious customer may also not engage in a transaction until he can be guaranteed what actions will be taken with his PII.

Transaction Path Elicitation

To establish what services or entities could be involved in a transaction, a protocol is needed to establish whether or not there is a suitable transaction path, and if there are multiple suitable paths, to find the best path to suit the customer.

In the scenario depicted in Figure 3.16, the primary service provider aggregates policy and operational information from the participating service providers and sends all the collected information in one message to the customer. Each service provider is responsible for collecting the appropriate information from the services that it directly contacts to complete the transaction. This forms a recursive protocol allowing all the possible transaction paths to be sent to the customer. Once the transaction path has been established, the transaction can proceed us-

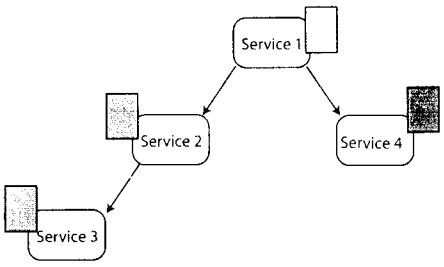

Figure 3.17: Tree representation derived from policy collection

ing other Web services protocols. The trust assurance protocol only operates to find a suitable transaction path, and has nothing to do with the actual transaction.

Apart from offering the customer primitive assurances, such as authentication, the protocol provides mechanisms in which the customer can state restrictions on what services may and may not do with his personal information, and he may also specify restrictions about the services themselves, such as jurisdiction information. For example, the customer can choose the most suitable transaction path by only choosing transaction paths that involve service providers from his country (if possible) or only use services that are certified as ISO 9001 compliant. As an illustration, a customer might be willing to pay more for a service that resides in the same legal jurisdiction as he/she does than a service offering a similar service that operates in a jurisdiction where the customer has no recourse should the transaction fail. In particular, services and customers residing in the European Union may not transfer information to services in other countries where there are not adequate privacy protection laws. This legislative restriction would force the customer to only choose services in countries that have adequate privacy protection legislation or countries with explicit agreements with the European Union, such as the United States, and then only with the companies that comply with the legal amendments in the Safe Harbor Act[48].

The information that the customer can use on which to base his decision must be contained in the policy document of a service provider. This provides a suitable base of mandatory information that service providers have to provide to the customer if requested. Whilst this may seem to be a significant assumption or requirement for the policy, the information contained in the policy is information that is required by South African and comparable electronic commerce laws, such as the European council directive 2002/58/EC to all member nations. These laws require service providers to make available all the information required to the customer of electronic transactions. There should thus be no reason why the service provider wouldn't be willing or able to provide the information requested. ECTA clearly stipulates what information a company has to provide if the service is conducted electronically.

In Figure 3.17 a tree diagram of the possible transaction paths is shown from the scenario depicted in Figure 3.16. The tree representation is useful as each path to an end-node represents a complete transaction chain that will function together to complete the transaction. Thus the client has to choose one of the paths in the tree in order to start the transaction. The client knows what services are involved in the transaction, and now has the ability to choose the

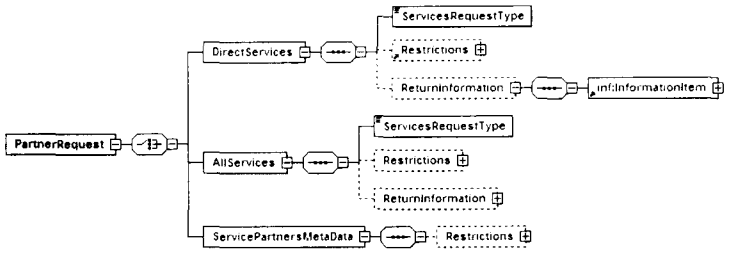

Figure 3.18: The PartnerRequest XML element schema decription

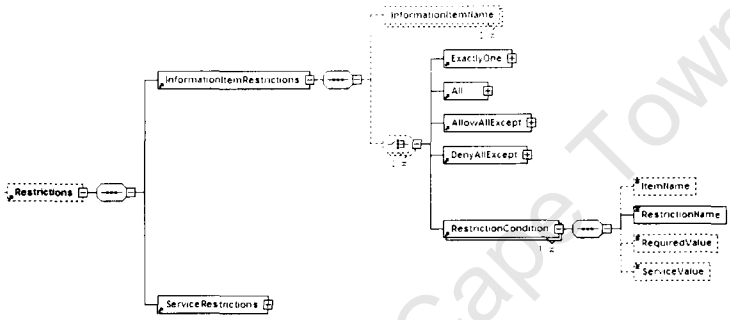

Figure 3.19: The Restrictions XML element schema decription

transaction path that is most suitable to his requirements. These requirements could the cost, privacy, quality of service or transaction jurisdiction.

Transaction path elicitation protocol

The root element of the transaction path elicitation protocol is the PartnerRequest element as shown in Figure 3.18. The PartnerRequest element can contain one of three XML elements, DirectServices, AllServices or ServicePartnersMetadata. DirectServices and AllServices are control elements that either request just the services that the service provider is directly in contact with or request all posible transaction paths, down to the last end node service. The DirectServices element contains a ServicesRequestType element which is a control element to instruct the service provider to present the path that best fits the customer's restrictions or to send back all valid options for the client to evaluate.

The Restrictions element allows the customer to specify exactly what he will and will not allow to happen to any information that he sends in the transaction. These restrictions will eliminate services based on their policies and the primary service provider will send back a list of suitable transaction paths. The restrictions element is shown in Figure 3.19. The optional ReturnInformation element allows the customer to specify what information he would like to receive from each service involved in the transaction. The client could request that each service's entire policy be sent to him. This is the default action if the client does not specify what information he would like. The customer could request that only the services' URLs are returned to him or that their fullname, their territorial jurisdiction and their Public certificates should be returned.

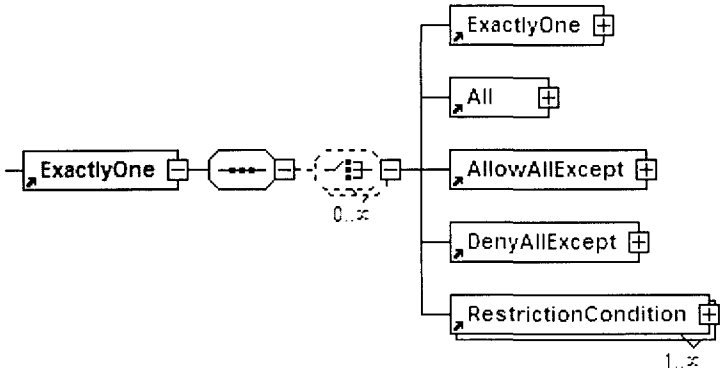

Figure 3.20: The ExactlyOne XML element schema decription

The AllServices element has the exact same structure of the DirectServices element and does not require extra explanation. The InformationItem element was described in Section 3.3.7 of this Chapter.

The ServicePartnersMetaData element allows the user to request information about the entire transaction path. The customer does not have to specify any restrictions or any other information, but can do so it he/she wishes to. The return element only lists the available transaction paths along with the contact URI of all the services, so that the customer can contact the services to inspect their policies if he desires to do so.

The Restrictions element shown in Figure 3.19 allows a customer to state the restrictions that service providers have to adhere to in order to qualify to the transaction. The Restrictions element specifies two different restriction conditions; restrictions that apply to the customers' personal information and restrictions that apply to the service itself. The Rescritions element uses the InformationItemRestrictions and the ServiceRestrcitions elements respectively to specifiy the restrictions. The structure of the elements are the same, and so only the InformationItemRestrictionsElement will be discussed.

The InformationItemRestrictions element contains an element to state the specific information item element that this restriction appies to. The element's name is InformationItemName. The InformationItemRestrictions element then allows a choice of elements, namely ExactlyOne, All, AllowAllExcept, DenyAllExcept and Restriction Condition. The details of the ExactlyOne element is shown is Figure 3.21. The ExactlyOne element has the exact same structure of the elemens All, AllowAllExcept and DenyAllExcept.

By defining the elements recursively, complex restrictions and equivalence conditions can be created from these elements. The recursive nature is akin to the structure of the policy assertion mechanism found in WS-Policy [44]. RestrictionConditions can be stated in the restrictions conditions to state the customer's preferences. The element contains the name of the restriction, and optionally the information item name, requiredValue of the restriction and a serviceValue element. The serviceValue element is a element that is used to report discrepancies between the required values and the values that a service provider has.

An example PartnerRequest element is shown below:

```
<?xml version="1.0" encoding="UTF-8"?> <PartnerRequest
xmlns:inf="http://dna.cs.uct.ac.za/TrustAssurance"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="C:\masters\masters\schema\PartnersRequest.xsd">
<DirectServices>
<ServicesRequestType>AllServices</ServicesRequestType>
<Restrictions> <InformationItemRestrictions>
    <All>
        <RestrictionCondition>
            <ItemName>FinancialData</ItemName>
            <RestrictionName>storage</RestrictionName>
            <RequiredValue>EncryptedStorage</RequiredValue>
        </RestrictionCondition>
        <RestrictionCondition>
            <ItemName>FinancialData</ItemName>
            <RestrictionName>sharing</RestrictionName>
            <RequiredValue>NoSharing</RequiredValue>
        </RestrictionCondition>
        <RestrictionCondition>
            <ItemName>FullName</ItemName>
            <RestrictionName>sharing</RestrictionName>
            <RequiredValue>NoSharing</RequiredValue>
        </RestrictionCondition>
    </All>
    </InformationItemRestrictions> <ServiceRestrictions>
    <AllowAllExcept>
        <RestrictionCondition>
            <RestrictionName>TerritorialJurisdiction</RestrictionName>
            <RequiredValue>Nigeria</RequiredValue>
        </RestrictionCondition>
    </AllowAllExcept>
</ServiceRestrictions>
</Restrictions>
</DirectServices>
</PartnerRequest>
```

The PartnerRequest element shown above states that a service provider has to securely store any information items containing FinancialData and that the service may not share any Financial-Data or the users full name. The customer also places a restriction on the service by stating that the service can be in any territorial jurisdiction other than in Nigeria. This simplified condition shows the descriptive power of the Restrictions element.

Transaction path elicitation protocol results

Once a PartnerRequest element has been sent to a primary service provider, the result needs to be sent back to the customer. This is done via the PartnersRequestResponeList element shown

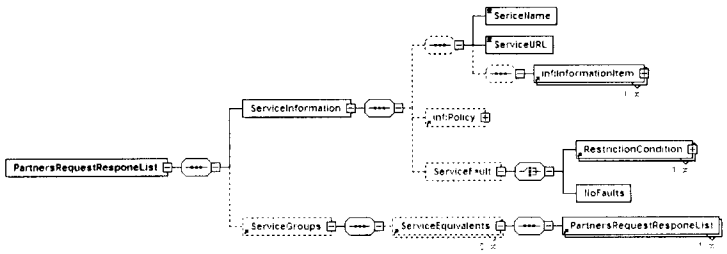

Figure 3.21: The ParnersRequestRepsonseList XML element schema decription

in Figure 3.21. The element consists of one mandatory element, ServiceInformation, and an optional element, serviceGroups.

The ServiceInformation element contains information about the service that created the PartnersRequestResponseList. The service states its name and URL in the ServiceName and ServiceURL elements respectively and can optionally add information items that the end user requested. The Service can also add its policy to the ServiceInformationElement. The ServiceInformation element contains another element, namely the ServiceFault element. If the service fails any restrictions set out by the end user, the service can state the reasons for the failure in the ServiceFault element. The ServiceFault element either contains a list of RescrictionCondition elements as described earlier or it can assert that NoFaults were recorded.

The PartnersRequestResponseList also contains the ServiceGroups element. If a service links to external services these services are defined in the ServiceGroups element. If there are two or more services that can perform the same function for the service provider, these elements can be stated in a serviceEquivalents element. Each ServiceEquivalents item represents a list of services from which a customer must choose one. The ServiceGroups element can contain any number of ServiceEquivalents elements. The ServiceEquivalents item consists of an unbounded number of PartnersRequestResponseList item, making it a recursive element.

Effectively, the primary service provider receives a PartnersRequest element and sends the restrictions to each of the services that it interacts with. If the service fails to meet the criteria, it sends an ErrorFault message. If it passes all the restrictions, then that service provider sends the restrictions to its external transaction partners. Each service in the transaction chain then responds with a PartnersRequestResponseList element, and all these element are presented to the customer in one PartnerRequestResponseList. This document contains all the possible transaction paths in a manner that the customer can easily evaluate.

3.4.2 **Trust parameter evaluation protocol**

Once a customer has received all the possible suitable transaction paths, the customer needs to be able to determine the trustworthiness of the service. The trust model described earlier in this chapter requires information to be sourced from various accreditation bodies, verification bodies, industry watchdogs, industy associations, and communities where agents can share and request information about service providers. A protocol is needed to request these information items so that the customer can obtain all the information required to assess the services and the

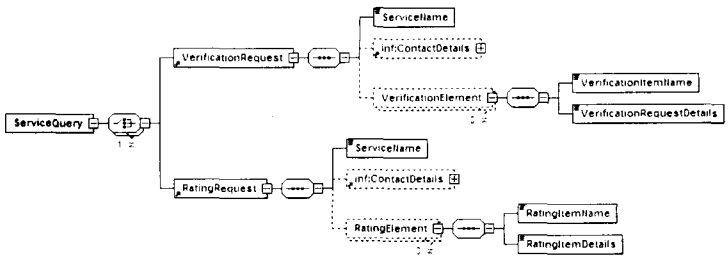

Figure 3.22: The ServiceQuery XML element schema decription

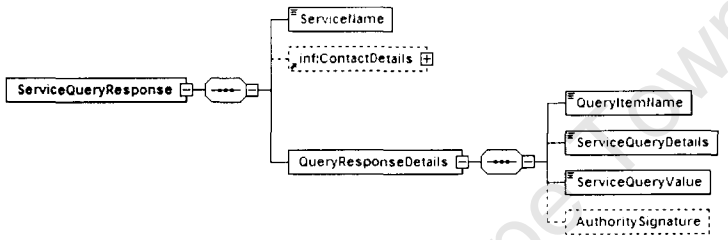

Figure 3.23: The ServiceQueryRespone XML element schema decription

transaction paths.

To issue a request to an authority of community end point, a ServiceQuery element is used. The ServiceQuery element is shown is Figure 3.22.

The ServiceQuery element allows the customer to make a boundless number of VerificationRequest and RatingRequest elements. The VerificationRequest element contains a ServiceName element to describe the service in question, a ContactDetails element to specify the ContactDetails information of the service in question and an optional VerificationElement which allows the customer to specify specific verification queries. The RatingRequest element is structured in the same manner as the VerificationRequest element, except that the VerificationElement element is replaced by the RatingElement element.

The figure shown in Figure 3.23 is the schema definition of the response to a ServiceQuery element. The ServiceQueryResponse element allows an authority or a community endpoint to assert its verification details or its rating of a service. The element consists of a ServiceName element, an optional ContactDetails element and a QueryResponseDetails element. The ServiceName element identifies the service about which the verification or rating is about. It can contain an unbounded numer of QueryResponseItem elements so that the authority can make assertions about numerous verification or rating queries, or the QueryResponsedatails element can contain an FaultMessage element, describing any errors that may have occured in the verification or rating process. The FaultItem element contains one or more FaultItem elements and describes the query name in the QueryItemName element, the details of the fault in the FaultDetails element, and optionally, the fault value in the FaultValue element as well as an optional AuthoritySignature element to sign the results of the verification or rating.

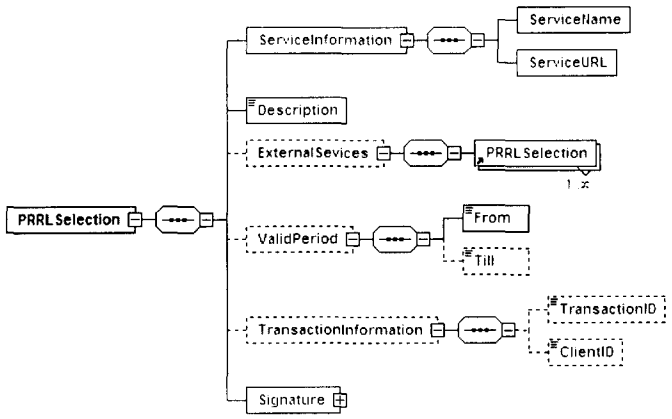

Figure 3.24: The PRRLSelection XML element schema decription

The QueryResonseItem element consists of the QueryItemName element, to name the particular query item, a ServiceQueryDetailselement to specify any details about the verification or rating, a ServiceQueryValue element to state the result of the verification or the rating value and lastly, an optional AuthoritySignature element to sign the value to ensure that the result of the query is not tampered with.

3.5 CONTRACT

Once the customer has requested the available services using the trust assurance protocol, and has calculated the most suitable transaction path for him/her, the client must notify the PSP as to which transaction chain the PSP must use in order to process the transaction. This file can then be propagated through to the other services involved so that they too can know which services they should use to complete their part of the transaction. The schema for the PRRLSelection element is shown in Figure 3.24.

In Figure 3.24, the PRRLSelection element contains the information about the service which the client wishes to use. This includes the service's URL and its name. The PRRLSelection element contains a description element so that a simple textual description of the agreement between the client and the services can be stated so that the signed PRRLSelection element can be binded to its intented use. The PRRL element also contains an optional XML element for the external services the service will use in the transaction. The client received the options for each service in the trust assurance protocol's PRRL element. The ExternalServices element contains an unbounded number of PRRLSelection elements, allowing client to create a PRRLS-election element for each service and embed it in another PRRLSelection element so that the transaction chain is stored. Additionally, the PRRLSelection element contains the ValidPeriod, TransactionInformaton and Signature elements.

The ValidPeriod contains to date elements allowing the client to specify the time during which the PRRLSelection message is valid and the TransactionInformation element allows the client to either specify his/her name, or a unique transaction identity number, so that services know

which services to use when they receive a transaction with the particular identification.

The signature element encapsulates a WS-Security signature hash and uses the WS-Security specification to include the signature hash in the PRRLSelection message itself. This forces each service to acknowledge the external services involved because they then sign signed PRRLSelection elements received from the services that they will use in the transaction. When the client receives this PRRLSelection element back from the PSP, the transaction path will be signed by all parties involved and will provide the client with the assurance that all parties in the transaction are in agreement about how the transaction will proceed. The client can verify the services by computing a hash of the PRRLSelection element and computing the signature using each service's public key. Because the PRRLSelection elements contains the PRRLSelection elements from other services, each service signs his own as well as any services that it uses's PRRLSelection element, ensuring that a chain of services is maintained.

CHAPTER 4

IMPLEMENTATION

4.1 INTRODUCTION

In this chapter, the following implementation artifacts, developed to provide a transaction assurance framework for Web services, are discussed:

- transaction execution framework
- implementation of the trust assurance protocol
- implementation of Web services
- client prototype
- legal ontology
- Industry Classification Benchmark (ICB) ontology.

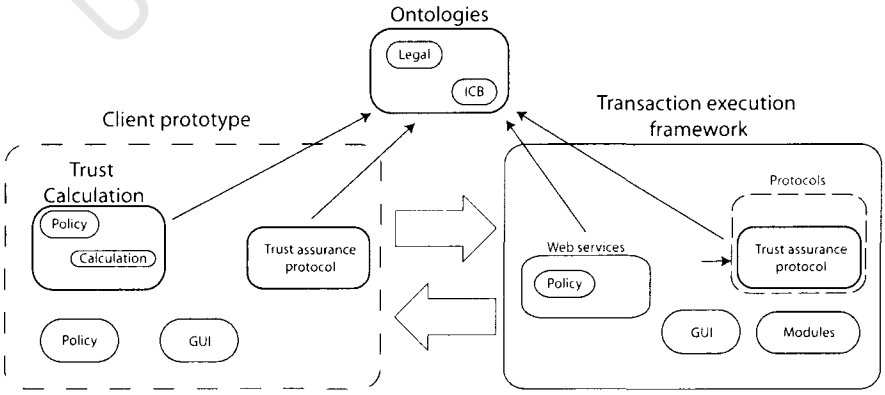

Figure 4.1: Implementation architecture

Figure 4.1 shows how the components connect to provide an environment in which Web services can be simulated and in which the trust assurance protocol can be evaluated. The client

prototype consists of a client side implementation of the trust assurance protocol and makes use of the ontologies to create standardized policies so that the trust assurance protocol works efficiently. The client creates policy documents to state his/her preferences. The client prototype then initiates the trust assurance protocol by engaging Web services. The Web service is deployed in the transaction execution framework and uses its policy and connected services to provide suitable information to the client of the transaction. Each of these items is discussed in this chapter.

The implementation of the artifacts described above serve two purposes. Firstly, evaluating the protocol in the context of different scenarios produce results that can be evaluated to determine whether the protocol is producing results consistent with expected results. The need to evaluate the protocol based on a qualitative scale stems from the fact that the trust assurance protocol is elegant in its simplicity. There is no scope for a dead lock further than a service not responding, whether due to network conditions or malicious activity. The protocol is designed not to require complex interaction between the services interacting in the transaction assurance step. The protocol however, does produce quite complex results and can be measured in two ways. The initial trust assurance protocol step to find suitable transaction paths can be evaluated by comparing the restrictions set by the user against the policies that are provided by the services. This step can be evaluated objectively because the customer's restrictions and the service's policy are both based on the same vocabulary, which in turn, is based on applicable legislation. The second aspect of evaluation falls on how the customer chooses the most suitable transaction path if there is more than one available option. This step is inherently subjective because the decision of choosing an appropriate transaction path from the available suitable transaction paths, depends solely on the desires of the client. Most customer's might be perfectly happy with using any of the suitable transaction paths, and may use other quality of services factors to choose which transaction path to choose. A more discerning customer may choose to determine the best transaction path on trust values determined by the customer. The trust evaluation part of the protocol, described in the previous chapter, was specifically designed to be flexible, and effectively independent of the protocol. Because there is no concise definition of trust, no one measure of trust can be considered better than any other. Thus only experience using online services and recommendations from others can guide a customer to choose a suitable trust evaluation method for him/her. The trust evaluation method described in this dissertation has an over-arching architecture that allows the customer to determine a level of trust using any trust system.

The Trust Assurance Protocol described in Chapter 3.3 needs to be implemented to verify that it produces results consistent with the requirements it set out to achieve. The artifacts produced for this dissertation is also applicable to similar research areas beyond the scope of this dissertation. For example, the transaction execution framework and the ontologies presented below can be used independently of the trust assurance protocol and are thus artifacts in their own right. All the artifacts created for this dissertation are discussed in subsequent Sections of this Chapter.

4.2 TRANSACTION EXECUTION FRAMEWORK

For the purposes of this investigation, the transaction execution framework (TEF) was created to provide an environment wherein scenarios using the trust assurance protocol could be simulated

and evaluated. TEF allows the user to create and deploy services easily so that the messages being passed between all the parties involved in transaction can be monitored. The framework's main objectives are to run Web services, monitor those Web services, report on the activities and results of the Web services and to provide a visual interface to configure the environment.

The execution framework is a modular java application that can be used to simulate any Web service. By only concerning itself with the messages sent to and received from services, TEF treats Web services as black boxes. TEF allows Web services to remain autonomous software artifacts that can be implemented in any way. The only requirement that TEF places on Web services is a message passing API so that the user of TEF can initialise, start, stop, configure and run the Web services. Another design goal of TEF was to allow both actual Web services deployed in a Web services container and instances of the business logic classes to be deployed in the framework. The reason to local java objects to act like a Web service was to speed up the creation of Web services to be used in the framework. By using local java objects, other benefits are also provided for the user of TEF. A few examples are that the Web services and their interactions can be monitored by a debugger, so that the development of Web services can be sped up. Another benefit is that scenarios can be saved and loaded without concern that a remote server may be down or not configured to handle the services that the user might want to use. The different execution methods that are used in TEF are described below.

4.2.1 Execution paradigms

In TEF, multiple execution methods can be used. TEF can monitor axis2 Web services, Web services as local objects in either a multi-threaded environment or a single-thread environment. A few motivations of why this is desired was highlighted above. The different execution methods are discussed here.

Deployed Web Services

TEF can monitor and report on deployed Web services deployed in an Web services container or server. These are fully deployed Web services that can be contained on any Web server. TEF can interact with these services if they extend a simple TEF messaging API, which allows TEF to communicate basic administration commands and control commands to the Web service. The business logic of the Web service and how it is implemented is left to the Web service itself. The Web service does not have to conform to any specific platform, provided the Web service implements the TEF messaging API. Currently TEF provides support for interacting with axis2 Web service containers and axis2 Web services. TEF uses the axis2 SOAP implementation and so has no interoperability issues with axis2 Web services, but any Web service can be monitored by TEF as long as it implements the TEF API.

The Web services have to extend messaging interface provided by TEF so that every Web service can be configured, initialized, started and stopped by the TEF controller. Currently TEF supports axis2 Web services and allows for easy creation of Web services for an axis2 container.

The TEF messaging API is a list of service endpoints that TEF can connect to to pass information, ranging from information requests to control commands to a Web service. These endpoints receive normal SOAP messages that convey specific instructions. The TEF messaging API stip-

ulate that the following end-points must be made available to the TEF controller:

- admin_receiveInformation
- admin_sendInformation
- admin_initialize
- admin_startService
- admin_stopService
- admin_getLog

The endpoints admin_receiveInformation and admin_sendInformation provide endpoints that can be used for communication between the Web service and TEF. The contents of the messages are dependent on the services themselves. There are no specific messages that have to passed, but allows the TEF controller to send information to the Web service so that a specific scenario can be run without the Web service have to be recompiled and deployed with the new information. In the transaction service protocol, all the Web services are based on a base Web service type, because they all implement the same protocol endpoints. The admin_receive endpoint is used to pass Web service specific information to the Web service at runtime. The service name, the specific policy that the Web service uses and information about other Web services that the Web service connects to is all passed to the Web service using the admin_receiveInformation endpoint. The admin_sendInformation endpoint allows the Web service to connect to TEF to send information to TEF. Both these endpoints are available to be used, but are not necessarily required by different Web services. Both the admin_sendInformation and admin_receiveInformation endpoints use SOAP body elements to contain the information that gets passed between TEF and the Web services.

The admin_initialize, admin_startService, admin_stopService endpoints allow the TEF controller to control the Web service. The Web service is responsible for implementing the methods, so it is up to the Web service to adhere to the requests received from TEF. SOAP are used to convey simple instructions written in XML.

The admin_getLog endpoint requests an XML log of the messages that has been received and sent from the Web service. When using services deployed in remote Web services containers, TEF cannot automatically records the message interaction that each Web service engages in unless all the Web services interact through the TEF proxy. As an alternative to using a proxy service to intercept all messages passed between Web services, the Web service can keep a simple log and sends that to TEF before the service shuts down or when TEF requests the log of a given Web service.

TEF provides a Web service interface that java Web services can implement to ensure that the Web service contains the necessary endpoints. The Web service is still responsible for ensuring that the methods have been implemented. This is because these commands and requests are Web service specific and allow the Web service to be implemented in anyway.

Figure 4.2: Tungsten Web service container showing deployed Web services

Figure 4.3: Web service endpoints showing TEF API endpoints

Figures 4.2 and 4.3 show Web services deployed in the Tungsten Web service container and the endpoints available to the deployed Web services respectively. Tungsten is discussed later in this Chapter.

Multi-threaded execution

Web services created to run locally in a java virtual machine and Web services deployed in a Web services container both implement the same external interface and accepts and responds to messages being sent to them. Because TEF treats Web services and their implementation as a black box, both systems act identically in TEF. The main difference between the two implementations of Web services is that the local objects make use of TEF's transport modules to receive and send messages. Thus there is no difference in the Web services code itself in how the messages get passed between Web services.

In a prototype environment where fully deployed services are not required, local objects can be used to instead of fully deployed Web services. The base code of the Web services are identical to that of the fully deployed Web services, but there is no need to deploy the Web services in a Web services container. This allows TEF more control on the dynamic creation of Web services as the services do not have to be deployed and configured remotely. By using local-objects to run the transaction, the cost of sending SOAP requests over HTTP is negated and thus the transaction run is sped up considerably. This allows the user of TEF to quickly produce a Web service to perform a desired function without having to deploy it in to a Web service container, which can complicate access to system resources before the Web service needs to be fully deployed. In TEF, each of these local objects can be created in their own thread and act autonomously from each other. This provides a clean abstraction from the implementation of Web services, but still retains the independence of the services from each other.

Single-threaded execution

By creating Web services as local classes, the interaction between the Web services can be controlled by a controller class more carefully. This allows more efficient monitoring of the services and the messages passed between the services. By configuring TEF to run an entire transaction in a single-thread, it can eliminate the multi-threaded nature of the Web services and reduce the interactions between the services a synchronized interaction. This can be advantageous, allowing a transaction to be easily monitored as the different services interact and allows the Web services and their interactions to be monitored in the java debugger. In distributed asynchronous applications, interaction bugs are very difficult to reproduce or find, and by using a single thread of concurrency, an application can be debugged more easily. By using local objects in the same java virtual machine, the developer can debug the entire transaction without needing to use distributed debugging techniques.

The ability to debug is desirable in TEF so that the user or developer can be sure that the Web services that have been created functions properly before attempting to monitor a protocol. The aim of TEF is to evaluate protocols and Web services without the complexity of running production ready code. Thus users of TEF should be able to quickly create Web services and test that they work before attempting to incorporate the Web services in complicated transactions.

The single-thread execution method is therefor useful to find errors that would be hard to find in multi-threaded and distributed environments. A transaction is also easier to monitor when only one action occurs at a given time.

TEF allows the user to choose any of these 3 execution methods and also allows the use of remotely deployed Web services alongside the local Web services.

4.2.2 Tools Used

TEF makes use of external libraries to provide the functionality required and to ease development of both TEF and the Web services that are controlled by TEF. The following tools were used to produce TEF and the functionality that TEF provides.

- Java

- Apache Axis2
- Apache Axiom
- Tungsten
- JUnit
- Java Logging
- Vector Visuals

Java

Java was chosen as the development environment for numerous reasons. Ease of development and the variety of tools designed specifically for Web services written in Java provided an ideal environment to build TEF. There are numerous open-source Web services containers available for Java and thus TEF can use any of them so that lock in does not occur. TEF currently uses axis2 as its Web services implementation, although any other (java-based) implementation can be added to TEF with relative ease. Non-java implementations can be used with TEF when using fully deployed Web services, but TEF cannot execute such web services in its own local execution environment.

The java programming language also provides a suitable logging framework that TEF and the Web services use to log and monitor interactions and results during the execution of a transaction.

Logging occurs at various stages in TEF. There are mainly two different logging purposes in TEF. Firstly, TEF produces a log to record its own actions and is useful to record the actions of TEF incase an error occurs within TEF. This log file is not normally used by users of TEF, but TEF produces the log files so that any abnormal behavior can be spotted. Secondly, and more importantly, TEF uses logs to record the actions of the Web services that interact in TEF. These logs can be used to replay a transaction, view the transaction at a specific state and allows the user to see exactly what happened in the transaction. The Web services maintain a log of messages sent and received, and TEF stores these logs, along with logs created by the transaction controller to store a complete record of the transaction.

Apache Axis2

Apache Axis2 was chosen as the Web services implementation for several reasons. Axis2 is an implementation of the SOAP protocol described in Section . Axis2 was chosen because of its adherence to multiple Web services specifications and standards, and its open architecture. Primarily, TEF uses the SOAP processing engine provided by axis2 as well as the XML processing facilities provided by Axiom. Axiom, a name derived from "AXIs Object Model" is a XML pull-parser that allows XML structures to be created, read and traversed. XML parsing is pivotal to using Web services and axis2 provides an elegant API to create and consume XML messages.

Tungsten

Tungsten, mentioned earlier in the Chapter, is a Web services container built around Axis2 and other Apache organization tools. Tungsten is designed to be a stand-alone, or embedded, Web services container that eases the installation of the tools required to deploy Web services. Tungsten is effectively a wrapper around axis2 and other required tools, and provides a simplified install and configuration environment so that Web services can be deployed quickly. In tungsten, Web services are packaged in the Axis2 Archive format, ".aar" and these services can be deployed in any axis2 compliant Web service container.

JUnit testing

JUnit is a testing environment which allows for the automatic execution of tests so that specific application properties can be tested. Test driven development produces code of greater quality and JUnit allows all the different modules of TEF to be tested to ensure that they work together and that there are no logical errors in the code that will cause TEF to function improperly.

Vector Visuals

Vector Visuals is a java library that simplifies interactions with Java2d images. Vector Visuals is used to manage the visual representation of the transactions and the interactions between the services.

4.2.3 TEF modules

TEF is made up of modules that can be extended to add functionality to TEF. The following list shoes the modules that make up TEF.

- TEF
- admin module
- archive module
- clients module
- config module
- GUI module
- logging module
- monitor module
- protocol module
- services module
- test module

- transport module
- utils module

TEF s the main application and effectively coordinates all other modules. Modules all implement the TEFModule interface to ensure that all modules have methods required by TEF to load and interact with them.

The admin module provides the controller for the transactions being simulated in TEF. The admin module is responsible for deploying Web services, configuring them and using a transport module to allow Web services to engage one another. The admin module is responsible for loading applicable service types so that instanced of the Web service can be deployed in a transaction. The admin module utilizes the Java reflection API to deploy services at runtime, and also to configure endpoints described in a services WSDL document.

The archive module is responsible for dealing with packaged Web services. Currently TEF uses the AAR module to process Axis archives. The AAR module is an extension of the Archive module that understands the AAR package, and can then use the resources packaged with the AAR file, even when the AAR file is not deployed in a Web services Container.

The clients module is a module that can be used by TEF if there if the client is the focal point of the transaction. In the trust assurance protocol, the client is an independent application that uses the information received from the protocol execution to determine the most appropriate path. TEF provides the client module the freedom to display a graphical user interface to allow interactive behaviour if required within the TEF graphical user interface and also allows the client module access to TEf's service visualizations and logging services. The Client module has a simple interface to allow the client access to the rest of the services deployed within TEF. Similar to the TEF message API, the client modules, and any extensions of the client modules, has methods for sending and receiving SOAP messages in synchronous and asynchronous methods. This allows TEF to be independent of how the client is implemented, whilst allowing the client to interact in the service.

The config module is a module that keeps information about the different modules together so that TEF can find the active modules, deploy different modules and maintain modules. The config module is also responsible for saving configurations, so that transaction paths,logs and monitor results can be saved and loaded. The config module makes use of Java's serialization API to store the state of different active modules so that they can be restored.

The GUI module provides the main application window for TEF. The GUI module contains an API that allows other modules to register their GUI's and allows them to be displayed within the main window. As an example, the admin module provides a few GUI components to administer the available service types and also the deployed services. These GUI components are then registered with the GUI module so that the user can use them to administer to services that are deployed in TEF. The GUI module allows any module to register a GUI component, allowing new modules to be created to provide new functionality. The modular architecture of TEF was designed to be extensible, so that modules could be created for specific transactions and that new functionality can be added to TEF.

The logging module provides the logging mechanisms for TEF and for the services modelled in TEF. THE logging module can be configured to produce log files or to send the logs to the console or a remote service. The logging module allows TEF to record logging at different levels so that the amount of details being logged can be determined by the logging module.

The monitoring module is similar to the logging module, except that is designed to deliver reports to the user. The monitoring module is designed to be extended for specific protocol evaluation so that the user is presented with applicable reports and results. The monitoring module has access to the logging modules information, can request information from services and can register its GUI component with the GUI module to display its results to the user.

The protocol module allows the user to describe a protocol and the conditions necessary for success. The protocol module can specify the requirements for a protocol to ensure that everything is set up in TEF to facilitate the protocol and can then report on the success of the protocol run based on information received from the logging, monitoring and client modules. The protocol module can be extended for each type of protocol being tested and can provide a GUI component so that the user can see when a protocol failed or succeeded.

The services module is a key module in TEF. The services module provides the necessary components to deploy services within TEF. It includes a GUI representation of a service, so that it can be displayed in TEF's graphical representations, the necessary configuration information for remotely deployed Web services or local objects. The services module is extended to create service types. the services module implements the TEF message API so that any service contains the required endpoints. The service module contains the service specific business logic and data to process specific requests and protocols. The service module can also be used with the AAR module to create Axis2 compliant Web services from the service modules.

The test module is a module that contains all the JUnit tests for TEF. From this module, all other modules can be tested to ensure that they conform to the interaction requirements, so that new modules can be tested to see if they will work with TEF and the other modules. Tests can also be created to test Web services to ensure that the process messages correctly and that they are functioning correctly.

The transport module provides the functionality of getting messages to the intended services. If a Web service is deployed remotely, the transport module will send the message to the Web service in the appropriate manner, either asynchronously or synchronously. If the Web service is a local java object, the transport module will find the correct object, and then use java's reflection API to invoke the correct endpoint with the SOAP message that needs to be sent. Thus services all send and receive messages in the same manner and leaves it up to the transport module to ensure that all messages are received correctly. The transport module can also be extended to simulate flawed transport medium, delays and disruptions.

The utils module provides small utilities that are available to all the other modules. The utilities include creating XML elements from files, parsing XML files, creating certain GUI components and other non-module specific functions.

TEF uses these modules to create, configure and run protocols, services, clients and all the other components necessary to simulate services, interactions between services and the results of the interactions. The modularity of TEF and the extensibility of the modules and the ability to create and register new modules makes TEF a very flexible system in which services, transactions and protocols can be evaluated.

4.2.4 User interface

The user interface of TEF allows the user to configure and deploy Web services, to view a visual representation of Web services and connections between different services, to view execution of interactions between services and to display any results or errors resulting from the transaction. TEF provides gui modules for creating, deploying and configuring services, and provides modules for monitoring the transaction and the results.

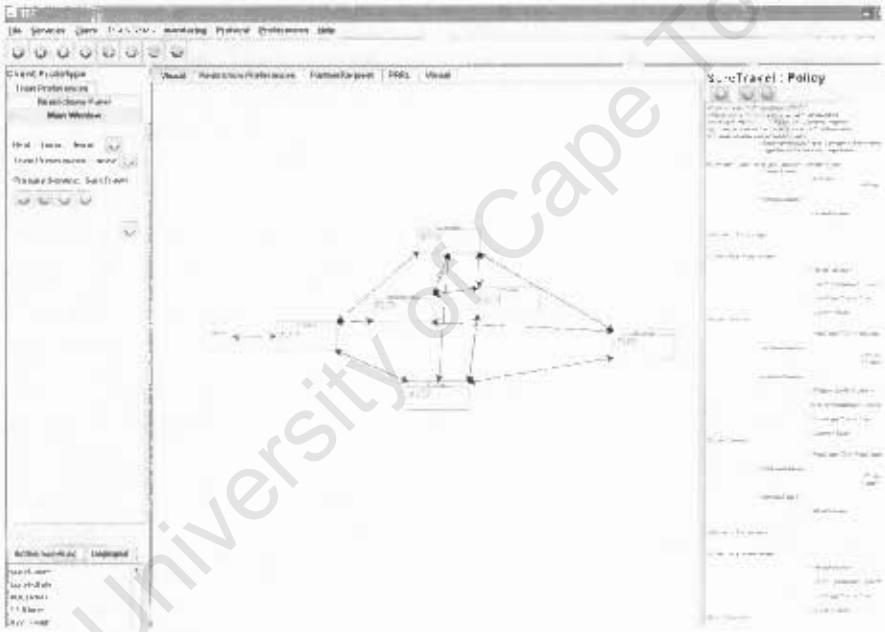

Figure 4.4: Screenshot of the TEF application interface

4.3 Client Prototype

The Trust Assurance Protocol described in Chapter 3.3 allows a customer to determine the possible transaction paths that are available to him/her and to decide which path to choose. Any complete path represented to the customer represents a successful transaction chain that conforms to the restrictions that the customer stipulated in the <PartnerRequest> element that he sent to the primary service provider. The client must choose a transaction path so that the transaction can be initiated. Any complete transaction path available to the client represents a transaction path where a client's restrictions on the service and usage of his/her personal information have been met by all the participating services in each complete transaction chain. Assuming that all the services involved in the transaction chains specified are honest to their word and will not act maliciously once they are involved in the transaction, the client can choose

any of the complete transaction chains and have confidence that he/she knows what will happen with his/her personal information.

However, the goal of the trust assurance protocol is not just to find partners who claim they are suitable, but to really determine if a service is honest in its claims about its operations. In the online world, malicious service providers can not be identified easily, as they have no initial identifiable warning signs that will steer the client away. In contrast, malicious services will look and act exactly like honest services and can be indistinguishable from their honest counterparts.

The trust assurance protocol delivers all the information that can be readily obtained from services and presents it to the client. The information is readily available because the information returned is consistent with the requirements placed upon the service provider by legislation. Any service who does not comply with the request for the information could be deemed suspicious because they would be in breach of the laws requiring them to provide the information.

The client prototype is an extension of the client module provided in the TEF. The client module allows the client to send the initial message to a service that is deployed in TEF. The client is similar to services in that it creates, sends and receives messages as the only external interaction with services. It differs in that it can have an interactive graphical interface so that the user can change the messages being sent or can act on messages coming in. The client module also has access to system resources, and can connect to databases, load files or make use of any other resources to it. It is up to the developer to either make use of a SimpleClient module where the client simply acts on a configuration setting and initiates the interaction with other services, or the user can create an interactive gui client based on the GUIClient module. This module displays a user application which can be configured and handle all messages it sends and receives. The contents of the GUIClient is up to the user entirely, apart from creating a visual representation of the services, and is independent of TEF. It is displayed in TEF for convenience, and allows the client to make use of TEF's other GUI components. The only methods in the client modules are the sendMessage and receiveMessage methods that allow the client to communicate with the other services. This allows the client to do all the processing necessary without adding the complexity to a transaction which is not concerned with such complexity.

There are several aspects to the client prototype. The client interacts with a service and then chooses a resulting transaction path based on several factors. The client's preferences and restrictions on uses of his/her personal information establishes the initial restrictions that are sent to participating services and eliminates services if they fail to comply with the restrictions of the client. The restrictions are contained in a client personal information preferences (PIP) document.

When the client receives the list of possible transaction paths, it uses TEF's service visualization GUI components to display possible paths and failed paths. From this point, it is up to the client to choose a transaction path. The client uses the trust evaluation protocol to receive information from various sources to influence his decision about the different services involved.

The client needs to be able to specify his preferences on using different trust sources. The client

uses a trust preference policy (TPP) document. These documents as well as PIP documents are all stored and reused to provide the customer with information for future transactions. The TPP document is used in conjunction with a database of known services so that the client can determine the *benevolence* factor for a particular service.

The following sections describe the PIP document, the TPP document, the trust calculation, the known services database and the graphical user interface that the client prototype uses.

4.3.1 Preferences and policy documents

For the trust assurance protocol, the client is quite involved with the transaction and requires quite complex interaction. Because the client is actively involved in the transaction and has to specify his/her preferences on personal information and his restrictions. This document relates to the initial starting point of the trust assurance protocol. The client must state what may or may not happen with his personal information and whom may have access to it.

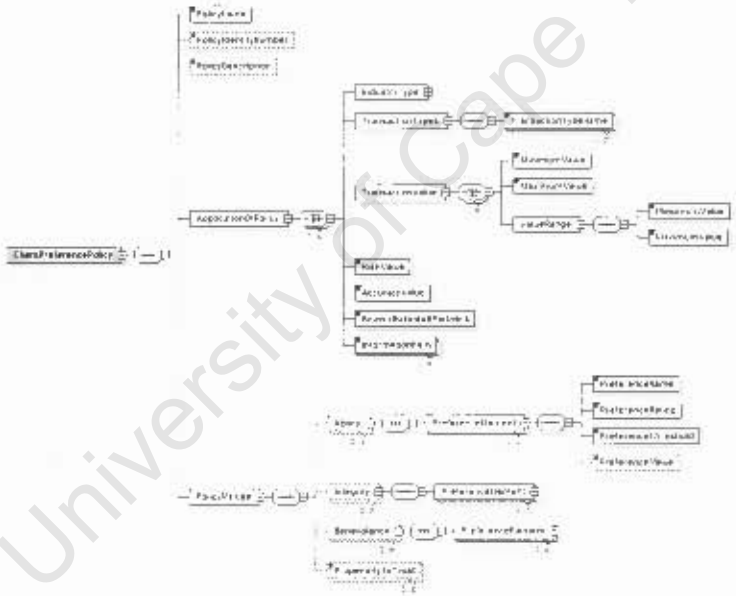

Figure 4.5: Schema of the Client Preference XML document

In Figure 4.5, the schema for the client preference document is shown. Effectively, the schema allows clients to create XML documents that stores information about their restrictions based on services and transactions. The ClientPreferencePolicy element contains the PolicyName, PolicyIdentityNumber, PolicyDescription, ApplicationOfPolicy and PolicyValues elements. The PolicyName, PolicyIdentityNumber and the PolicyDescription elements are text elements allowing the client to state the name, identity number and description of the preference policy respectively. The ApplicationOfPolicy elemenet allows the client to capture the conditions under which the preference document is valid. The client can specify that the document is valid for certain or all industries above or below a certain value and level of risk, required accuracy value and whether it applies to known business partners or not. The client can also specify for with items of information the preference is valid.

The PolicyValues element contains information about how the client weights the different parameters for the trust calculation. The client can state how it wants to weight any of the ability, integrity and benevolence values that were described in Table 3.1. The client can also specify a level of his/her propensity to trust for the trust calculation for which the policy is applicable.

The second preference policy document the client needs relates to the actual trust evaluation section. The client, over time, builds up policies concerning his weighting of the ability, integrity and benevolence factors for industry types, transaction value or risk, monetary value or required accuracy. These policy documents can be combined to create a policy document for each service involved in the transaction and is then used when the authority services are used to determine the trust rating of each service possibly involved in the transaction.

4.3.2 Trust evaluation protocol

Once a client has started the trust assurance protocol, he/she will receive a PartnersRequestResponseList (PRRL). As mentioned before, any path with all leaf nodes passing the restrictions test is eligible to be use d as specified by the clients partnersRequest XML element restrictions on the service and his/her personal information.

In order to source the best possible path out of the available paths, the client needs to evaluate the services involved in the path using the trust evaluation protocol. The trust evaluation occurs independent of the trust assurance protocol, which means the customer can decide how much calculation he needs to do for the transaction to proceed. If the transaction is not too risky for a trusting client, he might choose the first available path without doing any expensive calculations. However, should a client wish to verify any details about the services involved, he/she has the ability to do so. The trust evaluation protocol uses the client preference policy document to evaluate the services and checks to see if he/she has had any previous contact with the service. A trust value is established for each service and if all services pass the client's threshold, then the path can be chosen to proceed. This process can be interactive or automated depending on the client's desire.

4.3.3 User interface

The client prototype user interface consists of different sections. the prototype includes a graphical representation of the PartnerRequestResponseList (PRRL) XML element that is the result of the trust assurance protocol's PartnerRequest XML element. The client can interact with this graphical element to see information about services, including their policies and other information. The user interface also includes creating,loading and editing of PIP documents so that the client can specify his/her restrictions on services and the use of his/her personal information so that the client can start a trust assurance protocol. The interface also includes the facility to create, load and edit TPP documents, so that the client can state his preferences for the weightings of trust rating sources.

Once the client has evaluated the services and compared them to the database of known clients, the graphical user interface will indicate the ratings for the different services and transaction paths. The client can then select the transaction path that has the highest trust rating and create the transaction contract with the primary service provider.

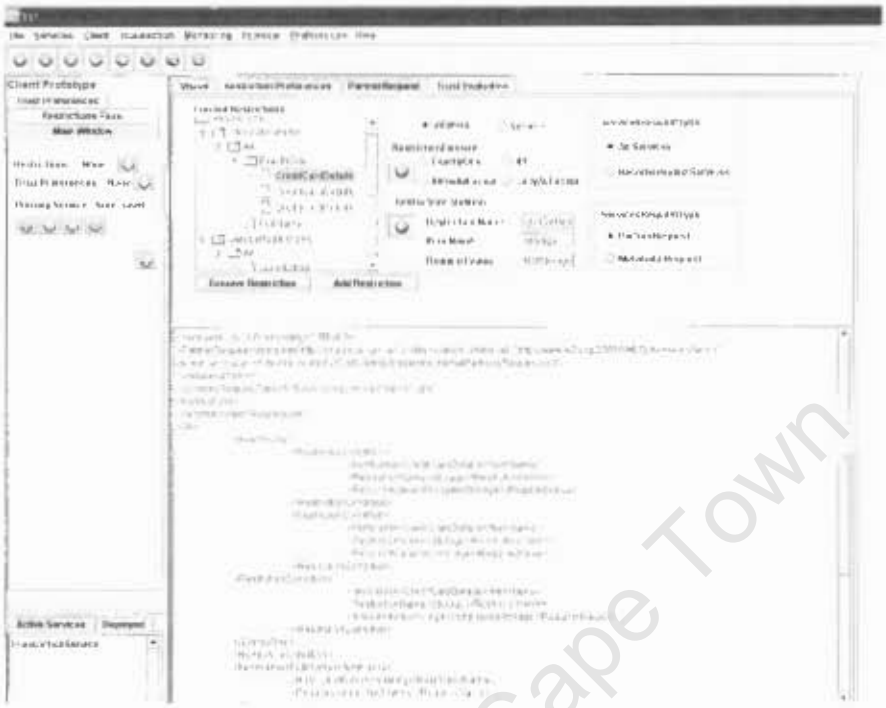

Figure 4.6: Client Prototype showing how the user creates requests for the trust assurance protocols

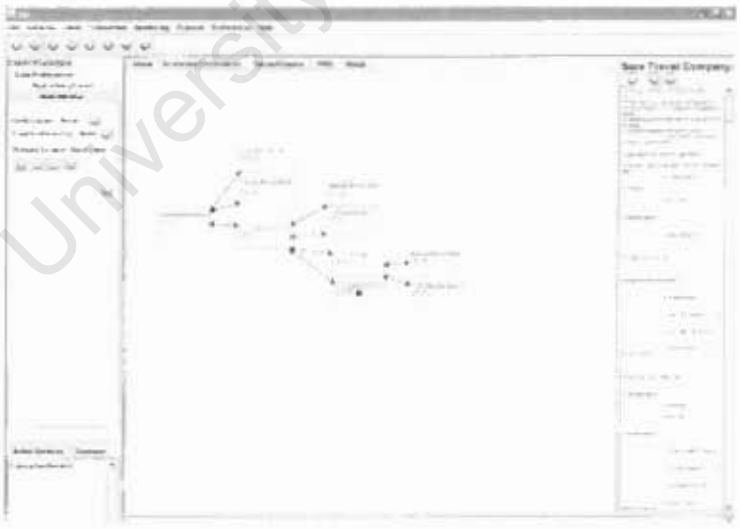

Figure 4.7: Client prototype displaying visual representation of a trust assurance protocol response

Figures 4.6 and 4.7 show the client prototype creating a request and displaying the response of the request respectively. The client prototype integrates with TEF to produce a seamless user experience by allowing the client prototype to use TEF's visualization library to display the

services and their connections.

4.4 ONTOLOGIES

The ontologies created for this dissertation was created so that there is a common vocabulary for client and service providers. A common vocabulary is necessary when describing a business entity because computers can only match identical names whereas businesses can be described in numerous ways in everyday language. This ambiguity complicates a protocol where different actions are taken for different industry types and company sizes. A common vocabulary eliminates ambiguity and allows for easier integration between different systems. The ontologies also serve as a guidebook for clients to know what to expect from companies in different industries. A client has to know whether there is an official industry registry of companies associated with that industry. If there is no common point of reference for the client to refer to and a malicious service omits information about the existence of such an industry association, the client will be none the wiser and thus not have all the pertinent information at his disposal. By using ontologies, the client can easily look up different industry types in different countries, and can look at the legal requirements, authoritative bodies and business procedures that apply to a particular company. If a malicious company then omits any information that would expose the service as fraudulent, then client will be able to pick up the omission easily and will not trust the service provider.

4.4.1 ICB ontology

The ICB ontology represents the different sectors of the stock market sectors and therefore places different companies into different categories. Similar to the legal ontology, the ICB ontology can be extended to include information about industry associations, watch dogs, third party authorities and can provide assistance with verification of a particular company in a particular industry. As an illustration, the Association of South African Travel Agents (ASATA) could verify a travel agent's claim of being a travel agent in South Africa. Any service who fails the verification with the industry association should be regarded as suspicious unless valid and conclusive proof (for the customer) is provided that the service is legitimate. By combining knowledge of different sectors and authority bodies that control different sectors, the ontologies become powerful in providing the client with near perfect information about the information it should receive from a service provider.

4.4.2 Legal ontology

The legal ontology contains a ontological representation of the information that an online service provider must provide. This allows the client to know whether the service provider has omitted any important information. The legal ontology can be extended to incorporate industrial and sectorial regulations as well as legal requirements for different company types. This allows the client to easily see which laws a company has to comply with and can then evaluate a company more accurately.

4.4.3 Tools used

In order to leverage the power of the ontologies, tools to create and interact with the OWL ontologies were used. To create and edit the structure of the ontologies, Protègè 3.2 was used. The images shown below show the Protègè editor and the different views that it provides.

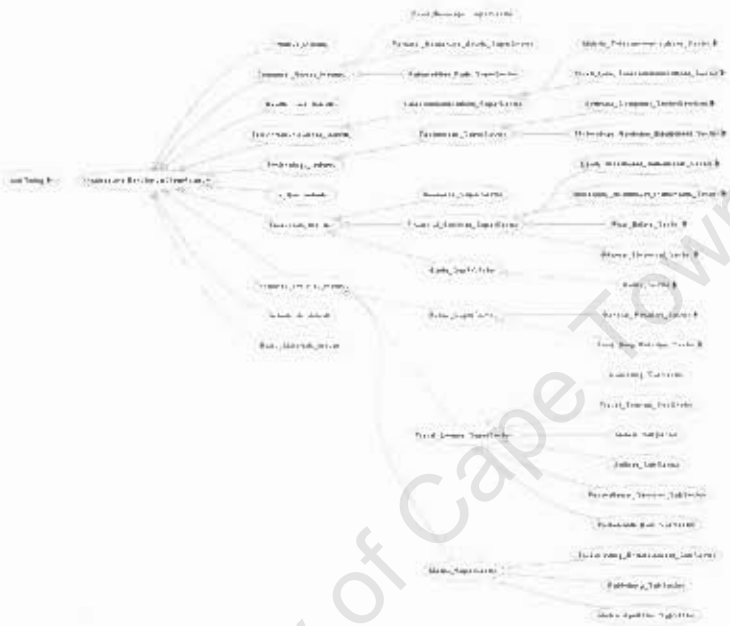

Figure 4.8: Graphical representation of the classes in the ICB ontology

Figure 4.9: Graphical representation of travel and leisure super-sector class in the ICB ontology

In Figure 4.8, the ICB structure can be seen. The ontology is effectively a hierarchial representation of the benchmark and provides the structure so that companies can be classified. Any rules that govern a super-sector will also govern any individuals in the sub-sectors of the super-sector, and thus the user of the ontology can see exactly what legislation and restrictions apply to a certain sector. Combined with an ontology describing the countries of the world, shown in Figure 4.11, these governing bodies can apply internationally or to companies in specific countries. Each class in the ICB ontology can contain individuals. In this case, individuals are companies. As can be seen in Figure 4.10, the companies are placed in the travel and tourism sub-sector of the travel and leisure super-sector. A user can now investigate which laws are applicable to the travel industry of a particular country, and can use this information to make

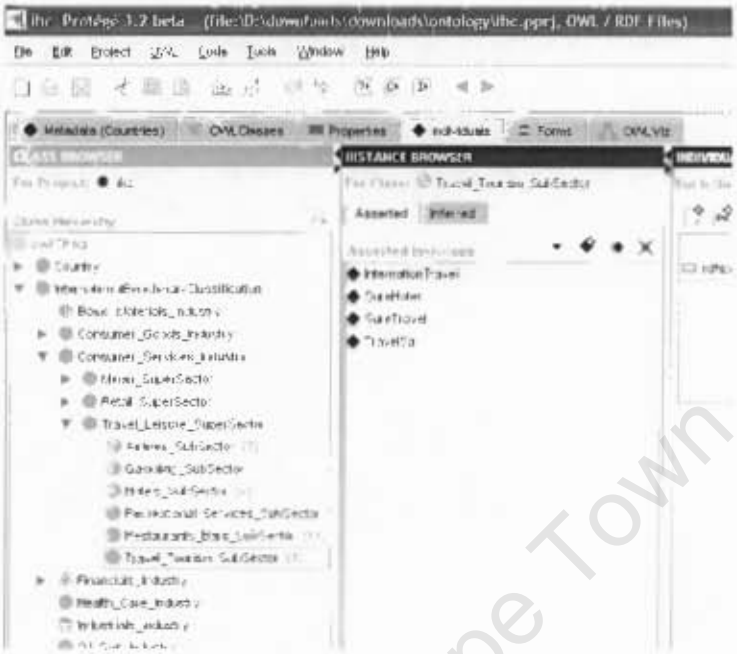

Figure 4.10: Graphical representation of the individuals contained in travel and tourism subsector the in the ICB ontology

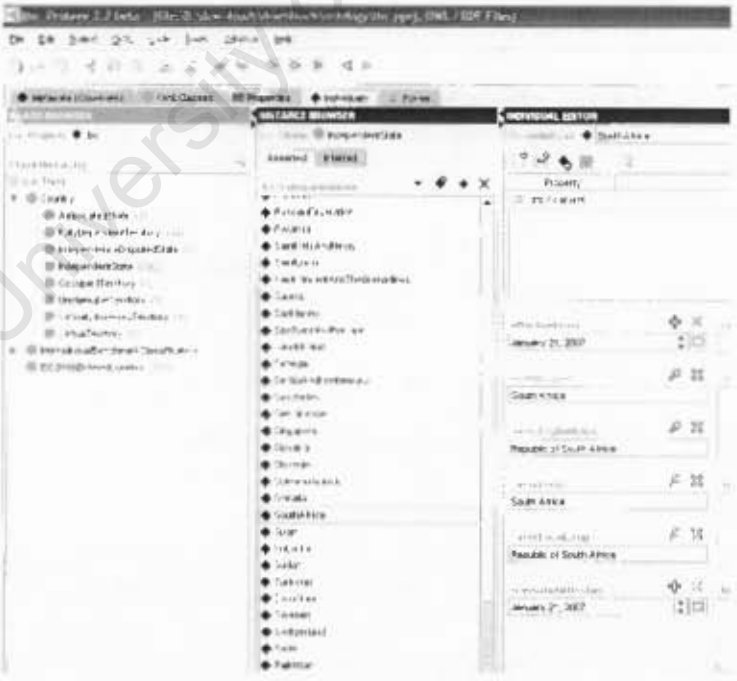

Figure 4.11: Graphical representation of an ontology containing information on all the countries of the world

informed choices about which services to interact with. To interact with an ontology, a library that provides an OWL API has to be used. The Jena framework written in java allows access to OWL ontologies. The Jena framework is described as "A semantic Web Framework for Java" and allows the client prototype to query and traverse the ontologies to find the applicable classes or individuals. The client prototype primarily only uses the ontologies as a vocabulary. This ensures that the service providers describe themselves in a manner that will allow the client to be sure that he/she is dealing with the right type of company.

4.5 SUMMARY

In this chapter, the various artifacts developed in this dissertation was discussed. These artifacts show the results produced from the requirements of creating a trust assurance protocol. The TEF framework allows any Web services transaction to be simulated and can be used for purposes beyond the requirements of this dissertation. Similarly, the ontologies created to standardize the descriptions of companies in terms of their primary business purposes, can be used in other Web service applications so that clients can know which laws are applicable to different companies. These ontologies are designed to be extensible, so that industry bodies can govern the information for a particular industry in a specific country. The client prototype created for this dissertation is a protocol specific application showing the involvement of the client in transactions where the client previously had no knowledge of the transaction. The prototype allows the client to specify restrictions and preferences, and then applies these to the trust assurance protocol.

CHAPTER 5

EVALUATION AND RESULTS

5.1 INTRODUCTION

In this Chapter, the artifacts created in this dissertation are discussed and evaluated. In particular, evaluation of the trust assurance protocol is discussed and the artifacts that can be used independently of this dissertation is evaluated. The results of this dissertation includes a protocol to find suitable transaction paths, a protocol and prototype to determine a trust rating for Web services, a simulation environment in which interactions between Web services can be monitored and vocabularies that can be used and extended to apply to different industries and sectors, so that services may conduct transactions confident that they have all relevant information at their disposal.

5.2 PROTOCOL EVALUATION AND ANALYSIS

The trust assurance protocol allows the client to create request messages to the primary service provider (PSP) to specify the client's preferences and restrictions that apply to services involved in the transaction and also to the use of the client's personal information.

The protocol itself is simple in that each service only receives a single message and sends out a single response. This simplicity negates complicated protocol failures such as deadlock resulting from a complicated protocol scenario. The only deadlock that can occur in the protocol, can result from a service waiting for a response from a service that is unable or not willing to respond. A failure to respond can be mitigated by the underlying transport protocol indicating that an error has occurred or that the connection has timed out and the service is then automatically excluded from the rest of the protocol. The simplicity of the protocol allows for intuitive reasoning to be used to evaluate the protocol.

The following five scenarios were created to exercise the trust assurance protocol and compare results to the expected outcomes.

1 A scenario involving only the client and a suitable PSP

2 A scenario involving only the client and an unsuitable PSP

3 A scenario involving the PSP providing a choice between two services

4 A scenario involving the primary service requiring the use of a mandatory service who requires two external services, each of which has an equivalent service option

5 A scenario involving multiple participating services which produces multiple transaction paths.

5.2.1 Scenario one

Scenario one contains a Web service transaction where the only parties involved in the transaction are the client and the PSP. The PSP does not require any additional services to complete the transaction.

Participants

Client The client in the transaction

PSP SureTravel Web service.

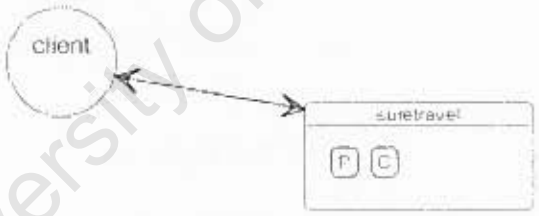

Figure 5.1: TEF graphic showing the participants

In Figure 5.1, the connection between the client service and the PSP, Sure Travel, can be clearly seen.

Before the trust assurance protocol is run, the client cannot be sure that Sure Travel will be the only service involved in the transaction. The client needs to send a PartnerRequest XML element to the service. This is created using the Client prototype within TEF.

Figure 5.2 show how the client creates a PartnerRequest element to send to the PSP. The client can specify restrictions on the service or on the use of personal information. From the PartnerRequest depicted in Figure 5.2, it can be seen that the client has created a request that contains no restrictions on any services.

Sure Travel responds by sending a PartnerRequestResponseList (PRRL) element which indicates whether or not the PSP has passed the restrictions that the client presented. In this sce-

Figure 5.2: TEF creating a PartnerRequest element with no restrictions

No of Services	Total No messages	Size of Request	Size of Response	Total Size of Messages
1	2	311 bytes	11.34 Kbytes	12.71 Kbytes

Table 5.1: Summary of the messages sent in the trust assurance protocol

nario the client didn't stipulate any restrictions. The PRRL of Sure Travel didn't include a ServiceGroup element, indicating that it doesn't use any services to perform any transactions.

The result of the PartnerRequest request is shown below. The policy document that is sent with the PPRL has been omitted for brevity.

```
<PartnersRequestResponseList
serviceURL="http://192.168.8.20:9762/axis2/suretravel"
serviceName="Sure Travel Company">
    <ServiceInformation>
        <Policy xmlns="http://dna.cs.uct.ac.za/TrustAssurance"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xsi:schemaLocation="http://dna.cs.uct.ac.za/TrustAssurance
            C:\masters\masters\schema\policy.xsd">
            <EntityName>Sure Travel Company</EntityName>
            <LegalStatus>Partnership</LegalStatus>
            <CompanyType>Travel_and_Leisure</CompanyType>
            ...
            ...
        </Policy>
    </ServiceInformation>
    <ServiceFault/>
</PartnersRequestResponseList>
```

If the client receives a PRRL from Sure Travel that indicates that Sure Travel complies with the customers restrictions and contains no other information about additional services to be used in a transaction, then the trust assurance protocol is completed. The customer now has to decide

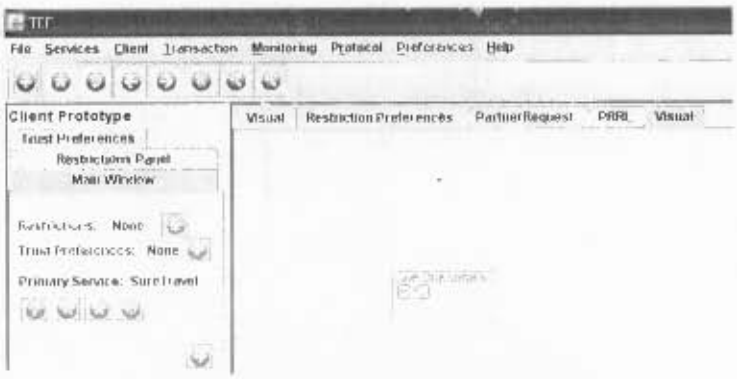

Figure 5.3: TEF Visualisation of the PRRL received by the client

whether or not he/she can trust the service. This step is discussed in Section 5.4. Based on the results obtained from the trust calculation, the customer either proceeds with the service or terminates any further action. The client proceeds by creating a contract that effectively states the PRRL back to the service so that the service can acknowledge which services will be involved in the transaction and what restrictions are placed on personal information items. The contract is discussed in Section 5.4.

5.2.2 Scenario two

Scenario two is identical to scenario 1 except that the client sends restrictions to Sure Travel and that Sure travel does not comply with the restrictions that the client has stipulated.

Participants

Client The client in the transaction

PSP SureTravel Web service.

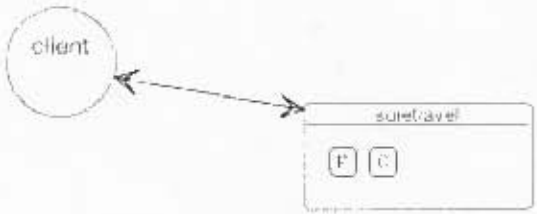

Figure 5.4: TEF graphic showing the participants

In Figure 5.4, the connection between the client service and the PSP, Sure Travel, is shown.

Before the trust assurance protocol is run, the client cannot be sure that Sure Travel will be the only service involved in the transaction. The client needs to send a PartnerRequest XML element to the service. This is created using the Client prototype within TEF.

Figure 5.5: TEF creating a PartnerRequest element with restrictions

Figure 5.5 show how the client creates PartnerRequest elements to send to the PSP. The client can specify restrictions on the service or on the use of personal information.

Sure Travel responds by sending a PRRL element which indicates whether or not it has passed the restrictions that the client has presented. In this scenario the client stipulated restrictions that Sure Travel didn't meet, and the PSP indicates the error in the PRRL.

The result of the PartnerRequest request is shown below. No policy Document was supplied because the service did not pass the restriction test.

```
<PartnersRequestResponseList
serviceURL="http://192.168.8.20:9762/axis2/suretravel"
serviceName="Sure Travel Company">
    <ServiceFault>
        <RestrictionCondition criticalFault="yes">
            <ItemName>FullName</ItemName>
            <RestrictionName>Sharing</RestrictionName>
            <ServiceValue>Unrelated3rdParties</ServiceValue>
            <RequiredValue>NoSharing</RequiredValue>
        </RestrictionCondition>
    </ServiceFault>
</PartnersRequestResponseList>
```

No of Services	Total No messages	Size of Request	Size of Response	Total Size of Messages
1	2	1411 bytes	404 bytes	1815 bytes

Table 5.2: Summary of the messages sent in the trust assurance protocol

Figure 5.6: TEF Visualisation of a PRRL received by the client

If Sure Travel failed the PartnerRequest restrictions, TEF would indicate the failure and indicate what the failure was. Figure 5.6 shows that the Sure Travel service has been marked as failing the restrictions set by the customer and shows the PRRL failure display indicating what conditions the service failed. The client can now either investigate the failure reasons presented by Sure Travel or simply terminate the transaction.

5.2.3 Scenario three

Scenario three extends the scenario presented in the first scenario by creating a PSP that makes use of an additional service to complete its transaction.

In this scenario, the PSP can use either of the two additional services to complete the transaction. In this instance, Sure Travel can use either XYZ credit merchant or the Nigerian credit merchant. In the modular architecture of Web services, many services can provide the same functionality and a service can thus make use of any service providing the same functionality. Only the PSP can decide which service it uses in its transactions. The PSP can either make use of services with which it already has existing business relationships or it can dynamically locate appropriate services. This process is transparent to the client and to the trust assurance protocol and does not affect the protocol operation at all.

Participants

Client The client in the transaction

PSP SureTravel Web service.

Secondary Service XYZ credit merchant

Secondary Service Nigerian credit merchant

In Figure 5.7, the client is in direct contact with Sure Travel. Sure Travel connects to the two credit merchant banks. From this information, it is not possible to know whether Sure Travel

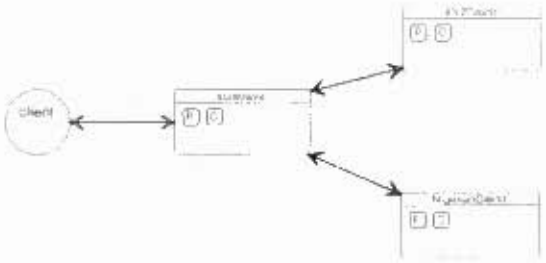

Figure 5.7: TEF graphic showing the participants

requires both services or not, the visualization only shows the connections that the services have with each other.

Before the trust assurance protocol is run, the client can not be sure that Sure Travel will be the only service involved in a transaction. The client needs to send a partnerRequest XML element to the service. This is created using the Client prototype within TEF. The PartnerRequest message is the same as the request created for the first scenario and is shown in Figure 5.2.

Sure Travel responds to the request by sending a PRRL element which contains all the information about the transaction. The PRRL of Sure Travel includes a ServiceGroup element, indicating that it uses additional services to perform its service.

The result of the PartnerRequest request is shown below. The policies sent with the PPRL has been omitted for brevity.

```
<PartnersRequestResponseList
xmlns:URL "http://192.168.0.20:8762/axis2/SureTravel"
serviceName="Sure Travel Company">
    <ServiceInformation>
        <Policy xmlns "http://dma.cs.uct.ac.za/TrustAssurance"
          xsi:schemaLocation="http://dma.cs.uct.ac.za/TrustAssurance C:\masters\masters\schema\policy.xsd">
            <EntityName>Sure Travel Company</EntityName>
        </Policy>
    </ServiceInformation>
    <ServiceGroup>
        <ServiceEquivalents>
            <PartnersRequestResponseList serviceURL "http://192.168.8.20:8762/axis2/XYCredit" serviceName="XYZ Merchant Bank">
                <ServiceInformation>
                    <Policy xmlns "http://dma.cs.uct.ac.za/TrustAssurance" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                      xsi:schemaLocation="http://dma.cs.uct.ac.za/TrustAssurance C:\masters\masters\schema\policy.xsd">
                        <EntityName>XYZ Merchant Bank</EntityName>
                    </Policy>
                </ServiceInformation>
                <ServiceFault/>
            </PartnersRequestResponseList>
            <PartnersRequestResponseList serviceURL="http://192.168.8.20:8762/axis2/NigerianCredit"
              serviceName="Nigerian Merchant Bank">
                <ServiceInformation>
                    <ServiceFault>
                        <RestrictionCondition trivialFault="yes">
                            <ItemName>Territorial Jurisdiction</ItemName>
                            <RestrictionName>South Africa</RestrictionName>
                            <ServiceValue>Y.gerias</ServiceValue>
                        </RestrictionCondition>
                    </ServiceFault>
                </ServiceInformation>
            </PartnersRequestResponseList>
        </ServiceEquivalents>
    </ServiceGroup>
    <ServiceFault/>
```

No of Services	Total No messages	Size of Request	Size of Response	Total Size of Messages
3	6	311 bytes	20.25 Kbytes	33.22 Kbytes

Table 5.3: Summary of the messages sent in the trust assurance protocol

The PRRL element from Sure Travel embeds the two external PRRL's into a ServiceEquivalents element to indicate to the client that the two services are used for the same purpose and that the services can be interchanged. The PRRL for XYZ credit merchant indicates that it complies with the restrictions supplied by the client, and has returned its service information. Nigerian credit merchant has cited its territorial jurisdiction as the critical reason why it could not comply with the clients restrictions. This does not mean that the client cannot choose to use Nigerian credit in the transaction, it simply indicates that Nigerian credit could not comply with the restrictions provided by the client. The visual representation of the PRRL received by the client can be seen in Figure 5.8.

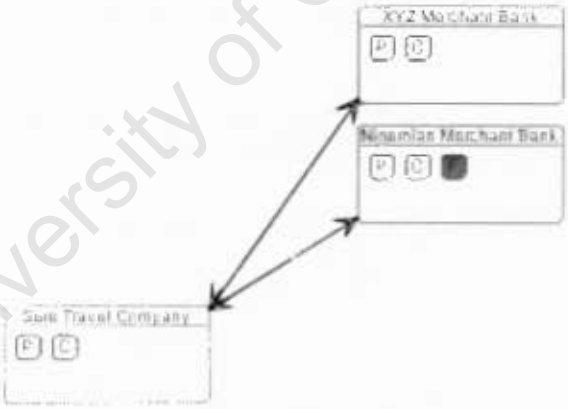

Figure 5.8: TET graphic showing the PRRL element received by the client

5.2.4 Scenario four

Scenario four shows multiple services involved in the transaction. Sure Travel uses a partner site, Sure Hotels, who in turn uses services to complete the transaction.

In this scenario, the primary service provider has to use the secondary service to complete the transaction. This shows to the client that their is no choice in which companies can be involved in the transaction at that point. Sure Travel has to use Sure Hotels to complete the transaction. Sure Hotels connects to either ABC Hotels or 123 Hotel, and to either XYZ credit merchant or the Nigerian credit merchant. The setup of the transaction is shown in Figure 5.9.

Participants

Client The client in the transaction

PSP SureTravel Web service.

Secondary Service Sure Hotels

Secondary Service XYZ credit merchant

Secondary Service Nigerian credit merchant

Secondary Service ABC Hotels

Secondary Service 123 Hotel

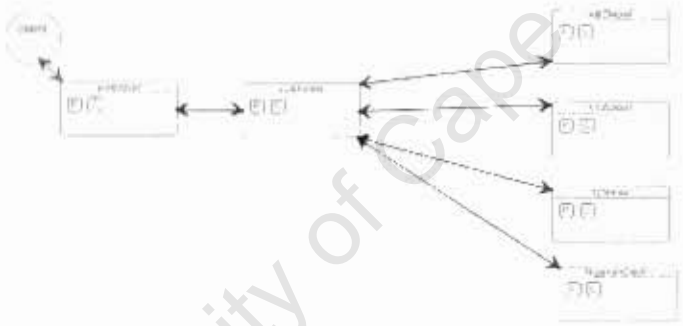

Figure 5.9: TEF graphic showing the participants

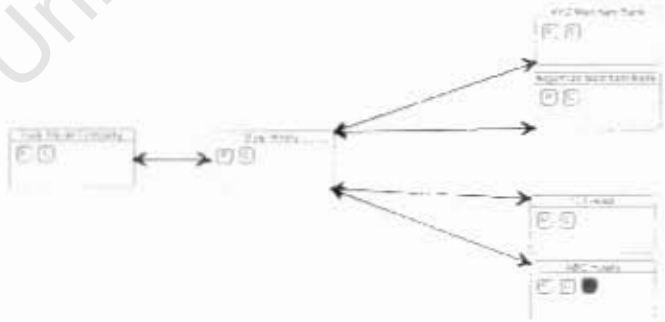

Figure 5.10: TEF graphic showing the PRRL element received by the client

In Figure 5.9, the client is in direct contact with Sure Travel. Sure Travel connects to Sure Hotels, who in turn connects to the two credit merchant banks, and two different hotels. In the PRRL view shown in Figure 5.10, it can be seen that the two hotels are equivalent and that the client has to choose one of the services. The same applies to the two credit merchants.

No of Services	Total No messages	Size of Request	Size of Response	Total Size of Messages
6	12	1411 bytes	42.25 Kbytes	99.18 Kbytes

Table 5.4: Summary of the messages sent in the trust assurance protocol

Before the trust assurance protocol is run, the client can not be sure that Sure Travel will be the only service involved in a transaction. The client needs to send a partnerRequest XML element to the service. This is created using the Client prototype within TEL. The PartnerRequest message is the same as the request created for the first scenario and is shown in Figure 5.5.

Sure Travel responds to the request by sending a PRRL element which contains all the information about the transaction. The PRRL of Sure Travel includes a ServiceGroup element, indicating that it uses additional services to perform its service.

The result of the PartnerRequest request is shown below. The policies sent with the PPRL has been omitted for brevity.

5.2.5 Scenario five

Scenario five shows multiple services being used to process the transaction. Sure Travel uses a partner site, Sure Hotels, who in turn uses services to complete the transaction.

In this scenario, the primary service provider uses multiple secondary service to complete the transaction. The primary service provider uses a specific service as well as one of two services that the client can choose. The setup of the services can be seen in Figure 5.9.

Participants

Client The client in the transaction

PSP SureTravel Web service.

Secondary Service Sure Hotels

Secondary Service XYZ credit merchant

Secondary Service Nigerian credit merchant

Secondary Service ABC Hotels

Secondary Service 123 Hotel

Figure 5.11: TEF graphic showing the participants

From Figure 5.11, the client is in direct contact with Sure Travel. Sure Travel connects to Sure Hotels and credit merchants. Sure Hotel connects to two credit merchant banks, and two different hotels. These hotels then in turn connect to two merchant banks. This scenario shows how a PRRL element can indicate that the same service can be used in the same transaction. If the client wishes, he can choose a service that appears multiple times in the service to reduce the number of unique services involved in the transaction. In the PRRL image shown in Figure 5.12, The client can see the paths that are available to him and can see that Sure Hotels has failed the restrictions that were supplied. Figure 5.12 parallels Figure 3.16 in how it collects and presents the transaction chain to the client. By receiving one hierarchial structure containing all the possible paths, the client can reconstructs any path and investigate the services involved in the transaction without having to make repetitive requests to services.

Before the trust assurance protocol is run, the client can not be sure that Sure Travel will be the only service involved in a transaction. The client needs to send a partnerRequest XML element to the service. This is created using the Client prototype within TEF. The PartnerRequest message is the same as the request created for the first scenario and is shown in Figure 5.5.

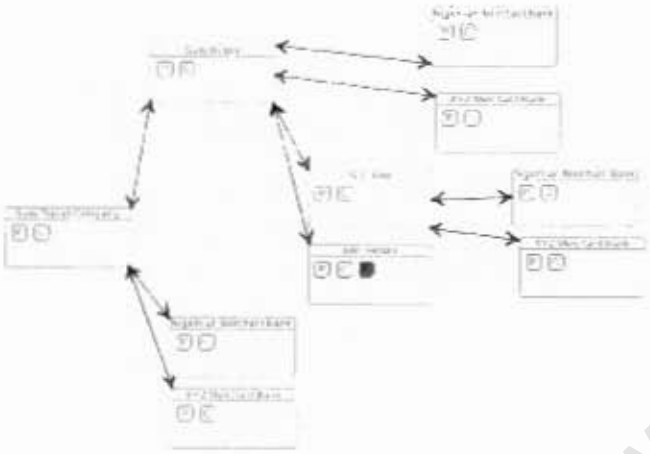

Figure 5.12: TEF graphic showing the PRRL element received by the client

The result of the PartnerRequest request is shown below. The policies sent with the PPRL has been omitted for brevity.

No of Services	Total No messages	Size of Request	Size of Response	Total Size of Messages
6	20	311 bytes	60.05 Kbytes	158.00 Kbytes

Table 5.5: Summary of the messages sent in the trust assurance protocol

Table 5.6 shows the list of messages that was sent throughout the trust assurance protocol. The table shows the messages that were sent and received and shows the message size. In TEF, the monitor module allows the user to view any of the messages. The monitor module is shown in Figure 5.13. This allows the user to investigate what messages were sent between the services during the protocol.

Figure 5.13: TEF monitor module showing list of messages

5.2.6 **Scenario discussion**

The scenarios show how the trust transaction protocol works and what messages are created and sent in the protocol. Table 5.7 show a summary of the scenarios presented. From the table it can be seen that there is no distinct correlation between the number of services involved in a transaction and the size of messages received. Scenario three and four both use six services, but scenario four created 20 messages whereas scenario three produced only 12. This is because the same Web services are used in for different purposes in scenario four. If each Web service was

Message No	From	To	Size of message
1	Client	Sure Travel	1411 bytes
2	Sure Travel	Sure Hotels	1411 bytes
3	Sure Hotels	ABC Hotel	1411 bytes
4	ABC Hotel	Sure Hotels	431 bytes
5	Sure Hotels	123 Hotel	1411 bytes
6	123 Hotel	XYZ credit	1411 bytes
7	XYZ credit	123 Hotel	8841 bytes
8	123 Hotel	Nigerian credit	1411 bytes
9	Nigerian credit	123 Hotel	213 bytes
10	123Hotel	Sure Hotels	17890 bytes
11	Sure Hotels	XYZ credit	1411 bytes
12	XUZ credit	Sure Hotels	8841 bytes
13	Sure Hotels	Nigerian credit	1411 bytes
14	Nigerian credit	Sure Hotels	213 bytes
15	Sure Hotels	Sure Travel	40712 bytes
16	Sure Travel	XYZ Credit	1411 bytes
17	XYZ credit	Sure Travel	8841 bytes
18	Sure Travel	Nigerian Credit	1411 bytes
19	Nigerian credit	Sure Travel	213 bytes
20	Sure Travel	Client	61492 bytes

Table 5.6: List of the messages sent in the trust assurance protocol

Scenario No	Total No Services	Total No Messages	Total Size	No of paths
1	1	2	12.71 Kbytes	1
2	1	2	1.77 Kbytes	0
3	3	6	33.22 Kbytes	1
4	6	12	99.18 Kbytes	2
5	6	20	158.00 Kbytes	8

Table 5.7: Summary of scenario results

unique, each service would only receive one request and send one response. Thus the number of messages would be $2n$ where n is the number of nodes in the graph. The trust assurance protocol thus scales nicely for every additional node entered into the network.

The amount of data sent with trough the trust assurance protocol is very much dependent on what information the client requests from each service. In the case of the scenarios, the client requested the full policy document of all the services, and therefore burdened the system with transferring the policies to the client. If the client requested a URL to a policy document from each service instead of the full policy, the total size of the messages sent and received in the scenario would be 3.23 Kbytes instead of 158 Kbytes. The protocol is thus quite practical in that the results it produces are not impractical, and that the size of the messages can be drastically reduced if the client chooses not to receive the full policy documents.

5.2.7 Transaction contract

Once the trust assurance protocol completes and returns all the available transaction paths back to the client, he/she can evaluate the services involved in the transaction and present the PSP with the transaction path that the client has chosen. The trust evaluation process is shown in Section 5.4. A contract for scenario 4 is shown below, where the client has chosen XYZ Credit, 123 Hotel, and Sure Hotel as the transaction partners for Sure Travel. The signatures have been omitted for brevity. The signatures ensures that the services acknowledge the impending transaction request. This allows the client to verify that all the services involved in the transaction know that the transaction will proceed and that the services will know which services they may use to process the transaction.

```
<?xml version="1.0" encoding="UTF-8"?> <FRELSelection
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="C:\master\xml-tests\schema\contract.xsd">
    <ServiceInformation URL="www.suretravel.com">
        <ServiceName>SureTravel</ServiceName>
        <ServiceURL>www.suretravel.com/GetPolicyURL</ServiceURL>
    </ServiceInformation>
    <Description>Signer acknowledges that it will only use agreed upon services as specified in this document</Description>
    <External Services>
        <FRELSelection xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xsi:noNamespaceSchemaLocation="C:\master\xml-tests\schema\contract.xsd">
            <ServiceInformation URL="www.xyzcredit.com">
                <ServiceName>XYZCredit</ServiceName>
                <ServiceURL>www.xyzcredit.com</ServiceURL>
            </ServiceInformation>
            <Description>Signer acknowledges that it will only use agreed upon services as specified in this document</Description>
            <External Services>
                <FRELSelection xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                  xsi:noNamespaceSchemaLocation="C:\master\xml-tests\schema\contract.xsd">
                    <ServiceInformation URL="www.123hotel.com">
                        <ServiceName>123Hotel</ServiceName>
                        <ServiceURL>www.123hotel.com</ServiceURL>
                    </ServiceInformation>
```

```
                <Description>signer acknowledges that it will only use agreed upon services as specified
                  in this document</Description>
                <ExternalServices>
                    <PRRLSelection xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                      xsi:noNamespaceSchemaLocation="C:\masters\masters\schema\contract.xsd">
                        <ServiceInformation URL="www.xyzcredit.com">
                            <ServiceName>XYZCredit</ServiceName>
                            <ServiceURL>www.xyzcredit.com</ServiceURL>
                        </ServiceInformation>
                        <Description>signer acknowledges that it will only use agreed upon services as
                          specified in this document</Description>
                        <Signature/>
                    </PRRLSelection>
                    <PRRLSelection xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                      xsi:noNamespaceSchemaLocation="C:\masters\masters\schema\contract.xsd">
                        <ServiceInformation URL="www.xyzcredit.com">
                            <ServiceName>XYZCredit</ServiceName>
                            <ServiceURL>www.xyzcredit.com</ServiceURL>
                        </ServiceInformation>
                        <Description>signer acknowledges that it will only use agreed upon services as
                          specified in this document</Description>
                        <Signature/>
                    </PRRLSelection>
                </ExternalServices>
                <Signature/>
            </PRRLSelection>
            <PRRLSelection xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
              xsi:noNamespaceSchemaLocation="C:\masters\masters\schema\contract.xsd">
                <ServiceInformation URL="www.xyzcredit.com">
                    <ServiceName>XYZCredit</ServiceName>
                    <ServiceURL>www.xyzcredit.com</ServiceURL>
                </ServiceInformation>
                <Description>signer acknowledges that it will only use agreed upon services as
                  specified in this document</Description>
                <Signature>
                    <PublicCertificateURL>www.xyzcredit.com/support/keys/</PublicCertificateURL>
                    <SignatureHash/>
                </Signature>
            </PRRLSelection>
        </ExternalServices>
        <Signature/>
    </PRRLSelection>
</ExternalServices>
<Signature/>
</PRRLSelection>
```

5.3 Web services execution framework Analysis and Evaluation

The use of TEF can be seen in the scenarios described above. TEF allows the user to set up scenarios, deploy services and protocols and incorporate a client into the transaction. TEF then monitors the interactions between the services to as the transaction progresses. TEF monitors the interactions between the services and the services themselves to provide the user with relevant information about the status of the transaction and allows a client module to be imported into TEF so that the client's decision can affect the transaction being simulated.

By deploying services in TEF either as fully deployed remote Web services or as local objects within the TEF environment, the user can use existing services, providing that they have support for the TEF message API, and create new services that can quickly be created to test a new protocol or service endpoints.

The modules created to provide TEF with its functionality provide a flexible architecture in which modules can interact with each other and display information to the user of TEF, without have to recompile TEF or change any fundamental internals.

5.4 TRUST MODEL EVALUATION

In order to choose the best available path for a transaction, the client must be able to determine whether or not a service is trustworthy. If a service is malicious or dishonest, there is no inherent information that the client can use to determine that a service could pose a threat to the client or his personal information. This is the reason that vocabularies and metadata about industries and legislation is required. This allows the client to spot and report on omission of information that a service supplies. However, the client still cannot tell if a client is a reputable service from this information. The trust calculation uses information available from different sources to determine whether a service can be trusted for a particular transaction. The amount of assurance that a client requires is completely dependent on the client and how highly he values the information he supplies in the transaction and how important the transaction is to the client.

The trust calculation collects information from third parties online according to the list of parameters provided in Table 3.1. An ontology of industry specific authorities, watch-dogs, industry bodies, experts and communities allows the client to find the necessary services to query to establish a trust rating for a company.

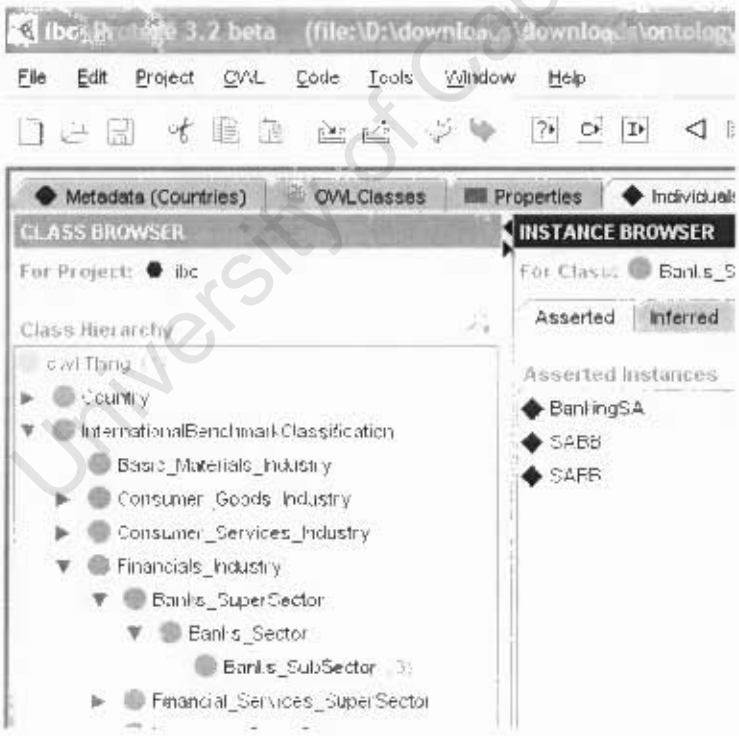

Figure 5.14: Ontology showing authorities that apply to a particular industry for a specific country

in Figure 5.14, an ontology contains information about authorities concerning the banking sector in South Africa. In this example, the three registered authorities include BankingSA, the South African Reserve Bank (SARB) and the South African Business Bureau (SABB). In this scenario, BankingSA is a community forum service that stores user's opinion of financial sector

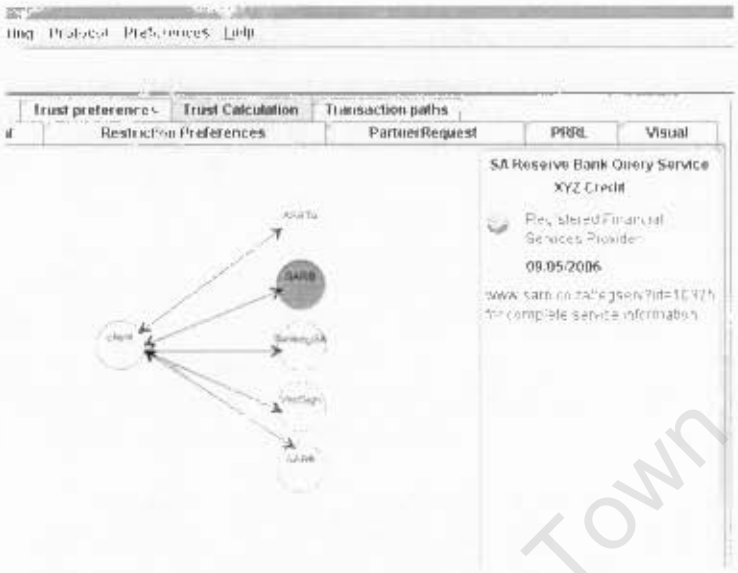

Figure 5.15: TEF graphic authority services that apply to a transaction

companies. SARB is the official authority in the sector and can verify whether or not a service is registered as a financial service provider. SABB is a 3rd party industry watch-dog that rates all financial services providers. From this ontology, the client can find the services that are required in order to evaluate the services. Figure 5.15 shows TEF's representation of the authority services and the reply received from querying the SARB service.

Using the client trust preference document with the results from querying the authorities relating to the services, a trust value can be established for each service in the transaction. The client collects the information regarding ability and integrity from the authority services and other external means and has no initial values for benevolence. From the client's preference document, the trust calculation parameters to be used in the evaluation for XYZ Credit is shown in Table 5.8. The table shows the parameters that the client has chosen for the evaluation, the weighting of each parameter and the values received from the different authorities. Based on this information, the ability rating of XYZ Credit can be worked out to be $A = 0.33(0.687) = 0.227$. Integrity is worked out as $0.9974(0.513) = 0.512$. The weighting of 0.9974 comes from the integrity value as it is the first transaction that the client has conducted with the service. From these values, the trust rating of XYZ Credit can be worked out to be 56.73%. XYZ Credit can be considered somewhat trustworthy by the client based on the trust weighting values the client decided to use and the results received from external services.

If the client deals with XYZ Credit again in the future, the client prototype will recognize XYZ credit as a known service and use the available benevolence ratings from the previous transactions and alter the weighting of benevolence and integrity to take into account the previous interaction. In the second transaction, Integrity has a weighting of 0.605 and benevolence 0.395. Table 5.9 shows the same table as Table 5.8, but has ratings for benevolence included. the calculation for the trust value will thus be $0.33(0.687) + \frac{2}{3}(0.605(0.513) + 0.395(0.864)) = 0.66$, equals a trust value of 66.14%. The increase in trust is expected because the benevolence values

	Name	Rating	Threshold	Value
Ability	Accreditation bodies	5		0.7
	Self-regulatory bodies	4		0.68
	Public rating of expertise in domain	4		0.72
	Third party ability rating	3		0.63
	Company type	7	1	1
	Public certificate	7	1	1
Integrity	Public reputation	4	0.5	0.6
	Current Legal challenges	3		0
	Consistency	4		0.7
	Operational openness	4		0.4
	Acceptance of codes of practice	4		1
	Sense of justice	4		0.34
	Observed consistency	3		0.4
	Jurisdiction	7		1
Benevolence	Timely delivery of goods or services	0		0
	Adherence to business contract	0		0
	Customer service	0		0
	Refunds or exchanges	0		0

Table 5.8: Table showing the trust evaluation parameters for XYZ Credit.

are high. If the customer had a bad experience with a service in the first transaction they had together, a benevolence table like the one shown in Table 5.10 could be expected. When trust is evaluated, a trust rating of 53.74% is obtained. This is expected because a bad benevolence value initially doesn't carry as much weight as integrity initially. If the same parameters are used for the third and fourth transactions, the trust values become 53.14% and 49.16% respectfully.

The trust model illustrates the ability to gather information about a service from different sources and to compute a value which can be used to measure homogenous services. If two services provide the same functionality and both conform to the client's restrictions, then the trust evaluation will allow the client to choose the service that is perceived to be more trustworthy by other people, watch-dogs and industry bodies or to base the selection of services on other factors, such as transaction cost. A client may be willing to pay a premium to use a transaction path that provides more assurances. In order to fully evaluate whether the trust model provides the client with an appropriate measure of a service's trustworthiness, a more robust study is required to investigate its feasibility. In this dissertation, the trust model addresses the requirement of distinguishing between two different services when currently, there are no fixed mechanisms in place to evaluate services.

Figure 5.16 shows TEF displaying different suitable transaction paths, along with the trust values that was calculated for each service. In the client panel, the cumulative trust value for a transaction chain is shown to give an overall impression of a transaction path. The two paths shown in Figure 5.16 is 55.78%, 55%and 56.17% from the top path in the picture to the bottom path. From this information, the client can choose the path that is most suitable and issue the

	Name	Rating	Threshold	Value
Ability	Accreditation bodies	5		0.7
	Self-regulatory bodies	4		0.68
	Public rating of expertise in domain	4		0.72
	Third party ability rating	3		0.63
	Company type	7	1	1
	Public certificate	7	1	1
Integrity	Public reputation	4	0.5	0.6
	Current Legal challenges	3		0
	Consistency	4		0.7
	Operational openness	4		0.4
	Acceptance of codes of practice	4		1
	Sense of justice	4		0.34
	Observed consistency	3		0.4
	Jurisdiction	7		1
Benevolence	Timely delivery of goods or services	3		0.8
	Adherence to business contract	4		1
	Customer service	4		0.56
	Refunds or exchanges	0		0

Table 5.9: Table showing the trust evaluation parameters for XYZ Credit.

	Name	Rating	Threshold	Value
Benevolence	Timely delivery of goods or services	3		0.3
	Adherence to business contract	4		0.5
	Customer service	4		0.26
	Refunds or exchanges	0		0

Table 5.10: Table showing benevolence values XYZ Credit

contract to the PSP.

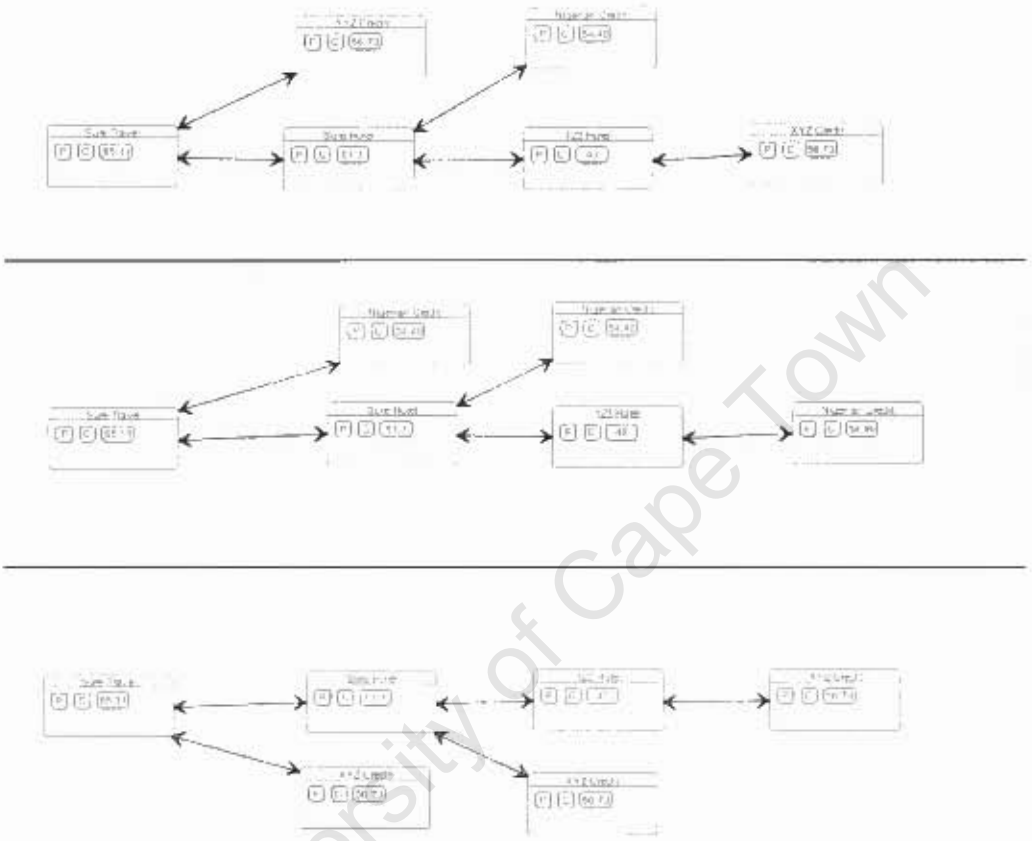

Figure 5.16: TEF graphic showing some possible transaction paths.

5.5 Comparison with other assurance models

As introduced in chapter 2, there are technologies that exist to provide user's with assurances when dealing in online transactions. Examples include digital certificates, federated identity management systems being developed to allow users to authenticate themselves to domains of services and privacy assurance companies such as TRUSTe or BBBOnLine that certify web sites and provide a seal of approval for a fee and provide assurances to the customer that they have evaluated the business policies.

Whilst these all provide specific assurances to the customer, each system is limited and in most cases, were created to benefit the companies that use them. The trust assurance protocol was developed because of a perceived lack of assurance provision for the customer in online transactions. Web services complicate providing assurances because Web services are designed to be modular software artifacts that can be strung together to perform complex business processes. A customer should be able to determine who is involved in their transaction and what those

companies intend to do with information they receive. The trust assurance protocol provides the power of information and choice to the customer, by compelling Web services to provide information that, by law, they should provide. Once the customer has more information about the transaction and the services involved, the service can make an informed choice about which transaction path to choose in order to conduct his online transaction.

The assurance methods above all fall short of providing the customer with the entire picture of a transaction. It is quite possible for a reputable service to use a third party service that has different policies to the primary service. The customer has no way to establish this information from a certificate, a federated identity system, or even the privacy seal companies. The trust assurance protocol fills a gap in the current assurance environment and shows what assurances could be harnessed from an assurance protocol. It should be noted that the trust assurance protocol provides mechanisms for any of the technologies above to be included in the protocol, thereby providing all the benefits of the respective technologies, as well as providing the customer with more information to base his decisions on.

CHAPTER 6

DISCUSSION AND CONCLUSIONS

6.1 INTRODUCTION

The need for assurances in online transactions are required because benevolent services are indistinguishable from malicious services when just their external appearance is evaluated. Both types of Web services provide the same endpoints and both create and consume well constructed SOAP messages. Assurances therefore need to be explicit and more detailed, so that all parties concerned can be made aware of possible risks involved in a transaction.

This dissertation focused on providing assurances to a customer of a Web services transaction. The customer is provided with assurances about the trustworthiness of a Web service, and assurances about how the (trustworthy) services will use his/her personal information during and after a transaction. Legislation governing electronic transactions provides a sufficient base on which a trust assurance protocol can be based. It minimizes the assumptions about what information the customer should have access to and allows a complete protocol to be structured around set regulations. The trust assurance protocol allows the customer to be sure of the services involved in his/her transaction and allows the customer the freedom to choose the path that is most suitable to the customer.

A trust framework, represented by several artifacts, was created to enable the formation of a trust assurance protocol. The Transaction execution framework (TEF) provides a simulation environment in which Web services can be deployed and monitored so that their interactions can be evaluated. The ontologies provide the necessary vocabularies so that all the parties involved in the transaction start form a common base and reduces the negative impact of ambiguity. The trust model developed for this dissertation extends an existing concept of trust to incorporate trust parameters that can be quantified and used to rate services against each other. As with any trust model, the results are subjective because there trust cannot be concisely defined for every type of interaction where trust is required.

6.2 ARTIFACTS

The artifacts developed in this dissertation were created to provide a solution to this dissertation's main aim: to provide a protocol that allows a customer to enter into transactions knowing

which services are involved, and what will happen with his/her personal information during and after the transaction. The vocabularies, protocols and software were all developed to provide functionality that an assurance protocol required. The vocabularies were required to ensure that the client had the right information at his/her disposal, and that the client and services could communicate without fear that ambiguity would jeopardise their interaction. The software was developed to implement the protocol and shows what is required from both the services and the client to fully utilize the trust assurance protocol.

6.2.1 Protocols

The trust assurance protocol consists of two sections; Requesting the transaction chains that can be used to complete a transaction, and determining the trust rating of a service by querying various external sources.

In Chapter 3.3, the trust assurance protocol was described and in Chapter 5 the results from using the protocol was shown and discussed. The trust assurance protocol is the main artifact resulting from this dissertation, as all the other artifacts were created to facilitate the creation and the execution of the protocol.

The protocols were designed to address this dissertation's main goal, namely, to provide the customer with control over his/her transaction. From the evaluation and results chapter, it can be seen that the protocol does provide the client with control over the choice of Web services to use in a transaction, and also provides the client with trust assurances about the (composite) services that he/she chooses. The protocol fulfills the aims of this dissertation to allow the customer to enter into online transactions with confidence.

6.2.2 Software

The transaction execution framework and the client prototype was created to show how the trust assurance protocol works. TEF was created to be a general Web services simulation environment that can be extended so that different Web service deployment architectures, Web services and protocols can be simulated. The modularity of TEF also allows additional modules to be created that can add new features to TEF to enhance is applicability to simulate and monitor Web services.

The client protocol provided the client interaction that the trust assurance protocol was designed to provide. The goal of this dissertation was to allow the customer to have power over a Web service transaction. However, it is not possible for a customer to harness the power of the trust assurance protocol without exposing the customer to the complexities of the Web services transaction chain. The prototype was designed to show what was expected from the user and how a customer could interact with the trust assurance protocol.

6.3 Future Work

There are many aspects of trust assurance that need to be addressed in order to provide a safe environment online in which to conduct transaction. The trust assurance protocol described in this dissertation needs to be adopted as a specification allowing services to implements the

protocol to provide the functionality that client's would need to utilize the trust assurance protocol. The standardization process involved in creating a standard from a specification will also increase the robustness of the specification, and will tie it more tightly to the rest of the Web services stack.

The vocabularies created in this dissertation should be adopted as an open standard and controlled by various bodies, including an unbiased third party to govern the addition of industries, authorities and legislation as the online environment evolves to cater for ever more complex transactions. Such a vocabulary will increase the ability to detect phishing attacks and malicious services whilst maintaining independence from any one country or company controlling the entire architecture.

BIBLIOGRAPHY

[1] Companies act no. 61 of 1973 of south africa, 1973.

[2] Alfarez Abdul-Rahman and Stephen Hailes. A distributed trust model. In *Proceedings of the New Security Paradigms Workshop (NSPW-97)*, pages 48–60, New York, September 23–26 1997. ACM.

[3] Alfarez Abdul-rahman and Stephen Hailes. Supporting trust in virtual communities (reformatted), May 28 2000.

[4] Colin Boyd Audun Jøsang, Roslan Ismail. A survey of trust and reputation systems for online service provision. 2005.

[5] Patrick Bajari and Ali Hortacsu. Winner's curse, reserve prices and endogenous entry: Empirical insights from ebay auctions, February 10 2000.

[6] J. Barney and M. Hansen. Trustworthiness as a source of competitive advantage. *Strategic Management Journal*, 15:175–190, 1994.

[7] David L. Baumer, Julia Brande Earp, and J. C. Poindexter. Internet privacy law: a comparison between the united states and the european union. *Computers & Security*, 23(5):400–412, 2004.

[8] T. Belwood and et al. UDDI version 3.0. http://uddi.org/pubs/uddi_v3.htm, 2000.

[9] Rajeev Bhattacharya, Timothy M. Devinney, and Madan M. Pillutla. A formal model of trust based on outcomes. In *The Academy of Management Review*, volume 23, pages 459–472. JSTOR, 1998.

[10] Blaze, Feigenbaum, and Lacy. Decentralized trust management. In *RSP: 17th IEEE Computer Society Symposium on Research in Security and Privacy*, 1996.

[11] Angela Bonifati and Stefano Ceri. Comparative analysis of five XML query languages. *ACM SIGMOD Record*, 29(1):68–79, March 2000.

[12] Cristiano Castelfranchi, Rosaria Conte, and Mario Paolucci. Normative reputation and the costs of compliance. *J. Artificial Societies and Social Simulation*, 1(3), 1998.

[13] Roberto Chinnici, Jean-Jacques Moreau, Arthur Ryman, and Sanjiva Weerawarana. Web services description language (WSDL) version 2.0 part 1: Core language. World Wide Web Consortium, Candidate Recommendation CR-wsdl20-20060327, March 2006.

[14] Cranor. 'I didn't buy it for myself' privacy and ecommerce personalization. In *ACM Workshop on Privacy in the Electronic Society*, 2003.

[15] European Parliament and Council. Directive 95/46/EC of the European Parliament and of the Council, October 1995.

[16] European Parliament and Council. Directive 2002/58/EC of the European Parliament and of the Council, July 2002.

[17] Batya Friedman, Peter H. Kahn, Jr., and Daniel C. Howe. Trust online. *CACM*, 43(12):34–40, December 2000.

[18] Diego Gambetta. Can we trust trust?, July 09 1988.

[19] Martin Gudgin, Marc Hadley, Noah Mendelsohn, Jean-Jacques Moreau, and Henrik Frystyk Nielsen. Soap version 1.2. Technical report, W3C, 2003.

[20] R.W. Hahn. An assessment of the costs of proposed online privacy legislation. AEI-Brookings Joint Center for Regulatory Studies, May 2001.

[21] R. Housley, W. Polk, W. Ford, and D. Solo. Internet x.509 public key infrastructure certificate and certificate revocation list (crl) profile. RFC 3280.

[22] IBM, Microsoft, Actional, BEA, Computer Associates, Layer 7, Oblix, OpenNetwork, Ping Identity amd Reactivity, and Verisign. Web services trust language (ws-trust). online, February 2005.

[23] BEA IBM and SAP. Web servies policy attachment(ws-policyattachment). http://www.verisign.com/wss/WS-PolicyAttachment.pdf, 2006.

[24] Microsoft Arjuna Hitachi IONA IBM, BEA Systems. Web services transactions specification (ws-transaction), August 2005.

[25] International Organization for Standardization. *ISO 8879:1986: Information processing — Text and office systems — Standard Generalized Markup Language (SGML).* August 1986.

[26] David R. Johnson and David G. Post. Law and borders - the rise of law in cyberspace. *First Monday*, 1(1), 1996.

[27] Audun Jøsang. The right type of trust for distributed systems. In *Proceedings on the Workshop on New Security Paradigms*, pages 119–131, New York, September 17–20 1997. ACM Press.

[28] Pradip Lamsal. Understanding trust and security, January 16 2001.

[29] B Mahadevan. Business models for internet based e-commerce: An anatomy. In *CALIFORNIA MANAGEMENT REVIEW*, volume 42, pages 55–69. UNIVERSITY OF CALIFORNIA, 2000.

[30] R.C. MAYER, J.H. DAVIS, and F.D. SCHOORMAN. An integrative model of organizational trust. *ACAD MANAGE REV*, 20:709–734, July 1995.

[31] Deborah L. McGuinness and Frank van Harmelen. OWL web ontology language overview. World Wide Web Consortium, Recommendation REC-owl-features-20040210, February 2004.

[32] Sheila A. McIlraith, Tran Cao Son, and Honglei Zeng. Semantic web services. *IEEE Intelligent Systems*, 16(2):46–53, 2001.

[33] D.H. McKnight and Norman L. Chervany. The meaning of trust. Technical report, MISRC, University of Minnesota, Management Information Systems Reseach Center, 96.

[34] L. Mui, M. Mohtashemi, and A. Halberstadt. A computational model of trust and reputation for e-businesses. In *HICSS '02: Proceedings of the 35th Annual Hawaii International Conference on System Sciences (HICSS'02)-Volume 7*, page 188, Washington, DC, USA, 2002. IEEE Computer Society.

[35] Anthony Nadalin, Chris Kaler, Phillip Hallam-Baker, and Ronald Monzillo. Soap message security 1.0 (ws-security 2004). Technical report, OASIS, 2004.

[36] James R. Otto and Q.B. Chung. A framework for cyber-enhanced retailing: Integrating e-commerce retailing with brick-and-mortar retailing. In *Electronic Markets*, volume 10, pages 185–191. Routledge, 2000.

[37] Parliament of the Republic of South Africa. Electronic Communications and Transactions Act, July 2002.

[38] Dave Raggett, Arnaud Le Hors, and Ian Jacobs. Html 4.01 specification, December 1999.

[39] Lars Rasmusson and Sverker Jansson. Simulated social control for secure Internet commerce. In *Proceedings on the Workshop on New Security Paradigms*, pages 18–26, New York, September 17–20 1997. ACM Press.

[40] Pauline Ratnasingham and Kuldeep Kumar. Trading partner trust in electronic commerce participation. In *ICIS*, pages 544–552, 2000.

[41] P. Resnick, R. Zeckhauser, J. Swanson, and K. Lockwood. The value of reputation on ebay: A controlled experiment, 2003.

[42] Paul Resnick, Richard Zeckhauser, John Swanson, and Kate Lockwood. The value of reputation on ebay: A controlled experiment, June 03 2003.

[43] Jothy Rosenberg and David Remy. *Securing Web Services with WS-Security*. Sams Publishing, 2004.

[44] Jeffrey Schlimmer. Web services policy framework (ws-policy). Technical report, IBM, Microsoft, BEA Systems, SAP, Verisign, Sonic Software, 2004.

[45] W. T. Luke Teacy, Jigar Patel, Nicholas R. Jennings, and Michael Luck. Coping with inaccurate reputation sources: experimental analysis of a probabilistic trust model. In *AAMAS '05: Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, pages 997–1004, New York, NY, USA, 2005. ACM Press.

[46] Anitta Thomas and Lucas Venter. Propagatin trust in the web services framework. In *Information Security South Africa*, 2004.

[47] United Nations Commission on International Trade Law. UNCITRAL Model Law on Electronic Commerce, 1996.

[48] U.S. Department of Commerce. Safe Harbour Framework. http://www.export.gov/safeharbor/, 2000.

[49] Eric M. Uslaner. Trust online, trust offline. *Commun. ACM*, 47(4):28–29, 2004.

[50] Asir S. Vedamuthu. Web services description language (WSDL) version 2.0 SOAP 1.1 binding. World Wide Web Consortium, Working Draft WD-wsdl20-soap11-binding-20060327, March 2006.

[51] W3C. Web services choreography interface (wsci) version 1.0. http://www.w3.org/TR/2002/NOTE-wsci-20020808, August 2002.

[52] OE Williamson. Calculativeness, trust and economic organization. *Journal of Law and Economics*, 36(1):453–486, 1993.

[53] XML schema 1.1, W3C recommendation, 2006. http://www.w3c.org/XML/Schema.

[54] François Yergeau, Eve Maler, Tim Bray, Jean Paoli, and C. M. Sperberg-McQueen. Extensible markup language (XML) 1.0 (fourth edition). W3C recommendation, W3C, August 2006. http://www.w3.org/TR/2006/REC-xml-20060816.

CHAPTER 7

APPENDIX

7.1 EXAMPLE POLICY DOCUMENT

The XML document shown is a complete sample policy document.

```
<?xml version="1.0"
encoding="UTF-8"?> <Policy
xmlns="http://dna.cs.uct.ac.za/TrustAssurance"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://dna.cs.uct.ac.za/TrustAssurance
C:\masters\masters\schema\policy.xsd">
 <EntityName>Sure Travel Company</EntityName>
    <LegalStatus>Partnership</LegalStatus>
 <CompanyType>Travel_and_Leisure</CompanyType>
 <ContactDetails>
  <Address>
   <Physical>
    <AddressDetails>
     <StreetAddress>
      <number>123</number>
      <street>Sure Street</street>
     </StreetAddress>
     <Town>Rondebosch</Town>
     <City>Cape Town</City>
     <Country>South Africa</Country>
     <PostCode>7700</PostCode>
    </AddressDetails>
   </Physical>
   <Physical>
    <AddressDetails>
     <POBox>12345</POBox>
     <Town>Rondebosch</Town>
     <City>Cape Town</City>
     <Country>South Africa</Country>
     <PostCode>7700</PostCode>
    </AddressDetails>
   </Physical>
   <Legal>
    <AddressDetails>
     <StreetAddress>
      <number>123</number>
      <street>Sure Street</street>
     </StreetAddress>
     <Town>Rondebosch</Town>
     <City>Cape Town</City>
     <Country>South Africa</Country>
     <PostCode>7700</PostCode>
    </AddressDetails>
   </Legal>
  </Address>
  <URI>
   <Website>http://www.suretravel.co.za</Website>
   <PolicyURL>http://www.suretravel.co.za/policy.xml</PolicyURL>
   <Service>http://192.168.8.20:9762/axis2/suretravel</Service>
  </URI>
  <PhoneNumbers>
   <NumberDetails>
    <Description>Office Phone</Description>
    <Number>555 1234</Number>
   </NumberDetails>
   <NumberDetails>
    <Description>Office Fax</Description>
    <Number>555 1235</Number>
```

```
  </NumberDetails>
 </PhoneNumbers>
</ContactDetails>
<JurisdictionInformation>
 <JurisdictionItem>
  <JurisdictionDescription>Territorial</JurisdictionDescription>
  <JurisdictionValue>South Africa</JurisdictionValue>
 </JurisdictionItem>
</JurisdictionInformation>
<ProductInformation>
 <ProductName>Travel Packages</ProductName>
 <ProductDescription>We provide complete travel packages</ProductDescription>
</ProductInformation>
<Conditions>
 <TermsOfAgreement>
  <ExchangePolicy>
   <ExchangePolicyElement>
    <PolicyItemName>NoExchanges</PolicyItemName>
    <PolicyItemDescription>TravelBookings can not be exchanged</PolicyItemDescription>
    <PolicyItemValue>true</PolicyItemValue>
   </ExchangePolicyElement>
  </ExchangePolicy>
  <RefundPolicy>
   <RefundPolicyElement>
    <PolicyItemName>Refund</PolicyItemName>
    <PolicyItemDescription>Refunds will be given in full if package is cancelled 3 weeks in advance</PolicyItemDescription>
    <PolicyItemValue>3 weeks from departure date</PolicyItemValue>
   </RefundPolicyElement>
  </RefundPolicy>
  <SecurityProceduresDescription>
   <SecurityProcedureName>NoExtraSecurity</SecurityProcedureName>
   <SecurityProcedureDescription>No Extraneous Security requirements in place</SecurityProcedureDescription>
  </SecurityProceduresDescription>
 </TermsOfAgreement>
 <PrivacyPolicy>
  <PrivacyPolicyItem>
   <PolicyItemName>ProvidedInformationCollection</PolicyItemName>
   <PolicyItemDescription>We collect infomation that is supplied to us</PolicyItemDescription>
   <PolicyItemValue>true</PolicyItemValue>
  </PrivacyPolicyItem>
  <PrivacyPolicyItem>
   <PolicyItemName>LogInformation</PolicyItemName>
   <PolicyItemDescription>We automatically log usage information, including IP address, date and time</PolicyItemDescription>
   <PolicyItemValue>true</PolicyItemValue>
  </PrivacyPolicyItem>
 </PrivacyPolicy>
 <PaymentPolicy>
  <PaymentInformation>
   <PaymentInformationName>Credit Cards Accepted</PaymentInformationName>
   <PaymentInformationDescription>We Accept these Credit Cards</PaymentInformationDescription>
   <PaymentInformationValue>VISA</PaymentInformationValue>
   <PaymentInformationValue>MasterCard</PaymentInformationValue>
   <PaymentInformationValue>Diners Club</PaymentInformationValue>
   <PaymentInformationValue>American Express</PaymentInformationValue>
  </PaymentInformation>
  <PaymentInformation>
   <PaymentInformationName>Surgcharges</PaymentInformationName>
   <PaymentInformationDescription>We do not charge a fee for credit card transactions</PaymentInformationDescription>
   <PaymentInformationValue>None</PaymentInformationValue>
  </PaymentInformation>
  <PaymentInformation>
   <PaymentInformationName>Currency</PaymentInformationName>
   <PaymentInformationDescription>We accept the following currencies</PaymentInformationDescription>
   <PaymentInformationValue>ZAR</PaymentInformationValue>
   <PaymentInformationValue>USD</PaymentInformationValue>
  </PaymentInformation>
  <PaymentInformation>
   <PaymentInformationName>PurchaseOrders</PaymentInformationName>
   <PaymentInformationDescription>Information regarding purchase orders</PaymentInformationDescription>
   <PaymentInformationValue>http://www.suretravel.co.za/FAQ/purchase-orders.html</PaymentInformationValue>
  </PaymentInformation>
 </PaymentPolicy>
 <PersonalInformation>
  <BrokerInformation>
   <InformationItems>
    <ItemName>CreditCard</ItemName>
   </InformationItems>
   <PII>
    <SpecificInformation>
     <PersonalInformationItem itemName="creditCard">
      <UsagePolicy>
       <UsagePolicyItem>
        <PolicyItemName>NotUsed</PolicyItemName>
        <PolicyItemDescription>CreditCard information is not used by SureTravel. When you pay us for a service, we use a merchant
          bank to faciliate payment</PolicyItemDescription>
       </UsagePolicyItem>
      </UsagePolicy>
      <StoragePolicy requiredByLaw="true">
       <StoragePolicyItem>
        <PolicyItemName>EncryptedStorage</PolicyItemName>
        <PolicyItemDescription>SureTravel has to strore transaction details as it is required by law. Credit card information is
          securely stored using appropriate encryption technologies</PolicyItemDescription>
        <Period>1 year</Period>
       </StoragePolicyItem>
```

```
    </StoragePolicy>
    <SharingPolicy>
     <SharingPolicyItem>
      <PolicyItemName>NoSharing</PolicyItemName>
      <PolicyItemDescription>SureTravel does not share your credit card details with any affiliates or 3rd parties, other than
        those to faciliate the specific transaction</PolicyItemDescription>
     </SharingPolicyItem>
    </SharingPolicy>
   < PersonalInformationItem>
  </SpecificInformation>
 </PII>
</BrokerInformation>
<EndClientInformation>
 <InformationItems>
  <ItemName>FullName</ItemName>
  <ItemName>BirthDate</ItemName>
  <ItemName>Gender</ItemName>
 </InformationItems>
 <PII>
  <SpecificInformation>
   <PersonalInformationItem itemName="FullName">
    <UsagePolicy>
     <UsagePolicyItem>
      <PolicyItemName>InternalUseOnly</PolicyItemName>
      <PolicyItemDescription>We collect your full name in order to process the transaction.</PolicyItemDescription>
     </UsagePolicyItem>
    </UsagePolicy>
    <StoragePolicy requiredByLaw="yes">
     <StoragePolicyItem>
      <PolicyItemName>InternalStorage</PolicyItemName>
      <PolicyItemDescription>We store all information we receive from you as a record of our transaction</PolicyItemDescription>
      <Period>1 year</Period>
     </StoragePolicyItem>
    </StoragePolicy>
    <SharingPolicy>
     <SharingPolicyItem>
      <PolicyItemName>ShareWithAffiliates</PolicyItemName>
      <PolicyItemDescription>From Time to time we share information collected about you with our affiliates and sister
        companies in order to provide better services to you</PolicyItemDescription>
      <OtherCompany>
       <CompanyName>SureHotels</CompanyName>
       <CompanyType>Travel_Leisure_company</CompanyType>
       <ContactDetails>
        <URI>
         <Website>http://www.surehotels.co.za</Website>
        </URI>
       </ContactDetails>
       <CompanyAffiliation>We are part of the same group of companies</CompanyAffiliation>
      </OtherCompany>
     </SharingPolicyItem>
    </SharingPolicy>
   </PersonalInformationItem>
   <PersonalInformationItem itemName="BirthDate">
    <UsagePolicy>
     <UsagePolicyItem>
      <PolicyItemName>InternalUseOnly</PolicyItemName>
      <PolicyItemDescription>We collect your BirthDate in order to process the transaction by checking to see whether you \
        are eligible for discounts based on your age.</PolicyItemDescription>
     </UsagePolicyItem>
    </UsagePolicy>
    <StoragePolicy requiredByLaw="yes">
     <StoragePolicyItem>
      <PolicyItemName>InternalStorage</PolicyItemName>
      <PolicyItemDescription>We store all information we receive from you as a record of our transaction</PolicyItemDescription>
      <Period>1 year</Period>
     </StoragePolicyItem>
    </StoragePolicy>
    <SharingPolicy>
     <SharingPolicyItem>
      <PolicyItemName>ShareWithAffiliates</PolicyItemName>
      <PolicyItemDescription>From Time to time we share information collected about you with our affiliates and
        sister companies in order to provide better services to you</PolicyItemDescription>
      <OtherCompany>
       <CompanyName>SureHotels</CompanyName>
       <CompanyType>Travel_Leisure_company</CompanyType>
       <ContactDetails>
        <URI>
         <Website>http://www.surehotels.co.za</Website>
        </URI>
       </ContactDetails>
       <CompanyAffiliation>We are part of the same group of companies</CompanyAffiliation>
      </OtherCompany>
     </SharingPolicyItem>
    </SharingPolicy>
   </PersonalInformationItem>
   <PersonalInformationItem itemName="Gender">
    <UsagePolicy>
     <UsagePolicyItem>
      <PolicyItemName>InternalUseOnly</PolicyItemName>
      <PolicyItemDescription>We collect Gender information to provide a better service to our clients.</PolicyItemDescription>
     </UsagePolicyItem>
    </UsagePolicy>
    <StoragePolicy requiredByLaw="yes">
     <StoragePolicyItem>
```

```
        <PolicyItemName>InternalStorage</PolicyItemName>
        <PolicyItemDescription>We store all information we receive from you as a record of our transaction</PolicyItemDescription>
        <Period>1 year</Period>
       </StoragePolicyItem>
      </StoragePolicy>
      <SharingPolicy>
       <SharingPolicyItem>
        <PolicyItemName>ShareWithAffiliates</PolicyItemName>
        <PolicyItemDescription>From Time to time we share information collected about you with our affiliates and
          sister companies in order to provide better services to you</PolicyItemDescription>
        <OtherCompany>
         <CompanyName>SureHotels</CompanyName>
         <CompanyType>Travel_Leisure_company</CompanyType>
         <ContactDetails>
          <URI>
           <Website>http://www.surehotels.co.za</Website>
          </URI>
         </ContactDetails>
         <CompanyAffiliation>We are part of the same group of companies</CompanyAffiliation>
        </OtherCompany>
       </SharingPolicyItem>
      </SharingPolicy>
     </PersonalInformationItem>
    </SpecificInformation>
   </PII>
  </EndClientInformaton>
 </PersonalInformation>
</Conditions>
</Policy>
```

7.2 POLICY SCHEMA

This schema is used to create policy documents for services.

```
<?xml version="1.0" encoding="UTF-8"?> <xs:schema
targetNamespace="http://dna.cs.uct.ac.za/TrustAssurance"
elementFormDefault="qualified" attributeFormDefault="unqualified"
xmlns:con="http://dna.cs.uct.ac.za/TrustAssurance"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
 <xs:include schemaLocation="C:\masters\masters\schema\contact.xsd"/>
 <xs:include schemaLocation="C:\masters\masters\schema\InformationItem.xsd"/>
 <xs:element name="Policy">
  <xs:complexType>
   <xs:sequence>
    <xs:element name="EntityName">
     <xs:simpleType>
      <xs:list itemType="xs:string"/>
     </xs:simpleType>
    </xs:element>
    <xs:element name="LegalStatus" type="xs:string"/>
    <xs:element name="CompanyType" type="xs:string"/>
    <xs:element name="RegistrationNumbers" minOccurs="0">
     <xs:complexType>
      <xs:sequence>
       <xs:element name="RegistrationDetails" maxOccurs="unbounded">
        <xs:complexType>
         <xs:sequence>
          <xs:element name="Description" type="xs:string"/>
          <xs:element name="Number" type="xs:string"/>
          <xs:element name="RegistrationAuthority" minOccurs="0">
           <xs:complexType>
            <xs:sequence>
             <xs:element name="AuthorityName" type="xs:string"/>
             <xs:element ref="con:ContactDetails"/>
            </xs:sequence>
           </xs:complexType>
          </xs:element>
         </xs:sequence>
        </xs:complexType>
       </xs:element>
      </xs:sequence>
     </xs:complexType>
    </xs:element>
    <xs:element ref="con:ContactDetails"/>
    <xs:element name="JurisdictionInformation">
     <xs:complexType>
      <xs:sequence>
       <xs:element name="JurisdictionItem" maxOccurs="unbounded">
        <xs:complexType>
         <xs:sequence>
          <xs:element name="JurisdictionDescription">
           <xs:simpleType>
            <xs:restriction base="xs:string">
             <xs:enumeration value="Territorial"/>
             <xs:enumeration value="Municipal"/>
             <xs:enumeration value="SubjectMatter"/>
```

```
        </xs:restriction>
       </xs:simpleType>
      </xs:element>
      <xs:element name="JurisdictionValue" type="xs:string"/>
      <xs:element name="AuthorityDetails" minOccurs="0">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="AuthorityName" type="xs:string"/>
         <xs:element ref="con:ContactDetails"/>
        </xs:sequence>
       </xs:complexType>
      </xs:element>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="Affiliations" minOccurs="0">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="RegulatoryBodies" maxOccurs="unbounded">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="Name" type="xs:string"/>
      <xs:element name="Description" type="xs:string"/>
      <xs:element ref="con:ContactDetails"/>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
   <xs:element name="AccreditationBodies" maxOccurs="unbounded">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="Name" type="xs:string"/>
      <xs:element name="Description" type="xs:string"/>
      <xs:element ref="con:ContactDetails"/>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="ProductInformation">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="ProductName" type="xs:string"/>
   <xs:element name="ProductDescription" type="xs:string"/>
   <xs:element name="ProductCost" minOccurs="0">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="TotalCost">
       <xs:complexType>
        <xs:simpleContent>
         <xs:extension base="xs:decimal">
          <xs:attribute name="currency" type="xs:string" use="optional"/>
         </xs:extension>
        </xs:simpleContent>
       </xs:complexType>
      </xs:element>
      <xs:element name="Taxes" minOccurs="0" maxOccurs="unbounded">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="TaxName" type="xs:string"/>
         <xs:element name="TaxValue" type="xs:string"/>
         <xs:element name="TaxCost" type="xs:decimal"/>
        </xs:sequence>
        <xs:attribute name="currency" type="xs:string" use="optional"/>
       </xs:complexType>
      </xs:element>
      <xs:element name="transportCosts" minOccurs="0" maxOccurs="unbounded">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="transportDescription" type="xs:string"/>
         <xs:element name="transportCost" type="xs:decimal"/>
        </xs:sequence>
        <xs:attribute name="currency" type="xs:string" use="optional"/>
       </xs:complexType>
      </xs:element>
      <xs:element name="OtherCosts" minOccurs="0" maxOccurs="unbounded">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="FeesDescription" type="xs:string"/>
         <xs:element name="FeesCost" type="xs:decimal"/>
        </xs:sequence>
        <xs:attribute name="currency" type="xs:string" use="optional"/>
       </xs:complexType>
      </xs:element>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
   <xs:element name="PaymentOptions" minOccurs="0">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="PaymentType" maxOccurs="unbounded">
```

```
        <xs:complexType>
         <xs:sequence>
          <xs:element name="BankDetails" minOccurs="0" maxOccurs="unbounded">
           <xs:complexType>
            <xs:sequence>
             <xs:element name="AccountNumber" type="xs:string"/>
             <xs:element name="BankName" type="xs:string"/>
             <xs:element name="BranchCode" type="xs:string"/>
             <xs:element name="BankContactDetails">
              <xs:complexType>
               <xs:sequence>
                <xs:element ref="con:ContactDetails"/>
               </xs:sequence>
              </xs:complexType>
             </xs:element>
            </xs:sequence>
           </xs:complexType>
          </xs:element>
          <xs:element name="CreditCardFacilities" minOccurs="0" maxOccurs="unbounded">
           <xs:complexType>
            <xs:sequence>
             <xs:element name="MerchantName" type="xs:string"/>
             <xs:element ref="con:ContactDetails"/>
            </xs:sequence>
           </xs:complexType>
          </xs:element>
          <xs:element name="PaymentMethod" maxOccurs="unbounded">
           <xs:complexType>
            <xs:sequence>
             <xs:element name="PaymentProcessor" type="xs:string"/>
             <xs:element name="PaymentDescription" type="xs:string"/>
            </xs:sequence>
           </xs:complexType>
          </xs:element>
         </xs:sequence>
        </xs:complexType>
       </xs:element>
      </xs:sequence>
     </xs:complexType>
    </xs:element>
   </xs:sequence>
  </xs:complexType>
 </xs:element>
 <xs:element name="Conditions">
  <xs:complexType>
   <xs:sequence>
    <xs:element name="TermsOfAgreement">
     <xs:complexType>
      <xs:sequence>
       <xs:element name="Guarantees" minOccurs="0">
        <xs:complexType>
         <xs:sequence>
          <xs:element name="GuaranteeElement" maxOccurs="unbounded">
           <xs:complexType>
            <xs:sequence>
             <xs:element name="GuaranteeName" type="xs:string"/>
             <xs:element name="GuaranteeDecription" type="xs:string"/>
             <xs:element name="ValidUntil" type="xs:date" minOccurs="0"/>
            </xs:sequence>
           </xs:complexType>
          </xs:element>
         </xs:sequence>
        </xs:complexType>
       </xs:element>
       <xs:element name="Warranties" minOccurs="0">
        <xs:complexType>
         <xs:sequence>
          <xs:element name="WarrantyElement" maxOccurs="unbounded">
           <xs:complexType>
            <xs:sequence>
             <xs:element name="WarrantyName" type="xs:string"/>
             <xs:element name="WarrantyDecription" type="xs:string"/>
             <xs:element name="ValidUntil" type="xs:date" minOccurs="0"/>
            </xs:sequence>
           </xs:complexType>
          </xs:element>
         </xs:sequence>
        </xs:complexType>
       </xs:element>
       <xs:element name="ServiceRenderTimeFrame" minOccurs="0">
        <xs:complexType>
         <xs:sequence>
          <xs:element name="CommencmentDate" type="xs:date" minOccurs="0"/>
          <xs:element name="ValidUntil" type="xs:date" minOccurs="0"/>
         </xs:sequence>
        </xs:complexType>
       </xs:element>
       <xs:element name="UserAccountInformation" minOccurs="0">
        <xs:complexType>
         <xs:sequence>
          <xs:element name="AccessInformation">
           <xs:complexType>
            <xs:sequence maxOccurs="unbounded">
             <xs:element name="AccessInformationDescription" type="xs:string"/>
```

```
          <xs:element name="AccessInformationValue" type="xs:string"/>
         </xs:sequence>
        </xs:complexType>
       </xs:element>
       <xs:element name="MaintenanceInformation">
        <xs:complexType>
         <xs:sequence maxOccurs="unbounded">
          <xs:element name="MaintenanceInformationDescription" type="xs:string"/>
          <xs:element name="MaintenanceInformationValue" type="xs:string"/>
         </xs:sequence>
        </xs:complexType>
       </xs:element>
      </xs:sequence>
     </xs:complexType>
    </xs:element>
    <xs:element name="TransactionInformation" minOccurs="0">
     <xs:complexType>
      <xs:sequence maxOccurs="unbounded">
       <xs:element name="AccessInformation">
        <xs:complexType>
         <xs:sequence maxOccurs="unbounded">
          <xs:element name="AccessInformationDescription" type="xs:string"/>
          <xs:element name="AccessInformationValue" type="xs:string"/>
         </xs:sequence>
        </xs:complexType>
       </xs:element>
       <xs:element name="MaintenanceInformation">
        <xs:complexType>
         <xs:sequence maxOccurs="unbounded">
          <xs:element name="MaintenanceInformationDescription" type="xs:string"/>
          <xs:element name="MaintenanceInformationValue" type="xs:string"/>
         </xs:sequence>
        </xs:complexType>
       </xs:element>
      </xs:sequence>
     </xs:complexType>
    </xs:element>
    <xs:element name="ExchangePolicy">
     <xs:complexType>
      <xs:sequence>
       <xs:element name="ExchangePolicyElement" maxOccurs="unbounded">
        <xs:complexType>
         <xs:sequence>
          <xs:element name="PolicyItemName" type="xs:string"/>
          <xs:element name="PolicyItemDescription" type="xs:string"/>
          <xs:element name="PolicyItemValue" type="xs:string" minOccurs="0" />
         </xs:sequence>
        </xs:complexType>
       </xs:element>
      </xs:sequence>
     </xs:complexType>
    </xs:element>
    <xs:element name="RefundPolicy">
     <xs:complexType>
      <xs:sequence>
       <xs:element name="RefundPolicyElement" maxOccurs="unbounded">
        <xs:complexType>
         <xs:sequence>
          <xs:element name="PolicyItemName" type="xs:string"/>
          <xs:element name="PolicyItemDescription" type="xs:string"/>
          <xs:element name="PolicyItemValue" type="xs:string" minOccurs="0"/>
         </xs:sequence>
        </xs:complexType>
       </xs:element>
      </xs:sequence>
     </xs:complexType>
    </xs:element>
    <xs:element name="DisputeResolutionCodes" minOccurs="0">
     <xs:complexType>
      <xs:sequence>
       <xs:element name="DisputeResolutionDetails" maxOccurs="unbounded">
        <xs:complexType>
         <xs:sequence>
          <xs:element name="CodeName" type="xs:string"/>
          <xs:element name="CodeDetails" type="xs:string"/>
          <xs:element ref="con:ContactDetails"/>
         </xs:sequence>
        </xs:complexType>
       </xs:element>
      </xs:sequence>
     </xs:complexType>
    </xs:element>
    <xs:element name="SecurityProceduresDescription">
     <xs:complexType>
      <xs:sequence>
       <xs:element name="SecurityProcedureName" type="xs:string"/>
       <xs:element name="SecurityProcedureDescription" type="xs:string"/>
       <xs:element name="SecurityProcedureValue" type="xs:string" minOccurs="0"/>
      </xs:sequence>
     </xs:complexType>
    </xs:element>
   </xs:sequence>
  </xs:complexType>
 </xs:element>
```

```
<xs:element name="PrivacyPolicy">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="PrivacyPolicyItem" maxOccurs="unbounded">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="PolicyItemName" type="xs:string"/>
      <xs:element name="PolicyItemDescription" type="xs:string"/>
      <xs:element name="PolicyItemValue" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="PaymentPolicy">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="PaymentInformation" maxOccurs="unbounded">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="PaymentInformationName" type="xs:string"/>
      <xs:element name="PaymentInformationDescription" type="xs:string"/>
      <xs:element name="PaymentInformationValue" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="PersonalInformation">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="BrokerInformation">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="InformationItems">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="ItemName" maxOccurs="unbounded">
          <xs:simpleType>
           <xs:restriction base="xs:string">
            <xs:enumeration value="IDNumber"/>
            <xs:enumeration value="UniqueIdentfierNumber"/>
            <xs:enumeration value="FullName"/>
            <xs:enumeration value="FirstName"/>
            <xs:enumeration value="Surname"/>
            <xs:enumeration value="MiddleName"/>
            <xs:enumeration value="NamePrefix"/>
            <xs:enumeration value="Birthdate"/>
            <xs:enumeration value="Gender"/>
            <xs:enumeration value="JobTitle"/>
            <xs:enumeration value="MailingAddress"/>
            <xs:enumeration value="HomeTelephoneNumber"/>
            <xs:enumeration value="TelephoneNumber"/>
            <xs:enumeration value="WorkTelephoneNumber"/>
            <xs:enumeration value="CellphoneNumber"/>
            <xs:enumeration value="FaxNumber"/>
            <xs:enumeration value="EmailAddress"/>
            <xs:enumeration value="URL"/>
            <xs:enumeration value="EmployersName"/>
            <xs:enumeration value="CompanyDepartment"/>
            <xs:enumeration value="CreditCardDetails"/>
            <xs:enumeration value="BankingDetails"/>
            <xs:enumeration value="FinacialInformation"/>
            <xs:enumeration value="ComputerInformation"/>
            <xs:enumeration value="DemographicData"/>
            <xs:enumeration value="PoliticalData"/>
            <xs:enumeration value="HealthData"/>
            <xs:enumeration value="TransactionData"/>
            <xs:enumeration value="ServiceUseData"/>
           </xs:restriction>
          </xs:simpleType>
         </xs:element>
        </xs:sequence>
       </xs:complexType>
      </xs:element>
      <xs:element name="PII" minOccurs="0">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="SpecificInformation" minOccurs="0">
          <xs:complexType>
           <xs:sequence>
            <xs:element name="PersonalInformationItem" maxOccurs="unbounded">
             <xs:complexType>
              <xs:sequence>
               <xs:element name="ItemName">
                <xs:simpleType>
                 <xs:restriction base="xs:string">
                  <xs:enumeration value="IDNumber"/>
                  <xs:enumeration value="UniqueIdentfierNumber"/>
                  <xs:enumeration value="FullName"/>
                  <xs:enumeration value="FirstName"/>
                  <xs:enumeration value="Surname"/>
```

```
                <xs:enumeration value="MiddleName"/>
                <xs:enumeration value="NamePrefix"/>
                <xs:enumeration value="Birthdate"/>
                <xs:enumeration value="Gender"/>
                <xs:enumeration value="JobTitle"/>
                <xs:enumeration value="MailingAddress"/>
                <xs:enumeration value="HomeTelephoneNumber"/>
                <xs:enumeration value="TelephoneNumber"/>
                <xs:enumeration value="WorkTelephoneNumber"/>
                <xs:enumeration value="CellphoneNumber"/>
                <xs:enumeration value="FaxNumber"/>
                <xs:enumeration value="EmailAddress"/>
                <xs:enumeration value="URL"/>
                <xs:enumeration value="EmployersName"/>
                <xs:enumeration value="CompanyDepartment"/>
                <xs:enumeration value="CreditCardDetails"/>
                <xs:enumeration value="BankingDetails"/>
                <xs:enumeration value="FinacialInformation"/>
                <xs:enumeration value="ComputerInformation"/>
                <xs:enumeration value="DemographicData"/>
                <xs:enumeration value="PoliticalData"/>
                <xs:enumeration value="HealthData"/>
                <xs:enumeration value="TransactionData"/>
                <xs:enumeration value="ServiceUseData"/>
               </xs:restriction>
              </xs:simpleType>
             </xs:element>
             <xs:element ref="con:UsagePolicy"/>
             <xs:element ref="con:StoragePolicy"/>
             <xs:element ref="con:SharingPolicy"/>
            </xs:sequence>
            <xs:attribute name="mustProvide" type="xs:string" use="required"/>
           </xs:complexType>
          </xs:element>
         </xs:sequence>
        </xs:complexType>
       </xs:element>
       <xs:element name="GeneralPolicy" minOccurs="0">
        <xs:complexType>
         <xs:sequence>
          <xs:element ref="con:UsagePolicy"/>
          <xs:element ref="con:StoragePolicy"/>
          <xs:element ref="con:SharingPolicy"/>
         </xs:sequence>
        </xs:complexType>
       </xs:element>
      </xs:sequence>
     </xs:complexType>
    </xs:element>
   </xs:sequence>
  </xs:complexType>
 </xs:element>
 <xs:element name="EndClientInformation" minOccurs="0">
  <xs:complexType>
   <xs:sequence>
    <xs:element name="InformationItems">
     <xs:complexType>
      <xs:sequence>
       <xs:element name="ItemName" maxOccurs="unbounded">
        <xs:simpleType>
         <xs:restriction base="xs:string">
          <xs:enumeration value="IDNumber"/>
          <xs:enumeration value="UniqueIdentfierNumber"/>
          <xs:enumeration value="FullName"/>
          <xs:enumeration value="FirstName"/>
          <xs:enumeration value="Surname"/>
          <xs:enumeration value="MiddleName"/>
          <xs:enumeration value="NamePrefix"/>
          <xs:enumeration value="Birthdate"/>
          <xs:enumeration value="Gender"/>
          <xs:enumeration value="JobTitle"/>
          <xs:enumeration value="MailingAddress"/>
          <xs:enumeration value="HomeTelephoneNumber"/>
          <xs:enumeration value="TelephoneNumber"/>
          <xs:enumeration value="WorkTelephoneNumber"/>
          <xs:enumeration value="CellphoneNumber"/>
          <xs:enumeration value="FaxNumber"/>
          <xs:enumeration value="EmailAddress"/>
          <xs:enumeration value="URL"/>
          <xs:enumeration value="EmployersName"/>
          <xs:enumeration value="CompanyDepartment"/>
          <xs:enumeration value="CreditCardDetails"/>
          <xs:enumeration value="BankingDetails"/>
          <xs:enumeration value="FinacialInformation"/>
          <xs:enumeration value="ComputerInformation"/>
          <xs:enumeration value="DemographicData"/>
          <xs:enumeration value="PoliticalData"/>
          <xs:enumeration value="HealthData"/>
          <xs:enumeration value="TransactionData"/>
          <xs:enumeration value="ServiceUseData"/>
         </xs:restriction>
        </xs:simpleType>
       </xs:element>
      </xs:sequence>
```

```
 </xs:complexType>
</xs:element>
<xs:element name="PII" minOccurs="0">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="SpecificInformation" minOccurs="0">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="PersonalInformationItem" maxOccurs="unbounded">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="ItemName" maxOccurs="unbounded">
          <xs:simpleType>
           <xs:restriction base="xs:string">
            <xs:enumeration value="IDNumber"/>
            <xs:enumeration value="UniqueIdentifierNumber"/>
            <xs:enumeration value="FullName"/>
            <xs:enumeration value="FirstName"/>
            <xs:enumeration value="Surname"/>
            <xs:enumeration value="MiddleName"/>
            <xs:enumeration value="NamePrefix"/>
            <xs:enumeration value="Birthdate"/>
            <xs:enumeration value="Gender"/>
            <xs:enumeration value="JobTitle"/>
            <xs:enumeration value="MailingAddress"/>
            <xs:enumeration value="HomeTelephoneNumber"/>
            <xs:enumeration value="TelephoneNumber"/>
            <xs:enumeration value="WorkTelephoneNumber"/>
            <xs:enumeration value="CellphoneNumber"/>
            <xs:enumeration value="FaxNumber"/>
            <xs:enumeration value="EmailAddress"/>
            <xs:enumeration value="URL"/>
            <xs:enumeration value="EmployersName"/>
            <xs:enumeration value="CompanyDepartment"/>
            <xs:enumeration value="CreditCardDetails"/>
            <xs:enumeration value="BankingDetails"/>
            <xs:enumeration value="FinacialInformation"/>
            <xs:enumeration value="ComputerInformation"/>
            <xs:enumeration value="DemographicData"/>
            <xs:enumeration value="PoliticalData"/>
            <xs:enumeration value="HealthData"/>
            <xs:enumeration value="TransactionData"/>
            <xs:enumeration value="ServiceUseData"/>
           </xs:restriction>
          </xs:simpleType>
         </xs:element>
         <xs:element ref="con:UsagePolicy"/>
         <xs:element ref="con:StoragePolicy"/>
         <xs:element ref="con:SharingPolicy"/>
        </xs:sequence>
        <xs:attribute name="mustProvide" type="xs:string" use="required"/>
       </xs:complexType>
      </xs:element>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
   <xs:element name="GeneralPolicy" minOccurs="0">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="UsagePolicy">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="UsagePolicyItem" maxOccurs="unbounded">
          <xs:complexType>
           <xs:sequence>
            <xs:element name="PolicyItemName">
             <xs:simpleType>
              <xs:restriction base="xs:string">
               <xs:enumeration value="CurrentTransactionOnly"/>
               <xs:enumeration value="NotUsed"/>
               <xs:enumeration value="InternalUseOnly"/>
               <xs:enumeration value="SiteAdministrationUse"/>
               <xs:enumeration value="InternalUse"/>
               <xs:enumeration value="InteralResearch"/>
               <xs:enumeration value="SiteCustomization"/>
               <xs:enumeration value="UserTracking"/>
               <xs:enumeration value="Marketing"/>
              </xs:restriction>
             </xs:simpleType>
            </xs:element>
            <xs:element name="PolicyItemDescription" type="xs:string"/>
            <xs:element name="PolicyItemValue" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
           </xs:sequence>
          </xs:complexType>
         </xs:element>
        </xs:sequence>
       </xs:complexType>
      </xs:element>
      <xs:element name="StoragePolicy">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="StoragePolicyItem" maxOccurs="unbounded">
          <xs:complexType>
           <xs:sequence>
```

```
                    <xs:element name="PolicyItemName">
                     <xs:simpleType>
                      <xs:restriction base="xs:string">
                       <xs:enumeration value="NoStorage"/>
                       <xs:enumeration value="CurrentTransactionOnly"/>
                       <xs:enumeration value="RequiredByLaw"/>
                       <xs:enumeration value="Indefinitely"/>
                       <xs:enumeration value="EncryptedStorage"/>
                      </xs:restriction>
                     </xs:simpleType>
                    </xs:element>
                    <xs:element name="PolicyItemDescription" type="xs:string" minOccurs="0"/>
                    <xs:element name="AcceptableValue" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
                    <xs:element name="Period" type="xs:string" minOccurs="0" >
                    <xs:element name="From" type="xs:date" minOccurs="0"/>
                    <xs:element name="To" type="xs:date" minOccurs="0" >
                    <xs:element name="OtherCompany" minOccurs="0" maxOccurs="unbounded">
                     <xs:complexType>
                      <xs:sequence>
                       <xs:element name="CompanyName" type="xs:string"/>
                       <xs:element name="CompanyType" type="xs:string"/>
                       <xs:element ref="con:ContactDetails" minOccurs="0"/>
                       <xs:element name="CompanyAffiliation" type="xs:string"/>
                      </xs:sequence>
                     </xs:complexType>
                    </xs:element>
                   </xs:sequence>
                  </xs:complexType>
                 </xs:element>
                </xs:sequence>
                <xs:attribute name="requiredByLaw" type="xs:string" use="required"/>
               </xs:complexType>
              </xs:element>
              <xs:element name="SharingPolicy">
               <xs:complexType>
                <xs:sequence>
                 <xs:element name="SharingPolicyItem" maxOccurs="unbounded">
                  <xs:complexType>
                   <xs:sequence>
                    <xs:element name="PolicyItemName">
                     <xs:simpleType>
                      <xs:restriction base="xs:string">
                       <xs:enumeration value="NoSharing"/>
                       <xs:enumeration value="SharingWithAgents"/>
                       <xs:enumeration value="DeliveryServices"/>
                       <xs:enumeration value="TransactionServices"/>
                       <xs:enumeration value="3rdParties"/>
                       <xs:enumeration value="Unrelated3rdParties"/>
                       <xs:enumeration value="GeneralPublic"/>
                      </xs:restriction>
                     </xs:simpleType>
                    </xs:element>
                    <xs:element name="PolicyItemDescription" type="xs:string" minOccurs="0"/>
                    <xs:element name="AcceptableValue" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
                    <xs:element name="OtherCompany" minOccurs="0" maxOccurs="unbounded">
                     <xs:complexType>
                      <xs:sequence>
                       <xs:element name="CompanyName" type="xs:string"/>
                       <xs:element name="CompanyType" type="xs:string"/>
                       <xs:element ref="con:ContactDetails" minOccurs="0"/>
                       <xs:element name="CompanyAffiliation" type="xs:string"/>
                      </xs:sequence>
                     </xs:complexType>
                    </xs:element>
                   </xs:sequence>
                  </xs:complexType>
                 </xs:element>
                </xs:sequence>
               </xs:complexType>
              </xs:element>
             </xs:sequence>
            </xs:complexType>
           </xs:element>
          </xs:sequence>
         </xs:complexType>
        </xs:element>
       </xs:sequence>
      </xs:complexType>
     </xs:element>
    </xs:sequence>
   </xs:complexType>
  </xs:element>
 </xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="UsagePolicy">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="UsagePolicyItem" maxOccurs="unbounded">
    <xs:complexType>
     <xs:sequence>
```

```
      <xs:element name="PolicyItemName">
       <xs:simpleType>
        <xs:restriction base="xs:string">
         <xs:enumeration value="CurrentTransactionOnly"/>
         <xs:enumeration value="NotUsed"/>
         <xs:enumeration value="InternalUseOnly"/>
         <xs:enumeration value="SiteAdministrationUse"/>
         <xs:enumeration value="InternalUse"/>
         <xs:enumeration value="InteralResearch"/>
         <xs:enumeration value="SiteCustomization"/>
         <xs:enumeration value="UserTracking"/>
         <xs:enumeration value="Marketing"/>
        </xs:restriction>
       </xs:simpleType>
      </xs:element>
      <xs:element name="PolicyItemDescription" type="xs:string"/>
      <xs:element name="PolicyItemValue" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="StoragePolicy">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="StoragePolicyItem" maxOccurs="unbounded">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="PolicyItemName">
       <xs:simpleType>
        <xs:restriction base="xs:string">
         <xs:enumeration value="NoStorage"/>
         <xs:enumeration value="CurrentTransactionOnly"/>
         <xs:enumeration value="RequiredByLaw"/>
         <xs:enumeration value="Indefinitely"/>
         <xs:enumeration value="EncryptedStorage"/>
         <xs:enumeration value="NormalStorage"/>
        </xs:restriction>
       </xs:simpleType>
      </xs:element>
      <xs:element name="PolicyItemDescription" type="xs:string" minOccurs="0"/>
      <xs:element name="AcceptableValue" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="Period" type="xs:string" minOccurs="0"/>
      <xs:element name="From" type="xs:date" minOccurs="0"/>
      <xs:element name="To" type="xs:date" minOccurs="0"/>
      <xs:element name="OtherCompany" minOccurs="0" maxOccurs="unbounded">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="CompanyName" type="xs:string"/>
         <xs:element name="CompanyType" type="xs:string"/>
         <xs:element ref="con:ContactDetails" minOccurs="0"/>
         <xs:element name="CompanyAffiliation" type="xs:string"/>
        </xs:sequence>
       </xs:complexType>
      </xs:element>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
  </xs:sequence>
  <xs:attribute name="requiredByLaw" type="xs:string" use="required"/>
 </xs:complexType>
</xs:element>
<xs:element name="SharingPolicy">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="SharingPolicyItem" maxOccurs="unbounded">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="PolicyItemName">
       <xs:simpleType>
        <xs:restriction base="xs:string">
         <xs:enumeration value="NoSharing"/>
         <xs:enumeration value="SharingWithAgents"/>
         <xs:enumeration value="DeliveryServices"/>
         <xs:enumeration value="TransactionServices"/>
         <xs:enumeration value="3rdParties"/>
         <xs:enumeration value="Unrelated3rdParties"/>
         <xs:enumeration value="GeneralPublic"/>
        </xs:restriction>
       </xs:simpleType>
      </xs:element>
      <xs:element name="PolicyItemDescription" type="xs:string" minOccurs="0"/>
      <xs:element name="AcceptableValue" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="OtherCompany" minOccurs="0" maxOccurs="unbounded">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="CompanyName" type="xs:string"/>
         <xs:element name="CompanyType" type="xs:string"/>
         <xs:element ref="con:ContactDetails" minOccurs="0"/>
         <xs:element name="CompanyAffiliation" type="xs:string"/>
        </xs:sequence>
       </xs:complexType>
      </xs:element>
```

```
      </xs:sequence>
     </xs:complexType>
    </xs:element>
   </xs:sequence>
  </xs:complexType>
 </xs:element>
</xs:schema>
```

7.3 PARTNER REQUEST SCHEMA DOCUMENT

This schema allows for the creation of partnerRequest and partnerRequestResponeList elements used in the trust assurance protocol.

```
<?xml version="1.0" encoding="UTF-8"?> <xs:schema
elementFormDefault="qualified" attributeFormDefault="unqualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:inf="http://dna.cs.uct.ac.za/TrustAssurance">
  <xs:import namespace="http://dna.cs.uct.ac.za/TrustAssurance" schemaLocation="C:\masters\masters\schema\policy.xsd"/>
  <xs:import namespace="http://dna.cs.uct.ac.za/TrustAssurance" schemaLocation="C:\masters\masters\schema\contact.xsd"/>
  <xs:element name="PartnerRequest">
    <xs:complexType>
      <xs:choice>
        <xs:element name="ServicePartners">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="ServicesRequestType">
                <xs:simpleType>
                  <xs:restriction base="xs:string">
                    <xs:enumeration value="RecommendedServices"/>
                    <xs:enumeration value="AllServices"/>
                  </xs:restriction>
                </xs:simpleType>
              </xs:element>
              <xs:element ref="Restrictions" minOccurs="0"/>
              <xs:element name="ReturnInformation" minOccurs="0">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element ref="inf:InformationItem"/>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="ServicePartnersMetaData">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="Restrictions" minOccurs="0">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element ref="InformationItemRestrictions"/>
                    <xs:element ref="ServiceRestrictions"/>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:choice>
      <xs:attribute name="depth" type="xs:int" use="optional"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="Restrictions">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="InformationItemRestrictions"/>
        <xs:element ref="ServiceRestrictions"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="ServicesGroup">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="ServicesGroupName">
          <xs:complexType>
            <xs:simpleContent>
              <xs:extension base="xs:string"/>
            </xs:simpleContent>
          </xs:complexType>
        </xs:element>
        <xs:element ref="ServiceInformation" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="groupNumber" type="xs:int" use="required"/>
    </xs:complexType>
  </xs:element>
```

```
<xs:element name="PartnerRequestResponse">
  <xs:complexType>
    <xs:choice>
      <xs:element name="ServicesList">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="ServicesGroup" maxOccurs="unbounded">
              <xs:complexType>
                <xs:sequence>
                  <xs:element name="ServicesGroupName">
                    <xs:complexType>
                      <xs:simpleContent>
                        <xs:extension base="xs:string"/>
                      </xs:simpleContent>
                    </xs:complexType>
                  </xs:element>
                  <xs:element ref="ServiceInformation" maxOccurs="unbounded"/>
                </xs:sequence>
                <xs:attribute name="groupNumber" type="xs:int" use="required"/>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="ErrorMessage" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="GroupNumber" type="xs:int"/>
            <xs:element name="ErrorMessage" type="xs:string"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element ref="inf:Policy"/>
    </xs:choice>
  </xs:complexType>
</xs:element>
<xs:element name="InformationItemRestrictions">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="InformationItemName" minOccurs="0" maxOccurs="unbounded">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="IDNumber"/>
            <xs:enumeration value="UniqueIdentfierNumber"/>
            <xs:enumeration value="FullName"/>
            <xs:enumeration value="FirstName"/>
            <xs:enumeration value="Surname"/>
            <xs:enumeration value="MiddleName"/>
            <xs:enumeration value="NamePrefix"/>
            <xs:enumeration value="Birthdate"/>
            <xs:enumeration value="Gender"/>
            <xs:enumeration value="JobTitle"/>
            <xs:enumeration value="MailingAddress"/>
            <xs:enumeration value="HomeTelephoneNumber"/>
            <xs:enumeration value="TelephoneNumber"/>
            <xs:enumeration value="WorkTelephoneNumber"/>
            <xs:enumeration value="CellphoneNumber"/>
            <xs:enumeration value="FaxNumber"/>
            <xs:enumeration value="EmailAddress"/>
            <xs:enumeration value="URL"/>
            <xs:enumeration value="EmployersName"/>
            <xs:enumeration value="CompanyDepartment"/>
            <xs:enumeration value="BankingDetails"/>
            <xs:enumeration value="FinacialInformation"/>
            <xs:enumeration value="ComputerInformation"/>
            <xs:enumeration value="DemographicData"/>
            <xs:enumeration value="PoliticalData"/>
            <xs:enumeration value="HealthData"/>
            <xs:enumeration value="TransactionData"/>
            <xs:enumeration value="ServiceUseData"/>
            <xs:enumeration value="CreditCardDetails"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:element>
      <xs:choice minOccurs="0" maxOccurs="unbounded">
        <xs:element ref="ExactlyOne"/>
        <xs:element ref="All"/>
        <xs:element ref="AllowAllExcept"/>
        <xs:element ref="DenyAllExcept"/>
        <xs:element ref="RestrictionCondition" maxOccurs="unbounded"/>
      </xs:choice>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="Assertions">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="AllowAllExcept"/>
      <xs:element name="DenyAllExcept"/>
      <xs:element name="ExactlyOne"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="ExactlyOne">
```

```
    <xs:complexType>
      <xs:sequence>
        <xs:choice minOccurs="0" maxOccurs="unbounded">
          <xs:element ref="ExactlyOne"/>
          <xs:element ref="All"/>
          <xs:element ref="AllowAllExcept"/>
          <xs:element ref="DenyAllExcept"/>
          <xs:element ref="RestrictionCondition" maxOccurs="unbounded"/>
        </xs:choice>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="All">
    <xs:complexType>
      <xs:sequence>
        <xs:choice minOccurs="0" maxOccurs="unbounded">
          <xs:element ref="ExactlyOne"/>
          <xs:element ref="All"/>
          <xs:element ref="AllowAllExcept"/>
          <xs:element ref="DenyAllExcept"/>
          <xs:element ref="RestrictionCondition" maxOccurs="unbounded"/>
        </xs:choice>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="DenyAllExcept">
    <xs:complexType>
      <xs:sequence>
        <xs:choice minOccurs="0" maxOccurs="unbounded">
          <xs:element ref="ExactlyOne"/>
          <xs:element ref="All"/>
          <xs:element ref="AllowAllExcept"/>
          <xs:element ref="DenyAllExcept"/>
          <xs:element ref="RestrictionCondition" maxOccurs="unbounded"/>
        </xs:choice>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="AllowAllExcept">
    <xs:complexType>
      <xs:sequence>
        <xs:choice minOccurs="0" maxOccurs="unbounded">
          <xs:element ref="ExactlyOne"/>
          <xs:element ref="All"/>
          <xs:element ref="AllowAllExcept"/>
          <xs:element ref="DenyAllExcept"/>
          <xs:element ref="RestrictionCondition" maxOccurs="unbounded"/>
        </xs:choice>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="RestrictionCondition">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="ItemName" minOccurs="0"/>
        <xs:element name="RestrictionName"/>
        <xs:element name="RequiredValue" type="xs:string"/>
        <xs:element name="ServiceValue" type="xs:string" minOccurs="0"/>
      </xs:sequence>
      <xs:attribute name="verifiable" type="xs:string" use="optional"/>
      <xs:attribute name="criticalFault" type="xs:string" use="optional"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="ServiceRestrictions">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="ServiceRestrictionCondition" type="xs:string" minOccurs="0" />
        <xs:choice minOccurs="0" maxOccurs="unbounded">
          <xs:element ref="ExactlyOne"/>
          <xs:element ref="All"/>
          <xs:element ref="AllowAllExcept"/>
          <xs:element ref="DenyAllExcept"/>
          <xs:element ref="RestrictionCondition" maxOccurs="unbounded"/>
        </xs:choice>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="ServiceInformation">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="inf:Policy"/>
        <xs:element ref="ServicesGroup"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="EvaluateRestrictions">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="PartnerRequest"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="EvaluateRestrictionsResponse">
    <xs:complexType>
```

```
    <xs:sequence>
      <xs:element name="Service">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="RestrictionMatchAssertion">
              <xs:complexType>
                <xs:sequence>
                  <xs:element name="SignedAssertion" minOccurs="0"/>
                  <xs:element name="AcknowledgedAssertion" minOccurs="0"/>
                </xs:sequence>
              </xs:complexType>
            </xs:element>
            <xs:element name="RestrictionMatchFailure">
              <xs:complexType>
                <xs:sequence>
                  <xs:element ref="RestrictionCondition" maxOccurs="unbounded"/>
                </xs:sequence>
              </xs:complexType>
            </xs:element>
            <xs:element ref="inf:Policy" minOccurs="0"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:complexType name="PartnersRequestType">
  <xs:choice>
    <xs:element name="DirectServices">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="ServicesRequestType">
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:enumeration value="RecommendedServices"/>
                <xs:enumeration value="AllServices"/>
              </xs:restriction>
            </xs:simpleType>
          </xs:element>
          <xs:element ref="Restrictions" minOccurs="0"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="AllServices">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="Restrictions">
            <xs:complexType>
              <xs:sequence>
                <xs:element ref="InformationItemRestrictions"/>
                <xs:element ref="ServiceRestrictions"/>
              </xs:sequence>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="ServicePartnersMetaData" type="xs:string"/>
  </xs:choice>
</xs:complexType>
<xs:element name="Services">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ServiceEquivalents" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
            <xs:element ref="Services"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="ServiceEquivalents">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="PartnersRequestResponseList" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="ServiceGroups">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="ServiceEquivalents" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="PartnersRequestResponseList">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ServiceInformation">
        <xs:complexType>
          <xs:sequence>
            <xs:sequence minOccurs="0">
```

```
            <xs:element name="ServiceName" type="xs:string"/>
            <xs:element name="ServiceURL" type="xs:string"/>
            <xs:sequence minOccurs="0">
              <xs:element ref="inf:InformationItem" maxOccurs="unbounded"/>
            </xs:sequence>
          </xs:sequence>
          <xs:element ref="inf:Policy" minOccurs="0"/>
          <xs:element name="ServiceFault" minOccurs="0">
            <xs:complexType>
              <xs:choice>
                <xs:element ref="RestrictionCondition" maxOccurs="unbounded"/>
                <xs:element name="NoFaults"/>
              </xs:choice>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element ref="ServiceGroups" minOccurs="0"/>
  </xs:sequence>
  <xs:attribute name="serviceName" type="xs:string" use="required"/>
  <xs:attribute name="serviceURL" type="xs:string" use="required"/>
  <xs:attribute name="FailedPRRL" type="xs:string" use="optional" default="no"/>
 </xs:complexType>
</xs:element>
<xs:element name="InformationItemDescription">
 <xs:complexType>
  <xs:choice>
   <xs:element ref="RestrictionCondition"/>
   <xs:element name="InfoElementInformation">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="ItemName" type="xs:string" minOccurs="0"/>
      <xs:element name="UsagePolicy" minOccurs="0">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="PolicyItem" minOccurs="0" maxOccurs="unbounded">
          <xs:complexType>
           <xs:sequence>
            <xs:element name="PolicyItemName"/>
            <xs:element name="PolicyItemDescription" type="xs:string" minOccurs="0"/>
            <xs:element name="AcceptableValue" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
           </xs:sequence>
          </xs:complexType>
         </xs:element>
        </xs:sequence>
       </xs:complexType>
      </xs:element>
      <xs:element name="StoragePolicy" minOccurs="0">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="PolicyItem" minOccurs="0" maxOccurs="unbounded">
          <xs:complexType>
           <xs:sequence>
            <xs:element name="PolicyItemName"/>
            <xs:element name="PolicyItemDescription" type="xs:string" minOccurs="0"/>
            <xs:element name="AcceptableValue" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
            <xs:element name="From" type="xs:date" minOccurs="0"/>
            <xs:element name="To" type="xs:date" minOccurs="0" />
            <xs:element name="OtherCompany" minOccurs="0" maxOccurs="unbounded">
             <xs:complexType>
              <xs:sequence>
               <xs:element name="CompanyName" type="xs:string"/>
               <xs:element name="CompanyType" type="xs:string"/>
               <xs:element ref="inf:ContactDetails" minOccurs="0"/>
               <xs:element name="CompanyAffiliation" type="xs:string"/>
              </xs:sequence>
             </xs:complexType>
            </xs:element>
           </xs:sequence>
          </xs:complexType>
         </xs:element>
        </xs:sequence>
       </xs:complexType>
      </xs:element>
      <xs:element name="SharingPolicy" minOccurs="0">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="PolicyItem" minOccurs="0" maxOccurs="unbounded">
          <xs:complexType>
           <xs:sequence>
            <xs:element name="PolicyItemName"/>
            <xs:element name="PolicyItemDescription" type="xs:string" minOccurs="0"/>
            <xs:element name="AcceptableValue" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
            <xs:element name="OtherCompany" minOccurs="0" maxOccurs="unbounded">
             <xs:complexType>
              <xs:sequence>
               <xs:element name="CompanyName" type="xs:string"/>
               <xs:element name="CompanyType" type="xs:string"/>
               <xs:element ref="inf:ContactDetails" minOccurs="0"/>
               <xs:element name="CompanyAffiliation" type="xs:string"/>
              </xs:sequence>
             </xs:complexType>
            </xs:element>
```

```
                    </xs:sequence>
                  </xs:complexType>
                </xs:element>
              </xs:sequence>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:choice>
 </xs:complexType>
</xs:element>
<xs:element name="VerificationRequest">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="ServiceName" type="xs:string"/>
   <xs:element ref="inf:ContactDetails" minOccurs="0"/>
   <xs:element name="VerificationElement" minOccurs="0" maxOccurs="unbounded">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="VerificationItemName" type="xs:string"/>
      <xs:element name="VerificationRequestDetails" type="xs:string"/>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="ServiceQuery">
 <xs:complexType>
  <xs:choice maxOccurs="unbounded">
   <xs:element ref="VerificationRequest"/>
   <xs:element ref="RatingRequest"/>
  </xs:choice>
 </xs:complexType>
</xs:element>
<xs:element name="RatingRequest">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="ServiceName" type="xs:string"/>
   <xs:element ref="inf:ContactDetails" minOccurs="0"/>
   <xs:element name="RatingElement" minOccurs="0" maxOccurs="unbounded">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="RatingItemName" type="xs:string"/>
      <xs:element name="RatingItemDetails" type="xs:string"/>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="ServiceQueryResponse">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="ServiceName" type="xs:string"/>
   <xs:element ref="inf:ContactDetails" minOccurs="0"/>
   <xs:element name="QueryResponseDetails">
    <xs:complexType>
     <xs:choice>
      <xs:element name="QueryResponseItem" maxOccurs="unbounded">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="QueryItemName" type="xs:string"/>
         <xs:element name="ServiceQueryDetails" type="xs:string"/>
         <xs:element name="ServiceQueryValue" type="xs:string"/>
         <xs:element name="AuthoritySignature" minOccurs="0"/>
        </xs:sequence>
       </xs:complexType>
      </xs:element>
      <xs:element name="FaultMessage">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="FaultItem" maxOccurs="unbounded">
          <xs:complexType>
           <xs:sequence>
            <xs:element name="QueryItemName" type="xs:string"/>
            <xs:element name="FaultDetails" type="xs:string"/>
            <xs:element name="FaultValue" type="xs:string" minOccurs="0"/>
            <xs:element name="AuthoritySignature" minOccurs="0"/>
           </xs:sequence>
          </xs:complexType>
         </xs:element>
        </xs:sequence>
       </xs:complexType>
      </xs:element>
     </xs:choice>
    </xs:complexType>
   </xs:element>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="RestrictionName">
 <xs:simpleType>
```

```
      <xs:restriction base="xs:string">
        <xs:enumeration value="Usage"/>
        <xs:enumeration value="Storage"/>
        <xs:enumeration value="Sharing"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:element>
</xs:schema>
```

7.4 CONTRACT SCHEMA

This schema document is used to create valid contracts between the client and the services involved in a transaction.

```
<?xml version="1.0" encoding="UTF-8"?> <xs:schema
elementFormDefault="qualified" attributeFormDefault="unqualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="PRRLSelection">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="ServiceInformation">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="ServiceName"/>
              <xs:element name="ServiceURL"/>
            </xs:sequence>
            <xs:attribute name="URL" type="xs:string" use="required"/>
          </xs:complexType>
        </xs:element>
        <xs:element name="Description" type="xs:string"/>
        <xs:element name="ExternalSevices" minOccurs="0">
          <xs:complexType>
            <xs:sequence>
              <xs:element ref="PRRLSelection" maxOccurs="unbounded"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="ValidPeriod" minOccurs="0">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="From" type="xs:date"/>
              <xs:element name="Till" type="xs:date" minOccurs="0"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="TransactionInformation" minOccurs="0">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="TransactionID" type="xs:string" minOccurs="0"/>
              <xs:element name="ClientID" type="xs:string" minOccurs="0"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="Signature">
          <xs:complexType>
            <xs:sequence minOccurs="0">
              <xs:element name="PublicCertificateURL" type="xs:string"/>
              <xs:element name="SignatureHash" type="xs:string"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="PRRLSelectionConfirmation">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="PRRLSelectionConfirmation" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element ref="PRRLSelection"/>
        <xs:element name="Signature">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="PublicCertificateURI" type="xs:string"/>
              <xs:element name="SignatureHash" type="xs:string"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```