A SERVICE–ORIENTED ARCHITECTURE-BASED FRAMEWORK FOR E-PROCUREMENT

BY

ADESINA ADERONKE ATINUKE
(B.sc. Management Information System)

A DISSERTATION SUBMITTED TO THE DEPARTMENT OF COMPUTER AND INFORMATION SCIENCES, COVENANT UNIVERSITY, OTA

IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE AWARD OF M.Sc DEGREE IN MANAGEMENT INFORMATION SYSTEM (MIS)

2009

CERTIFICATION

I hereby certify that this project, written by **ADESINA ADERONKE A.** was supervised by me and submitted to the department of computer and information sciences, college of science and technology, Covenant University, Ota, for the award of M.Sc. (MIS)

DR. C.K. AYO
Project Supervisor

Signature and Date

DR. E.F. ADEBIYI
Head of Department

Signature and Date

EXTERNAL SUPERVISOR

Signature and Date

DEDICATION

Mrs J.A. Adesina and Engr. & Mrs. Babajide Adedapo

ACKNOWLEDGEMENT

A big thank you to God Almighty for the grace, strength and wisdom bestowed unto me in the course of this programme. Daddy, you are indeed my very present help in time of need. Glory to your holy name.

From the depth of my heart I appreciate the fatherly love of my supervisor, Dr. C.K. Ayo. Thank you for your encouragement, guidance, and for believing in me. May the good God continue to bless you.

I appreciate entire staff of the Department of Computer and Information Sciences for their encouragement. I want to specially thank Mr J.O. Daramola my co-supervisor for his unquantifiable contribution to this project work. Thank you for your relentless effort, the push, inspiration and valuable ideas provided from the onset of this project till the very end. God will prosper you in all your endeavors in Jesus' name.

I remain thankful to the Management of Covenant University for availing me opportunity to be a partaker of the YATRAP initiative. May the Lord continue to strengthen and prosper the University.

To my entire family, thank you for your support, love, encouragement and prayers. I love you all, God bless you.

I love and appreciate all my colleagues in the department (Sis Sena, Aunty Itunu, Aunty Chi and Oyne) and at the YATRAP level, I deeply appreciate you all for your encouragement. God will prosper all of us in our endeavors in Jesus' name.

Once again I appreciate the God for His mighty hands at work in my life. Thank you Jesus.

TABLE OF CONTENT

LIST OF TABLES

LIST OF FIGURES

INTRODUCTION

1.1 BACKGROUND INFORMATION

Purchasing of goods and services represents the single largest expense in most organisations. Corporate procurement process is a shared responsibility among business unit managers, employees and business partners. It requires strategic, timely information in order to maximize efficiency. Procurement is an inter-enterprise process conducted to achieve exchange of documents in order to effect the purchase of goods and services. Electronic procurement is the contractual business relationship referred to as Business-to-Business (B2B) collaboration between the buying and the selling organisations. E-procurement is integral to the overall development of the procurement process and involves the use of an electronic system(s) to acquire goods, works and services from third parties or business partners as well as to provide e-business capability for business units [1]. Among the challenges faced by most E-procurement systems is the ability to achieve application-to-application interaction between business partners software.

Service Oriented Architecture (SOA) is an architectural paradigm and a discipline that may be used to build infrastructures enabling those with needs (consumers) and those with capabilities (providers) to interact via services across disparate domains of technology and ownership. SOA is a software architecture that starts with an interface definition and builds the entire application topology as a topology of interfaces, interface implementations and interface calls [2]. Web services are simply one set of technologies that can be used to implement it successfully. SOA is a relationship of services and service consumers, both software modules are large enough to represent a complete business function. Services are software modules that are accessed by name via an interface, typically in a request-reply mode. Service consumers are software that embeds a service interface proxy (the client representation of the interface).

Web Services provide a distributed computing technology for revealing the business services of applications on the Internet or intranet using standard XML protocols and formats. The use of standard XML protocols makes Web Services to be platform, language, and vendor independent, and also an ideal candidate for use in Enterprise Application Integration (EAI) solutions.

Web Services help to eliminate the interoperability issues of existing EAI solutions, such as CORBA and DCOM, by leveraging open Internet standards - Web Services Description Language (WSDL - to describe), Universal Description, Discovery and Integration (UDDI - to advertise), and Simple Object Access Protocol (SOAP - to communicate).

1.2 STATEMENT OF THE PROBLEM

Achieving effective B2B collaboration requires inter-enterprise communication among business partners. Prominent challenges organisations face in achieving B2B collaboration in their E-procurement model include:

- Lack of standard application-to-application interaction between business partners software.
- Lack of transparency and uniformity in the procurement model of big organisations
- Inefficient purchasing decision as a result of insufficient market information.
- An organisation controlling both ends of the information exchange. Such includes organisations with Electronic Data Interchange (EDI) as electronic medium of exchange.

1.3 AIM AND OBJECTIVES OF THE STUDY

The aim of this work is to evolve a model of interoperable SOA-based E-procurement framework for multi-tiered business organisations.

The objectives are:

- To identify the requirements for efficient B2B oriented E-procurement framework.
- To evolve an SOA-based framework for E-procurement.

- As a proof of concept to design and implement an E-procurement system prototype that is based on the proposed SOA framework.
- To evaluate the effectiveness and efficiency of E-procurement models

1.4 METHODOLOGY

This project will be implemented using SOA as the design principle for creating and using business processes packaged as services. The various business functions involved in the procurement process will be implemented as web services so as to promote software reusability, achieve platform independence and ubiquity. The Microsoft .Net framework is chosen as the programming implementation platform because of its rich graphical interface and the ease of creating and accessing web services. The implementation tools include:

- Extensible Markup Language (XML): for metadata.
- Simple Object Access Protocol (SOAP): serves as communication protocol.
- Web Service Description Language (WSDL): use to describe the services.
- Service Registry: service directory where business entities and the web services they offer are published.
- C#: is the programming language and SQL server is the database use for storing data and interacting with the web site.

1.5 SIGNIFICANCE OF THE STUDY

This work offers significant benefits to the e-business domain because of the following:

- It provides a flexible and cost efficient approach of carrying out business-to-business (B2B) e-business. It facilitates application-to-application communication.
- It helps organisations know a reliable means of pursuing their B2B collaboration. Exchange of message can be accomplished cheaper and faster.
- It provides ordered way of exchanging messages with business partners without having to control both ends of data exchange or an organisation being compelled to get their software work with major partners in that space.

- It offers fast accessibility to information and reduces the transaction cost.

1.6 CONTRIBUTION TO KNOWLEDGE

This study offers a demonstration of the use of SOA-based model for E-procurement that has platform independent interoperability in multi-tiered organisations.

1.7 LIMITATIONS OF THE STUDY

The scope is limited to the development of E-procurement model for Covenant University. Also the implementation of the prototype SOA-based system was based on the procurement function identified within Covenant University context.

1.8 OUTLINE

Chapter One presents the introduction, the statement of the problem, the aim and objective of the study, the significance of the study, the methodology used in the study, the limitations of the study and the organisation of the study. Chapter Two is a review of literature on E-procurement, SOA and web service applications.

Chapter Three presents the conceptual architecture of the E-procurement framework, its design and its development processes.

Chapter Four discusses the implementation experience as it relates to our case study of Covenant University. In Chapter Five, the discussion of result and conclusion are given.

CHAPTER TWO

LITERATURE REVIEW

2.1 E-PROCUREMENT DEFINITION

The Office of the Government Commerce (OGC), United Kingdom defined E-procurement as "The term used to describe the use of electronic methods in every stage of the purchasing process from identification of requirements through to payment, and potentially to contract management" [3].

E-procurement is integral to the overall development of procurement processes and involves the use of an electronic system to acquire goods, works and services from third parties. It covers all aspects of the purchasing cycle including:

• ***E-tendering:*** This is a web based system for sourcing tenders. E-tendering systems enable all stages of a tendering process to be completed over the Internet, from advertising a requirement to evaluating supplier responses and awarding a contract.

• ***E-purchasing:*** Is a complete 'end to end' purchasing system, integrated with back office systems, use in raising purchase orders, goods received, receipt and payment of invoices and management information and incorporating automated bank payment clearance.

• ***E-Marketplace:*** This is a web based system that enables 'punch out' to approve supplier catalogues.

• ***E-Auction:*** An e-auction is a web based system for conducting an auction of goods and services. An E-Auction will allow suppliers to compete for commodity contracts by out-bidding each other on price, during a 'live' auction, having already satisfied the qualitative and technical requirements of a contract.

2.2 REVIEW OF EXISTING E-PROCUREMENT SYSTEMS

2.2.1 CONVENTIONAL PROCUREMENT SYSTEM

Conventional procurement system begins with collection of data from material requirement planning of the enterprise. The purchasing department analyzes the inventory data and derives a strategic procurement plan for the purchasing of needed materials with a significant price variation. Request for quotations (RFQs) are then made based on the current market price of the materials [4].

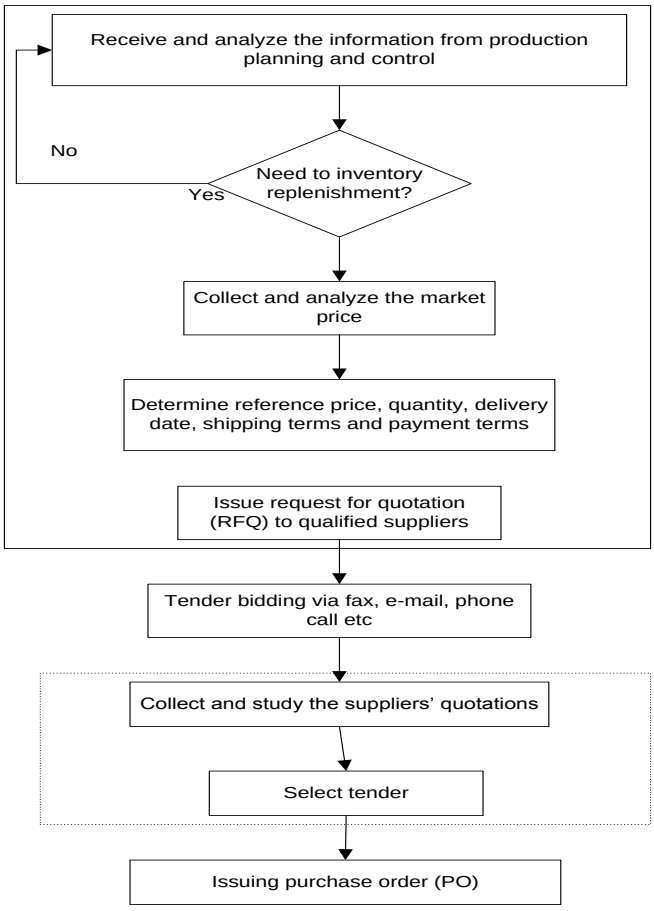

Figure 2.1: Conventional Procurement Approach [4]

2.2.2 AN AGENT-ORIENTED AND KNOWLEDGE-BASED SYSTEM FOR STRATEGIC E-PROCUREMENT (AOKBS) [4]

The AOKBS is designed based on a component-based architecture [4]. The architecture composed of six components: a knowledge repository (KR), autonomous agents, Request

For Quotation (RFQ) advisor, an online bidding module, a tender evaluation module and a supplier evaluation module. The KR is composed of a case library, database, business rules and dynamic data for supporting the other applications. The RFQ advisor and tender evaluation module accomplish the assimilation of the knowledge of procurement activities. The online bidding module enables the suppliers to tender anytime and anywhere via a standard browser.

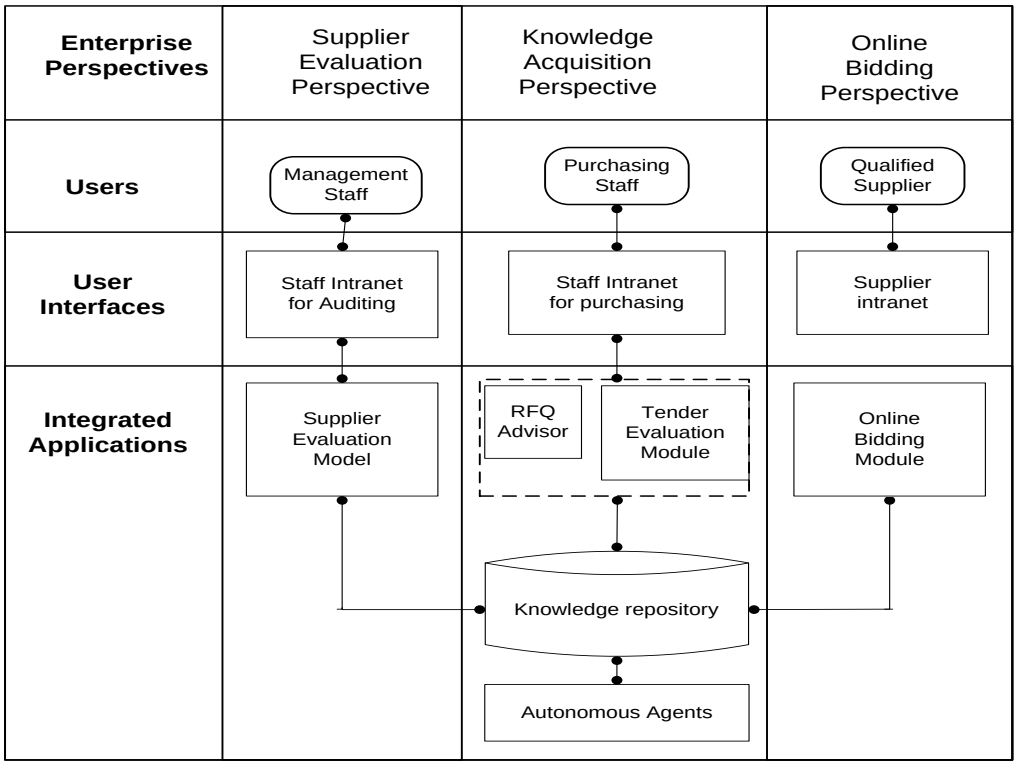

Figure 2.2: Architecture of AKOBS

2.2.3 GOVERNMENT E-PROCUREMENT SYSTEM (GePS) [5]

Samsung SDS National E-procurement system for Government E-procurement digitally processes complicated procedures and paperwork.

The GePS is the single portal for all public procurement in the Republic of Korea [5]. GePS was built to handle all public procurement online for all public organisations. GePS is a web based procurement system where all documents and procedures are processed electronically. Functionality of the system includes online bidding. The system architecture of GePS is divided into three main components. First the E-procurement

application service provider (ASP) handles E-procurement activities between public organisations and suppliers. Second, the connection with relevant organisations, the third is the product catalog information to supplier E-procurement.

GePS is linked to 53 external systems through the Internet. As an all-embracing system, it can share information for procurement with government organizations, relevant groups and associations. Through interface with six authorized certification agencies, certified electronic signatures can be used to secure safe online transaction. GepS is connected to 6 construction related organizations for sharing information on suppliers such as their management status and performance. A sum of 11 guarantee agencies process guarantee certificates regarding prepayment, bidding, contract signing and warranty. Via a connection to 17 major commercial banks means that GePS can process real-time electronic payments. The right hand-side and the bottom of figure 2.2 shows organisations which provide procurement information online and on the left hand-side are the public organisations and suppliers which are the principal users of GePS. Electronic documents for message exchange in GePS uses the World Wide Web Consortium recommend XML schema and the core components of ebXML. For message exchanges, GePS complies with international standards such as Simple Object Access Protocol (SOAP) and ebXML Message Service.

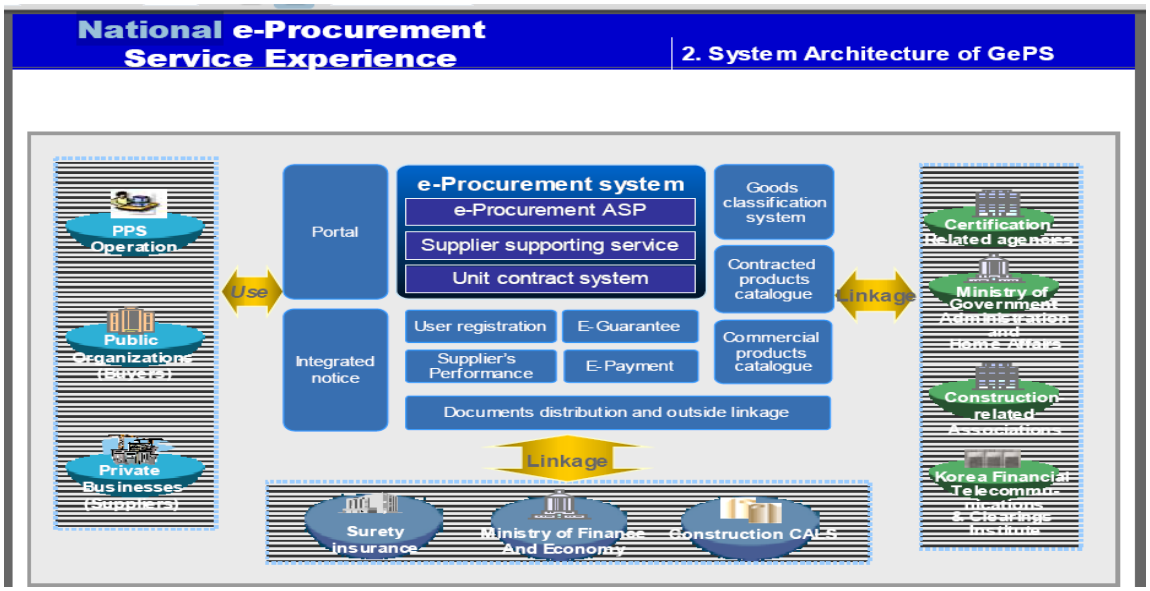

Figure 2.3: Samsung System for Government E-procurement [5]

The summary of the features GePS include:

- Digitizing all procedures from purchase requests to payment;
- Application of e-signature which is achieved through association with six authorized certification agencies;
- Sharing information including suppliers' data;
- Electronic handling of various contract-related bonds;
- Realization of real-time electronic payments; and
- Combination of information protection technology and a new E-commerce model.

2.2.4 DEPARTMENT FOR ADMINISTRATIVE AND INFORMATION SERVICES (DAIS) SOUTH AUSTRALIA E-PROCUREMENT SYSTEM INITIATIVE [6]

The Department for Administrative and Information Services (DAIS) South Australia has implemented an E-procurement system known as "E-Purchase SA". The functionalities provided by the E-Purchase SA are:

Requisition:

- Where a supplier has an established electronic catalogue that allows 'punch-out' capability, buyers will use E-Purchase SA to access the catalogue, identify goods they wish to purchase and these items are then brought back into the E-purchase system to populate a requisition.
- Where a supplier does not have an established catalogue, the buyer will use an electronic form and manually enter the details of the items to be purchased onto the form, which is then included onto the requisition.
- For regularly purchased items, buyers can copy previous requisitions or create templates to enable rapid generation of items on to requisitions.

Purchase orders: Once the requisition is completed, the system then uses automated workflow to route the order to the appropriate supervisors for approval (this is done

internally within the organisation). Once approved, the requisition is converted into a purchase order which is then automatically sent to the suppliers.

Purchase orders are sent to suppliers in several different ways again depending on organisation's capability:

• Where suppliers can receive orders in a complete electronic format (i.e. EDI Purchase Order) then orders are sent using message routing to deliver the orders to these suppliers.

• Where suppliers wish to receive orders via email, the supplier provides an appropriate email address and the system delivers orders as an attached word document to the email address.

• For suppliers with no electronic capability, the system automatically delivers orders to the supplier by facsimile.

Invoices: Where suppliers have the capability to return an electronic invoice, the suppliers can post an electronic invoice in a specified format to a location known by the E-procurement application. The E-Purchase SA system will then retrieve these invoices, match them against the original purchase order and then apply 3-way matching rules which match the purchase order and goods receipt to the invoice received. Where invoices match, they are automatically passed for payment. Exceptions or mismatches are flagged and addressed by appropriate staff to accept or reject the variations.

2.2.5 WEB SERVICE INTEROPERABILITY SUPPLY CHAIN MANAGEMENT SYSTEM [7]

The Web Services Interoperability Organization (WS-I) has developed a supply chain management business scenario to demonstrate features of the WS-I Basic Profile 1.0. This business scenario depicted by figure 2.4 shows how e-business can be conducted with SOA approach implemented using web services. This approach was used to develop solutions to real-world business requirements, that are based on interoperability principles defined in the WS-I Basic Profile. The business scenario for the supply chain management is such that the retailer system through web service requests fulfillment of a consumer's order from the internal company warehouse, which responds as to whether

line items from the order can be filled. If stock for any line item falls below a minimum threshold in the warehouse, a replenishment order is sent to an external manufacturer using the B2B model. The Retailer System contains the following web services:

• Retailer Service
• Warehouse Service
• Warehouse Callback Service

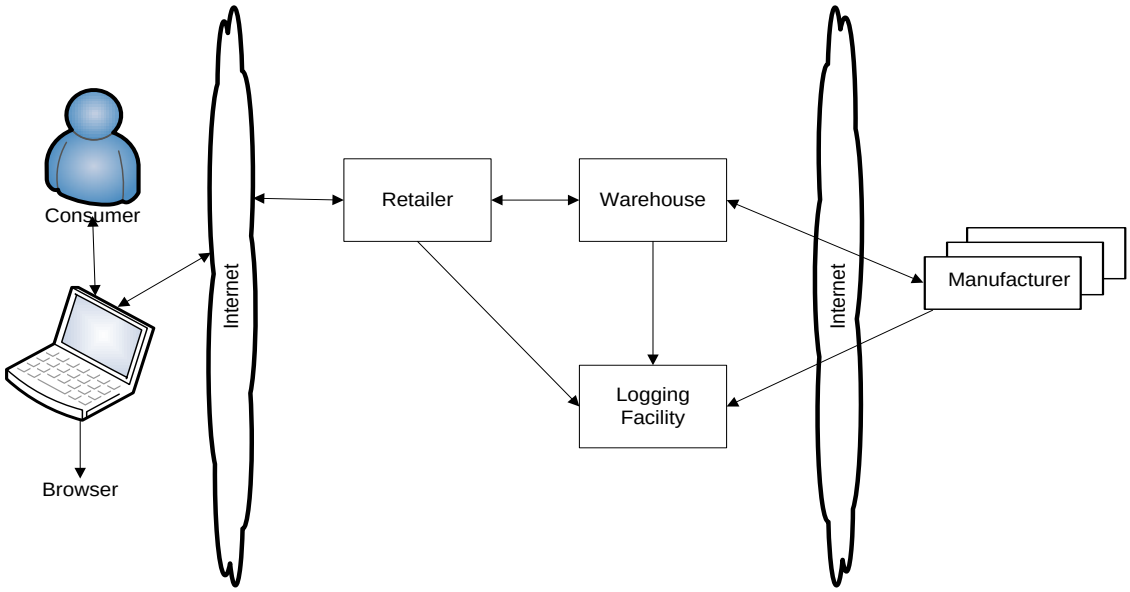

Figure 2.4: High level business context for WS-1 supply chain management [7]

2.2.6 BUSINESS-TO-BUSINESS INTERACTION

Web service architecture depicted in figure 2.5 gives an example of web service technology as a means to solving enterprise application integration (EAI) problem by enabling interaction among different applications within the same organization [10]. It enables computer-to-computer applications of different organizations to interact without human intervention. The business scenario on which it was modeled was to achieve ordering of goods using electronic communication. Company A (a customer) orders goods from Company B (a distributor). Company B checks the availability of the goods from Company C (a supplier), and then arranges for payment and shipping of the goods to Company A without human intervention.

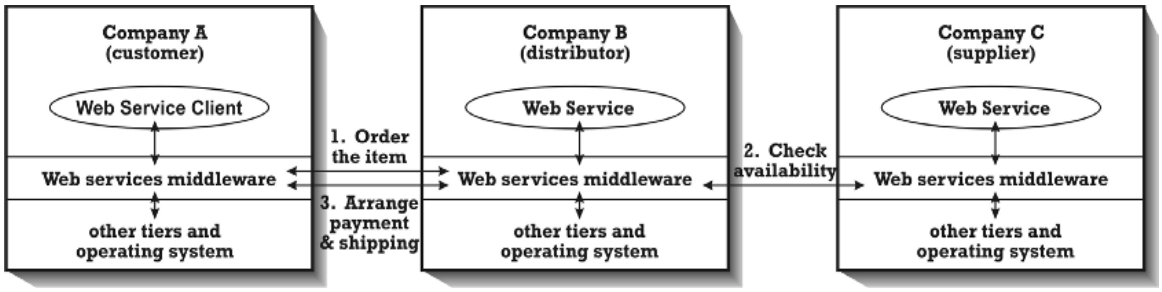

Figure 2.5: Use of Web services in Business-to-Business interactions [10]

2.2.7 B2B E-COMMERCE SYSTEM SPECIFICATION AND IMPLEMENTATION EMPLOYING USE-CASE DIAGRAMS, DIGITAL SIGNATURES AND XML

Sheldon et al. in [11], developed a remote order and delivery web-based B2B E-commerce system for an auto-parts manufacturing company. The main goal of the system was to improve the order and delivery process with remotely located businesses partners (contractors). The B2B E-commerce system developed using Use-Case diagrams and Scenarios for requirements analysis as well as, Digital Signatures and XML. The system was an object-oriented system. A formalized specification of the requirements employing Use-Case Diagrams was developed and coded using XML. Digital signatures were employed for implementing security. Effective communication between users and developers, processing time, process cost, reusability, efficiency, and security were considered as critical success factors for building a successful ecommerce system. Other factors such as ease-of-use, speed, accuracy, security and reliability; were all also considered as essential for the success of the B2B E-commerce system. DTDs (Document Type Definitions) and XML elements to denote the input and output of the service and values. Figure 2.6 is the context diagram of the B2B E-commerce system. The B2B system exchanges only server programs and encoded files while communicating with other business companies. The XML documentation is transmitted by using a socket on the application. The DTD and XSL are saved in a global repository and are used by the web browser's parser for validation based on the information in the XML documentation itself that is sent and received.

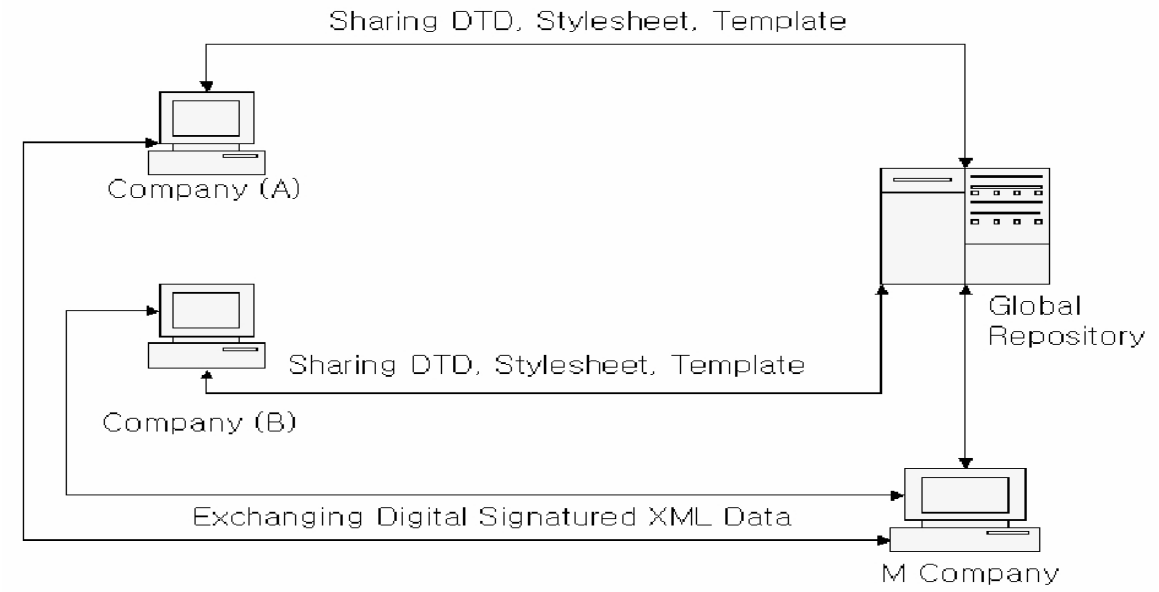

Figure 2.6: B2B E-commerce system context diagram [11]

2.3 INTEROPERABLE ELECTRONIC CATALOGUE AND ORGANISATION E-PROCUREMENT

Interoperability is the capacity to move data between different software applications. Through interoperability, supplier's catalogue can become a central reference for many other business processes within an organisation. An e-catalogue refers to a list of product and services that are for sale incorporating a price list. An online catalogue has the potential to become more than just a list of products and services available for purchase. For efficient electronic business-to-business transactions, e-catalogue structure of information needs to be "consistent" "complete" (product well described), "comprehensive", and "enriched" and integrated with the organisation's accounting system, purchasing system, reporting system and sales system [6].

Enriched catalogue content refers to additional information in the catalogue; such as image, ADOBE PDF files, word documents, and 3D files (multimedia files, Auto-CAD or MIME compatible file) with the aim of producing product contents for buyers to find, compare and buy in a more interactive, intuitive and easier way than paging through printed documents.

When catalogue is linked to buyers' purchasing system, it will become a living part of the purchasing and selling process. This gives the buyer an advantage to select product automatically or manually from the supplier's catalogue, include these in a purchase order and send it electronically to the supplier. Simultaneously on receipt of the order, the supplier would be able to upload the buyer's order(s) into their sales, inventory, and financial systems. In the complete purchase-to-purchase cycle, the supplier would then dispatch the goods to the buyer; send an electronic invoice to the buyer; who would be in the position to do a three-way data match (purchase order, goods receipt, invoice) and payment can be made as previously agreed by the two parties.

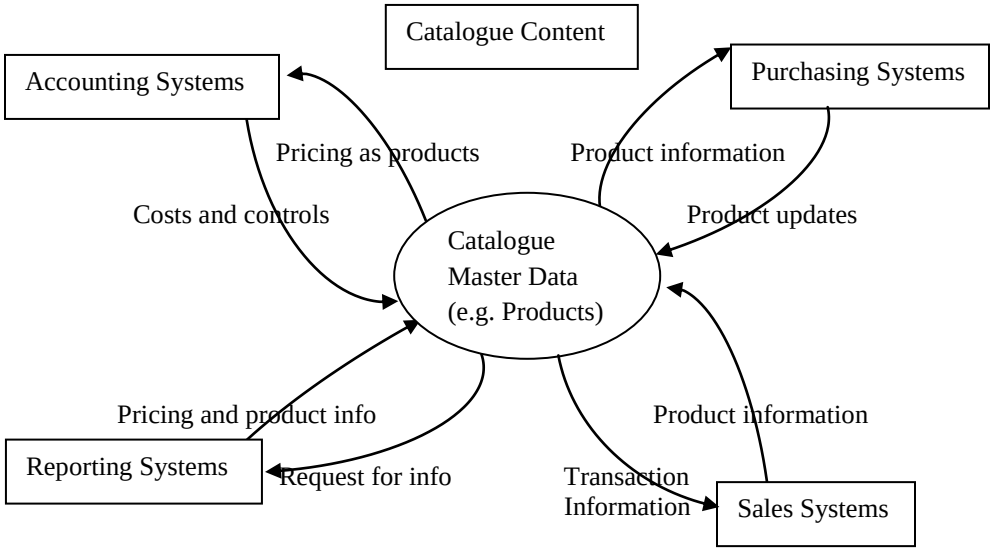

Figure 2.7: Relationship within e-catalogue and other organisation's business systems [5]

2.4 PARTIES INVOLVED IN PROCUREMENT PROCESSES

The Office of the Government Commerce (OGC) United Kingdom identified three bodies has parties involved in procurement processes [3].

1. **Buyer Side Parties**

- **Purchasing Manager:** Purchasing manager is a party in the buyer organisation responsible for the arrangement and maintenance of trading agreements with sellers.
- **Originator:** Originator is buyer party that identifies demand for goods or services and is subsequently the contact point for queries relating to the requirement, when referenced in subsequent documents.
- **Order Point:** Order point is the party that creates Purchase Order and any revision or cancellation to Purchase Order.
- **Delivery Point:** Delivery Point is the buyer Party that receives goods and services and identifies variances in receipt. Delivery Point is required for each Purchase Order and Invoice Line.
- **Invoice To:** Invoice To is the buyer Party responsible for payment-related issues.

2. **Supply Side Parties**

- **Sales Point:** Sales Point is the principal seller party responsible for purchasing issues prior to fulfillment of Purchase Order.
- **Customer Service:** Customer Service is the seller party responsible for Purchase Order fulfillment issues prior to invoicing, including problems relating to dispatch, delivery and carriage.
- **Dispatch Point:** Dispatch Point is the seller's party responsible for the dispatch, delivery and carriage.
- **Accounts Receivable:** Accounts Receivable is the seller's party for payment-related issues.

3. **Third Parties**

- **EFT Address:** Electronic Funds Transfer (EFT) Address contains the information needed about the bank and the bank account primarily to transfer funds using BACS (Bankers Automated Clearing System) or SWIFT (Society of Worldwide Interbank Financial Telecommunications).

- **Carrier:** Carrier is a third party assigned by either the seller or the buyer to deliver or return goods.
- **Factor:** Factor is a third party assigned by the seller to collect payment on its behalf.

2.5 POTENTIAL BENEFITS OF E-PROCUREMENT

E-procurement is an inter-organisation information system which can enable firms to make significant improvements in efficiency of their procurement processes. Through E-procurement, organisations can achieve significant cost savings as a result of electronic document preparation and better information exchange. The potential benefits can be numerous; the most important are summarized as follows [1, 8]:

- Improved control of vendor relationships.
- Purchasing information reaches larger audience.
- Competition among vendors and driving down prices.
- Accurate fulfillment of the processes.
- Improved effectiveness of the purchasing process.
- Achievement of higher service levels.
- Reduced prices from the key suppliers.
- Reduced inventory carrying cost.
- Reduction of the order cycle.

2.6 INTRODUCTION TO SERVICE-ORIENTED ARCHITECTURE (SOA)

Service Oriented Architecture is an architectural paradigm and discipline that may be used to build infrastructures enabling those with needs (consumers) and those with capabilities (providers) to interact via services across disparate domains of technology and ownership. SOA is an approach to designing integration architecture based on concept of service. It is a software architecture that starts with an interface definition and builds the entire application topology as a topology of interfaces, interface implementations and interface calls. Service-oriented architecture presents an approach

for building distributed systems that deliver application functionality as services to either end-user applications or other services [8]. SOA is a relationship of services and service consumers, both software modules large enough to represent a complete business function.

Services are software modules that are accessed by name via an interface, typically in a request-reply mode [8,9]. A service is a unit of work such as a business function, a business transaction, or a system service completed by a service provider to achieve desired end results for a service consumer. Service consumers are software that embeds a service interface proxy (the client representation of the interface). The SOA concept separates the service's implementation from its interface. Service consumers view a service simply as an endpoint that supports a particular request format or contract [8, 12]. Service consumers are not concerned with how the service goes about executing their requests; they expect only that it will undoubtedly produce an answer.

Consumers also expect that their interaction with the service will follow a contract, an agreed-upon interaction between two parties. The way the service executes tasks given to it by service consumers is irrelevant. The service might fulfill the request by executing a servlet, a mainframe application, a C# or a Visual Basic application. The only requirement is that the service sends the response back to the consumer in the agreed-upon format.

2.6.1 SOA and E-Business

SOA provides the emerging trend for organisations' transformation program. This is to make information resources substiantially independent, reusable, and to create an adaptable environment. Business and technical services are published using open standard protocols that create self-describing services that can be used independently of underlying technology [9]. Technical independence allows sevices to be more easily used in different contexts to achieve standardisation of business processes, rules and polices. Collaboration, internal and external to an enterprise, can more easily be established through improvement in process and information consistency [9]. Most significant SOAs

are proprietary or customized implementations based on reliable messaging and Enterprise Application Integration middleware (for example WebSphere Business Integration). SOA is an approach for building distributed system that delivers application functionality as services to either end-user applications or other services [8].

2.6.2 COLLABORATION BETWEEN SOA ENTITIES

SOA consist of three entities: Service provider, Service Consumer and Service Registry [8, 9, 11]. The collaboration between these three entities follows the "find, bind, and execute" paradigm as shown in Figure 2.8, allows the consumer of a service to ask a third-party registry for the service which it is intending to bind to. If the registry has such a service, it gives the consumer a contract and an endpoint address for the service.

Service Consumer: The service consumer is an application, service, or some other type of software module that requires a service. It is the entity that initiates the location of the service in the registry, binding to the service over a transport, and executing the service function. The service consumer executes the service by sending it a request formatted according to the contract.

Service Provider: The service provider is a network-addressable entity that accepts and executes requests from consumers. It can be a mainframe system, a component, or some other types of software system that executes the service request. The service provider publishes its contract in the registry for access by service consumers.

Service Registry: A service registry is a network-based directory that contains available services. It is an entity that accepts and stores contracts from service providers and provides those contracts to interested service consumers.

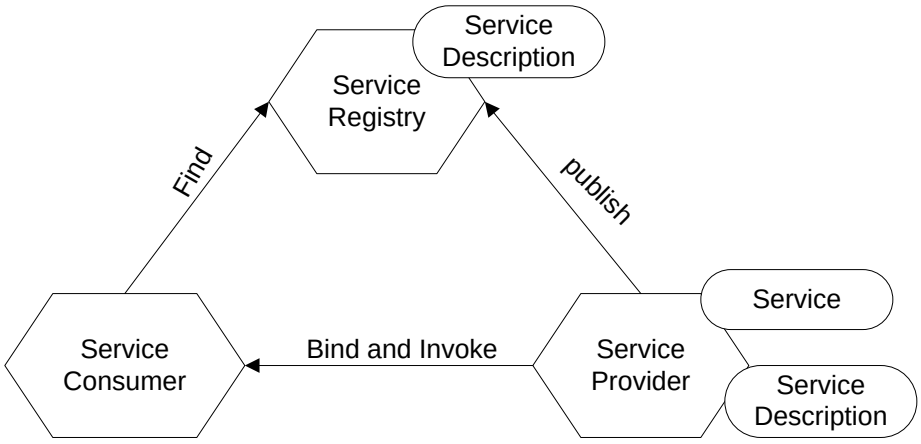

Figure 2.8: Collaboration in Service Oriented Architecture

The operations in a SOA are:

- **Publish:** for a service to be accessible, the service provider publishes the service description so that it can be discovered and invoked by service consumers.
- **Find:** service requester locates a service by querying the service registry for a service that meets its criteria.
- **Bind and invoke:** after successfully retrieval the service description, the service consumer then invokes the service based on the information provided by the service description.

Service Contract: A contract specifies the way a consumer of a service will interact with the provider of the service. It specifies the format of the request and response from the service. A service contract may require a set of preconditions and post-conditions. The preconditions and post-conditions specify the state that the service must be in to execute a particular function. The contract may also specify quality of service (QoS) levels. QoS levels are specifications for the non-functional aspects of the service [8]. For instance, a quality of service attribute is the amount of time it takes to execute a service method.

2.6.3 SERVICE PROVIDER AND SERVICE CONSUMER RELATIONSHIP

1. Negotiated: both consumer and provider jointly agree to how the services should work. In scenarios where there are many participants and where services are common to many providers, it is important that the industry considers standardizing those services [9,11]. This include:

- Close partners agreeing on the service interface as a natural part of reaching and implementing a commercial agreement
- Forming standard for vertical sectors in the industry.

2. Mandated: this is a take-it or leave-it scenario [12]. Very large or dominant organisation(s) dictate the business practice in their industry. Examples include:
    - Provider-led situations – such as Ford Motors "use this service or we can't do business".
    - Consumer-led situations – such as Walmart and Tesco [9].

2.6.4 CHARACTERISTICS OF SOA

The concept of services in software engineering has been in existence long before the advent of service-oriented architecture. However, service-oriented software architecture like every other software architecture reflects principles that make it suitable for implementing distributed functionalities as service. The following are SOA characteristics:

1. **Discoverable and Dynamically Bound**

   SOA supports the concept of service discovery. A service consumer that needs a service discovers what service to use based on a set of criteria at runtime. The service consumer asks a registry for a service that fulfills its needs.

2. **Self-Contained and Modular**

   Services are self-contained and modular. A service supports a set of interfaces. These interfaces should be cohesive, meaning that they should all relate to each other in the context of a module. The principles of modularity should be adhered to in designing the services that support an application so that services can easily be aggregated into an application with a few well-known dependencies.

   Modular Decomposability: The *modular decomposability* of a service refers to the breaking of an application into many smaller modules where each module is performing distinct function within an application. This is sometimes referred to as "top-down design," in which the bigger problems are iteratively decomposed

into smaller problems. The crust of this is to achieve reusability. The goal for service design is to identify the smallest unit of software that can be reused in different contexts [11].

Modular Composability: The *modular composability* of a service refers to the production of software services that may be freely combined as a whole with other services to produce new systems. Service designers should create services sufficiently independent to reuse in entirely different applications from the ones for which they were originally intended. This is sometimes referred to as bottom-up design [9,11].

Modular Understandability: The *modular understandability* of a service is the ability of a person to understand the function of the service without having any knowledge of other services.

Modular Continuity: The *modular continuity* of a service refers to the impact of a change in one service requiring a change in other services or in the consumers of the service. This is as a result of an interface not sufficiently hiding its implementation details. It will require changes to other services and applications that use the service when the internal implementation of the service changes. Every service must hide information about its internal design. A service that exposes this information will limit its modular continuity, by exposing internal design decision through the interface [11,12].

Modular Protection: The *modular protection* of a service is sufficient if an abnormal condition in the service does not cascade to other services or consumers. Faults in the operation of a service must not impact the operation of a client or other service or the state of their internal data or otherwise break the contract with service consumers. Therefore, we must ensure that faults do not cascade from the service to other services or consumers.

Direct Mapping: A service should map to a distinct problem domain function. This is to allow service designers create a self-contained and independent module [11].

Conceptual Service Model: The conceptual service model consists of a model of the problem domain. Techniques for defining module interfaces assume that the problem domain is known a priori. The conceptual model of the business, i.e. the *business architecture.* A conceptual model is one created without regard for any application or technology. It typically consists of a structural model derived from a set of use cases that illustrate how the business works.

Contracts and Information Hiding: An interface contract is a published agreement between a service provider and a service consumer. The contract specifies the arguments the service requires to be invoked, the return values a service supplies and the service's pre-conditions and post-conditions. The pre-conditions are those that must be satisfied before calling the service, to allow the service to function properly.

3. **Interoperability:** Service-oriented architecture stresses interoperability: the ability of systems using different platforms and languages to communicate with each other. Each service provides an interface that can be invoked through a connector type. An interoperable connector consists of a protocol and a data format that each of the *potential* clients of the service understands. Interoperability is achieved by supporting the protocol and data formats of the service's current and potential clients.

4. **Loose Coupling:** *Coupling* refers to the number of dependencies between modules. There are two types of coupling: loose and tight. Loosely coupled modules have a few well known dependencies. A system's degree of coupling directly affects its modifiability. The more tightly-coupled a system is, the more a change in a service will require changes in service consumers. Coupling is increased when service consumers require a large amount of information about the service provider to use the service. In other words, if a service consumer

knows the location and detailed data format for a service provider, the consumer and provider are more tightly coupled. If the consumer of the service does not need detailed knowledge of the service before invoking it, the consumer and provider are more loosely coupled. SOA accomplishes loose coupling through the use of contracts and bindings. A consumer asks a third-party registry for information about the type of service it wishes to use. The registry returns all the services it has available that match the consumer's criteria. The consumer chooses which service to use, binds to it over a transport, and executes the method on it, based on the description of the service provided by the registry. The consumer does not depend directly on the service's implementation but only on the contract the service supports [8,10,11].

Although coupling between service consumers and service producers is loose, implementation of the service can be tightly coupled with implementation of other services. For instance, if a set of services shares a framework, a database, or otherwise has information about each other's implementation, they may be tightly coupled.

5. **Network-Addressable Interface:** A service must have a network-addressable interface. A consumer on a network must be able to invoke a service across the network. The network allows services to be reused by any consumer at any time. The ability for an application to assemble a set of reusable services on different machines is possible only if the services support a network interface. The network also allows the service to be location–independent, meaning that its physical location is irrelevant.

6. **Coarse-Grained Interfaces:** The concept of granularity applies to the scope of the domain the entire service implements and also the scope of the domain that each method with the interface implements. If a service implements all the functions in its domain, it is referred to as coarse grained, but if it implements just a function in its domain, we consider it as fine grained. The appropriate level of granularity for a service and its methods is relatively coarse. A service generally

supports a single distinct business concept or process. It contains software that implements the business concept so that it can be reused in multiple large, distributed systems.

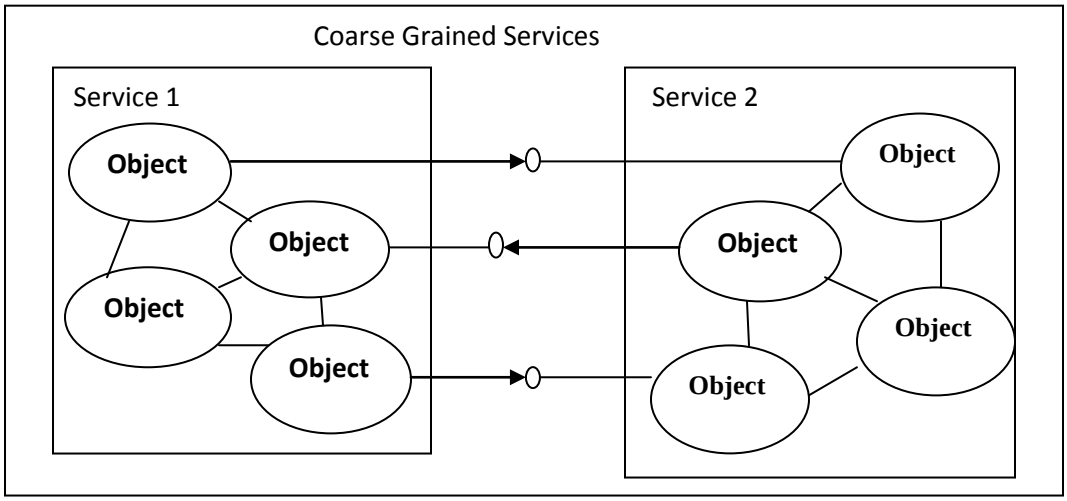

Figure 2.9: Coarse Grained Services

7. **Location Transparency:** Consumers of a service do not know a service's location until they locate it in the registry. The lookup and dynamic binding to a service at runtime allows the service implementation to move from location to location without the client's knowledge. The ability to move services improves service availability and performance.

8. **Composability:** A service may be composed in three ways: application composition, service federations, and service orchestration.

   An application composition is essentially an assembly of services, components, and application logic that binds these functions together for a specific purpose.

   Service federations are collections of services managed together in a larger service domain. Service orchestration is the execution of a single transaction that impacts one or more services in an organization. It is sometimes called a business process. It consists of multiple steps, each of which is a service invocation. If any of the service invocations fails, the entire transaction should be rolled back to the state that existed before execution of the transaction.

For a service to be composed into a transactional application, federation, or orchestration, the service methods themselves should be *sub-transactional*. That is, they must not perform data commits themselves. The orchestration of the transaction is performed by a third-party entity that manages all the steps. It detects when a service method fails and asks all the services that have already executed to roll-back to the state they existed before the request. If the services have already committed the state of their data, it is more difficult for the method to be composed into a larger transactional context.

9. **Self-Healing:** A *self-healing* system is one that has the ability to recover from errors without human intervention during execution [8]. *Reliability* measures how well a system performs in the presence of disturbances. Reliability depends on the hardware's ability to recover from failure [14]. The network must also allow for the dynamic connection to different systems at runtime.
   Service-based systems require that the interface be separate from the implementation, implementations may vary. For instance, a service implementation may run in a clustered environment. If a single service implementation fails, another instance can complete the transaction for the client without the client's knowledge. This capability is possible only if the client interacts with the services interface and not its implementation.

2.6.5 SOA IMPLEMENTATION MODELS

SOA is an architectural style that presents an approach for building distributed systems that deliver application functionality as services to either end-user applications or other services. Early adopters of the service-oriented architecture approach used messaging systems to create service-oriented enterprise architecture. Examples of these include IBM WebSphere MQ. Currently, the SOA arena has expanded to include the World Wide Web (WWW), Web Services and Enterprise Service Bus (ESB).

ESB: is an architectural practice for implementing a service-oriented architecture. As shown in Figure 2.10, it establishes an enterprise-class messaging bus that combines

messaging infrastructure with message transformation and content-based routing in a layer of integration logic between service consumers and providers [7].

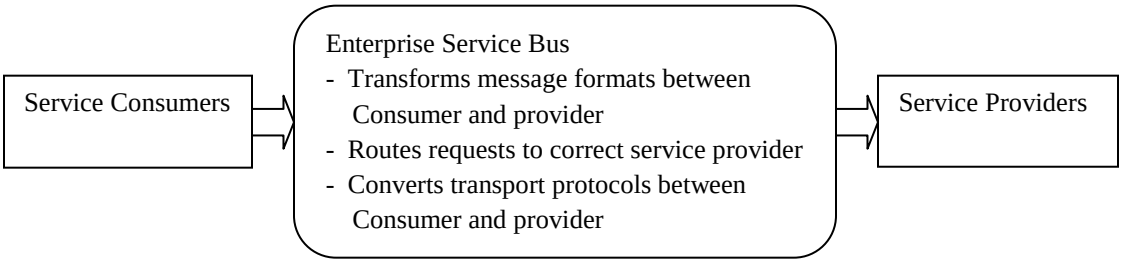

Figure 2.10: Enterprise Service Bus [7]

The ESB incorporates a standards-based, enterprise-class messaging backbone, together with enhanced systems connectivity using Web services, Java 2 Enterprise Edition (J2EE), Microsoft .NET, and other standards. In essence, ESB makes large-scale implementation SOA principles manageable in the heterogeneous world [15].

The ESB helps to provide virtualization of the enterprise resources, by allowing the business logic of the enterprise to be developed and managed independently of the infrastructure, network, and provision of those business services. Using ESB, one can link individual enterprises together for extended process efficiency across the supply chain and allow them to become more flexible and adaptable to rapidly changing requirements. The ESB lets an enterprise leverage its previous investments by supporting the deployment of processes over existing software and hardware infrastructure. ESB supported standards include:

- Java Message Service (JMS) for communication;
- Web services, J2EE, and .NET for connectivity to various systems;
- Extensible Stylesheet Language Transformation (XSLT) and Xquery for transformation; and
- Lightweight Directory Access Protocol (LDAP), Secure Sockets Layer (SSL), and others for security.

Implementing an Enterprise Service Bus requires an integrated set of middleware services that support the following architectural styles [7]:

- *Services oriented architecture:* where distributed applications are composed of granular re-usable services with well-defined, published and standards-compliant interfaces.
- *Message-driven architectures:* where applications send messages through the ESB to receiving applications
- *Event-driven architectures:* where applications generate and consume messages independently of one another.

Other technologies that are at partly service-oriented and have been widely used in achieving interoperability include: Common Object Request Broker Architecture (CORBA), Remote Method Invocation (RMI), and Distributed Component Object Model (DCOM).

CORBA: an open-standards-based solution to distributed computing developed by Object Management Group (OMG). The purpose of developing CORBA was to define a component based software interoperability and communication standard that would model the real world through a representation of objects. These objects are the encapsulation of attributes, relationships and methods of software identifiable program components. The primary advantage of CORBA is that clients and servers can be written in any programming language because the objects are defined with a high level of abstraction provided by the interface definition language (IDL). Among the deficiencies of CORBA is that is does not take the Internet into consideration, it was developed as a standard, not a specification and therefore, a "CORBA standard" from one organisation may not work with that of another organisation [21, 23].

DCOM: This is a Microsoft approach to solving distributed computing problem. It is an object-oriented middleware technology that allows clients and server in a distributed environment to communicate with one another. The DCOM server publishes its methods to the clients by supporting multiple interfaces [9]. The disadvantage of using DCOM is that it is a proprietary tool and not cross-platform independent.

RMI: RMI enables one to create Java-to-Java applications, in which the methods of remote Java objects can be invoked from other Java virtual machines. A Java program can make a call on the remote object once it obtains a reference to the remote object. The disadvantage of RMI is that it is Java language-oriented.

2.7 WEB SERVICE

A Web service is a software system designed to support interoperable machine-to-machine interaction over a network [14]. According to W3C, "A Web service is a software application identified by a URI, whose interfaces and bindings are capable of being defined, described, and discovered as XML artifacts [10]. A Web service supports direct interactions with other software agents using XML-based messages exchanged via Internet-based protocols" [8,9]. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards [8].

Web service is a useful tool in enabling collaboration and sharing of business process between two or more enterprises. It offers technology neutrality and standard approach than using proprietary integration technologies [9]. Web services promises to offer enterprise application the capability that World Wide Web did to interactive end-user application. Primarily, web service is a technique that allows disparate server systems to communicate with each other and exchange information for which the web and traditional web browser is the primary data access point [9]. Web service is a good beginning toward implementing service-oriented architecture because its concept supports many of the characteristics of service-oriented architecture.

2.7.1 WEB SERVICE ARCHITECTURE

Web services architecture describes the relationship among various components and technology that comprises web services "stack". A valid implementation must consist of at least the components in the basic architecture. The basic architecture includes web services technology that allows:

- Exchange messages;
- Describing web services; and
- Publishing and discovering Web service descriptions as depicted in figure 2.7 [16].

The basic web services architecture defines an interaction between software agents as an exchange of messages between service requesters and service providers. Requesters are software agents that request the execution of a service. Providers are software agents that provide a service. Agents can be both service requesters and providers. Providers are responsible for publishing a description of the service(s) they provide. Requesters must be able to find the description(s) of the service(s).

The basic web service architecture models the interactions between three roles: the service provider, service discovery agency, and service requester. The interactions involve *publish*, *find*, and *bind* operations. In a typical scenario, a service provider hosts a network accessible software module (an implementation of a web service). The service provider defines a service description for the web service and publishes it to a requester or service discovery agency. The service requester uses a find operation to retrieve the service description locally or from the discovery agency (i.e. a registry or repository) and uses the service description to bind with the service provider and invoke or interact with the web service implementation. Service provider and service requester roles are logical constructs and a service may exhibit characteristics of both.

A software agent in the web services architecture can act in one or multiple roles, acting as requester or provider only, as both requester and provider, or as requester, provider, and discovery agency. A service is invoked after the description is found, since the service description is required to establish a binding.

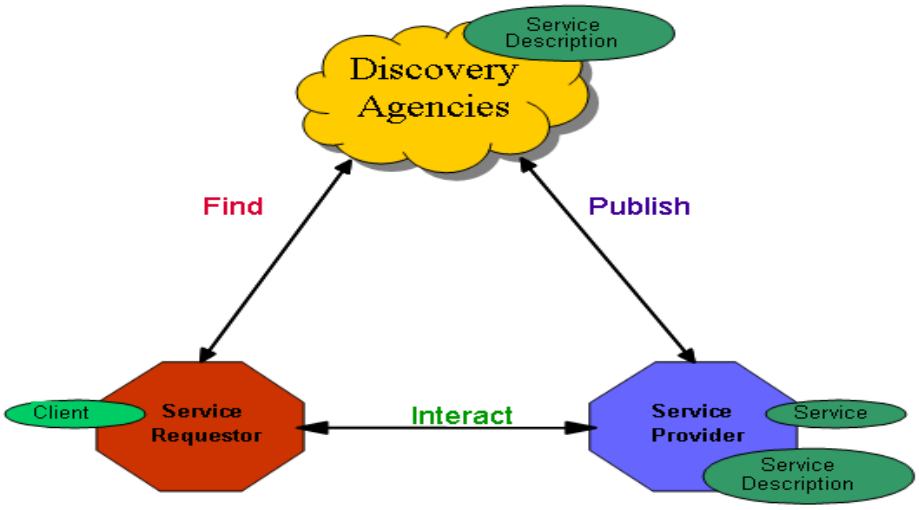

Figure 2.11: Web service architecture [17]

Figure 2.7 illustrates the basic Web services architecture, in which a service requester and service provider interact, based on the service's description information published by the provider and discovered by the requester through some form of discovery agency.

2.7.2 WEB SERVICE TECHNOLOGY

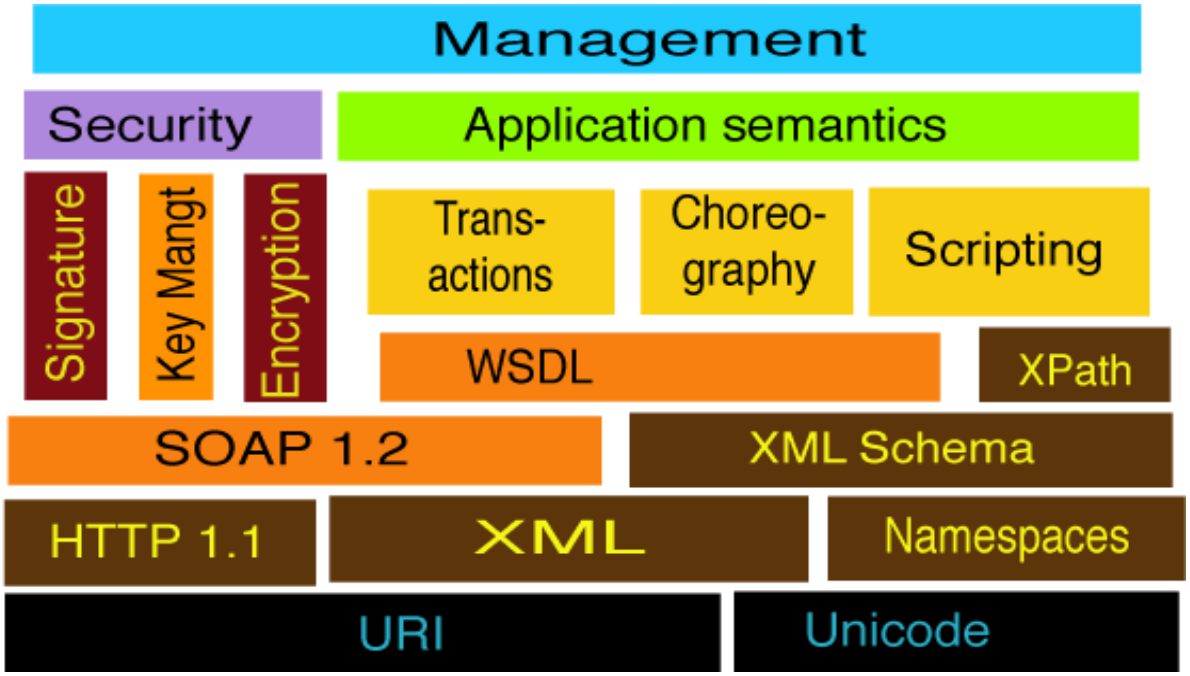

Figure: 2.12: Technology within the web service technology [18].

The management layer: is a supervisory layer allowing the control of the many agents involved in a web services-based operation.

The Application semantics: layer indicates the necessity for any useful interoperability.
Web Services Description Language (WSDL): A WSDL is an XML format to describe how a particular web service can be called. WSDL description specifies how to interact with the web service, what data must be sent, what operations are involved, what protocol is to be used to invoke the service, and what data can be expected in return.

A WSDL document uses the following elements in the definition of network services [7, 9]:

- **Types**– a container for data type definitions using some type system (such as XSD).
- **Message**– an abstract, typed definition of the data being communicated.
- **Operation**– an abstract description of an action supported by the service.
- **Port Type**–an abstract set of operations supported by one or more endpoints.
- **Binding**– a concrete protocol and data format specification for a particular port type.
- **Port**– a single endpoint defined as a combination of a binding and a network address.
- **Service**– a collection of related endpoints.

Simple Object Access Protocol (SOAP)

SOAP is an open Internet standard designed with three goals in mind [9]:

- It should be optimized to run on the Internet
- It should be simple and easy to implement
- It should be based on XML

Soap supports two types of message patterns: the first is the one-way exchange, where a client issues a request against a server, and will not receive an answer. The second is the pattern which consists of request response interaction. Here, the client use HTTP request for a resource on a server, and the server replies by sending a HTTP response.

Universal Description, Discovery, and Integration (UDDI)

The UDDI specification as prescribed by OASIS. UDDI 2004 provides a framework for describing and discovering web services. UDDI supports application developers in finding information about web services so that they know how to write clients applications that can interact with those services. It also enables dynamic binding by allowing clients to query the registry and obtain references to services in which they are interested. The information within a UDDI registry can be categorized as follows [9]:

- Listings of organizations, contact information, and services that those organizations provide;
- Classifications of companies and web services according to taxonomies that are either standardized or user defined;
- Descriptions of how to invoke web services, by means of pointers to service description documents, stored outside the registry, for example, at a service provider's site.

A UDDI registry contains web services descriptions with four different kinds of information elements described as follows:

- **businessEntity:** An organization that provides web services, including the company's name, address, and other contact information.
- **businessService:** A group of related web services offered by a businessEntity. Typically, it corresponds to one kind of service (such as a procurement or reservation service).
- **bindingTemplate:** Technical information needed to use a web service, such as the address at which the web service can be found and references to documents (called tModels) that describe the web service interface and other service properties. It also defines how operation parameters should be set and what the default values are.
- **tModel:** Technical model, which is a container for any kind of specification. For example, it might represent a WSDL service interface, a classification, or an interaction protocol, or it might provide the semantics of an operation.

2.8 WEB SERVICES AND B2B COLLABORATION

B2B collaboration has passed through differnet stages of interoperability technology from proprietary standards such as CORBA, and DCOM. Presently, the eXtensible Markup Language (XML) is a better tool for interoperability. XML is statisfying the business need being a platform independent technology.

SOA and XML web services based on Internet standard protocol (HTTP) has redefined e-business as collaborative commerce. Through the Internet, companies collaborate with suppliers, customers, distributors, and service providers to conduct business functions as well as to provide value for customers. Business collaboration turns participating companies into virtual enterprises that emphasizes rapid information exchange among particpating companies and inter-organisation systems. Such collaboration also facilitates information sharing, transparency, data integrity, and flexibility. Other benefits that businesses can derive from such integration also include: cost and time reduction, and improved collaborative planning and forcasting. Web services technology can be used to implement various B2B collaboration options [9].

Table 1.1: Web services for B2B collaboration: Options, requirements, and value propositions

Collaboration Options	**Requirements**	**Values of Web Services**
Buyer-based, one-to-many private exchange	Forge a strong collaboration with supply chain partners	Lower cost of transactions, increased integration
Seller-based, one-to-many private exchange	Foster colaboration with the end customers	Customer retention
One-to-one proprietary linkages	Extend a firm's traditional EDI or EAI integration	Enhanced application integration
Independent, public many-to-many exchange	Strengthen the role of intermediary in the exchange	Economies of scale, security, access
Consortia-based many-to-many exchnge	Attain common goals of participating companies	Process integration, flexibility

Web services for e-collaboration require the integration of data, application, and business processes:

- **Data-level integration:** this involves the movement of data between application. It requires the use of enterprise-wide metadata to define the mapping and neccessary transformations required to move data between applications. Data-level integration requires an understanding of both database structure and its usage.
- **Application-level integration:** this enables integration of message from any source. Message exchange is carried out using asychronous message flow to reduce dependency between application.
- **Process-level integration:** this is essentially an e-business colaboration. It defines the business processes underlying corporate strategies, addresses business processes crossing application boundaries.

2.9 CHALLENGES IN ENTERPRISE ADOPTION OF WEB SERVICES

The challenges enterprises encounter in adopting Web services for B2B integration include: inadequate employee skills to support implementation, insufficient senior management support, a lack of technical standards, security and poor application interface. Challenges encounter in adopting web services may also come in form of environmental factors such as: un-readiess of business partners and industry inertia. Visibility of the service(s), complexity, and tools also pose marginal challenges.

2.10 WEB SERVICES TECHNOLOGY FOR PRIVATE EXCHANGE

Although public web services promise to facilitate e-business collaboration, they are limited because they only offer functionality for revealing information such as data sourcing, number conversion, sensing, or data processing. Web services can be used within private exchnges on either the buyer-side or seller-side one to many exchnage. The important factor is the need for companies to maintain agile relationship with partners and protect the identity of services the business offers from anonymous access, and to cement relationships through business agreement not only through process integration.

CHAPTER THREE

REQUIREMENTS AND DESIGN

3.1 REQUIREMENTS FOR B2B E-PROCUREMENT

B2B E-commerce application has its roots in electronic data interchange (EDI) networks established between large buyers and suppliers. It is used to conduct business transactions involving the exchange of goods and services between two or more parties using electronic tools and techniques. The main goal of a B2B E-commerce application is to improve the order and delivery process between a company and remotely located business partners, and thus needs to satisfy various requirements, which are summarized below [1, 5, 9, 10]:

- **Harmonious communication:** B2B E-commerce includes the use of exchanges – internet-based system with which companies can collaborate to purchase or sell a variety of products. It thus must provide a means of harmonious communication between users and developers.
- **Reduced processing time:** E-commerce system increases the processing speed, accuracy, and efficiency of business and personal transactions. Through B2B exchanges, companies can receive rapid responses, shorten fulfillment cycles, and implement just-in-time procurement strategies, which help reduce lag times and allow companies to be more effective;
- **Reduced processing cost:** Online B2B exchanges enables companies to directly improve their order-to-fulfillment cycle by streamlining work-flow and business processes so as to achieve better order processing and tracking, better leverage company spending, increase return on investment, and ultimately optimize overall procurement efficiency;
- **Accuracy of business and transactions data**: Exchange technologies are basically web sites that use a standard language, XML, to facilitate application-to-

application data exchange. XML allows information regarding orders, purchases, payments, and products to be easily understood by other computers.

- **Expand Revenues** -- The public exchange of information provided through B2B exchanges has allowed many companies that sell to other companies to reach a greater number of potential commercial buyers of their products, which has led to increased sales. It also provides greater visibility between customers and suppliers.
- **Shortened systems development life cycle** – B2B systems requires implementation approach that is efficient and quickly implemented in order to achieve quick response to changing users requirements and market demand.
- **Reliability** – E-procurement system should work robustly without loss of any transaction data.

3.2 THE PROPOSED E-PROCUREMENT FRAMEWORK

The proposed SOA-based E-procurement framework is shown in figure 3.1. The framework shows 3-tier architecture in which Covenant University is used to model this B2B interaction because it is constantly interacting with a number of businesses to service the various departments within the institution.

A 3-Tier Architecture (3TA) is customarily used when building web applications. It makes a logical separation between the presentation layer, the business logic layer, and the database layer. The Service Oriented Architecture (SOA) builds on top of the 3TA. Instead of considering IT infrastructure as a set of application tower, SOA looks at a set of services and applications. Services implement some kind of functionality and are used by applications and other services. Services communicate between each other and with applications by exchanging XML documents. Comparing SOA to 3TA, SOA applications (typically service consumers) correspond to the 3TA presentation layer, while SOA services correspond to the 3TA business logic and data layers.

As shown in figure 3.1, the data layer comprises of various departments within Covenant University that can use the E-procurement Web service module to reveal procurement

information. The application layer shows the business entities that can be users (service consumers) of the E-procurement application through remote invocation of the web service methods.

The E-procurement web service module lists the service interfaces used to publish procurement information for sourcing tender. The internal procurement processes of the architecture shows the procurement functions that are to be carried out between requesting departments and the organisations' procurement officers.

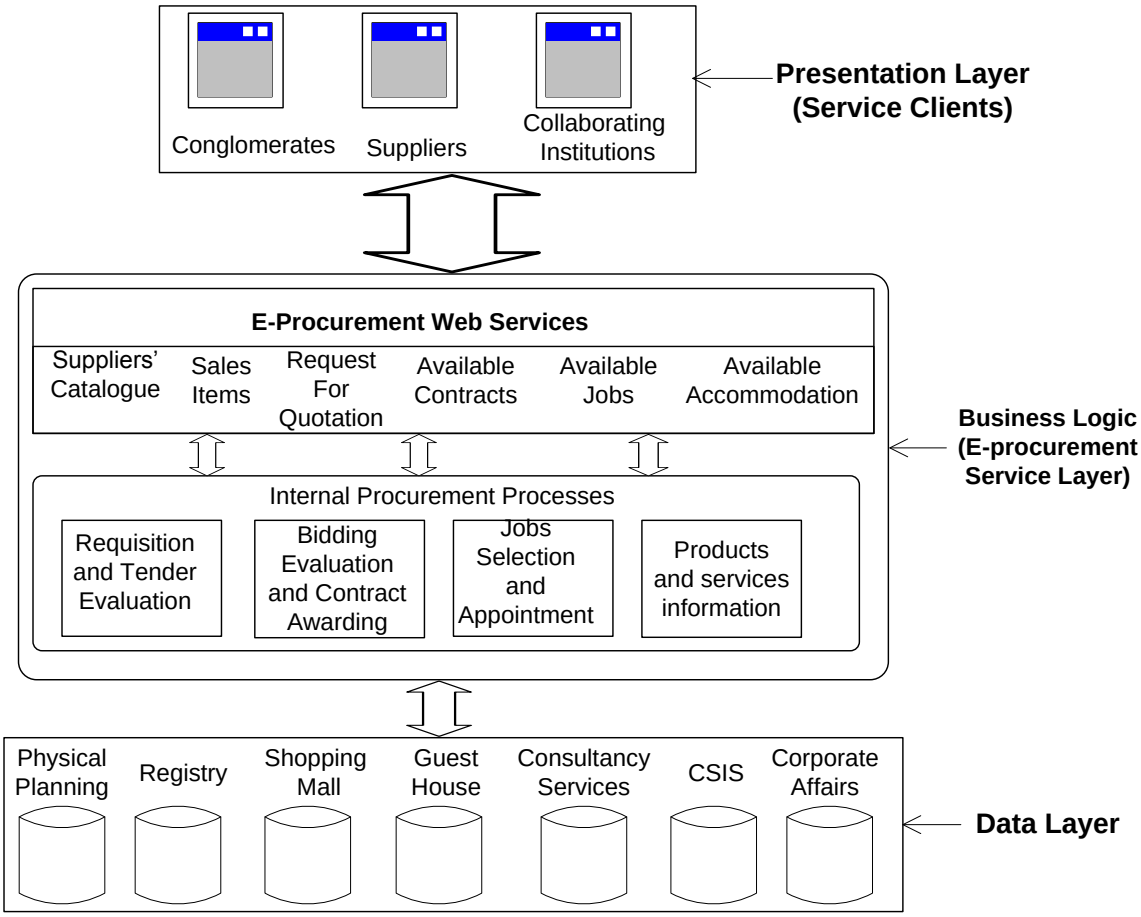

Figure 3.1: The SOA-Based E-procurement Framework

Figure 3.2 shows the contextual view of the SOA-based model for inter-organisation exchange of E-procurement information between Covenant University, conglomerates and other business partners all referred to as service consumers. This model is based on

distributed computing approach for integrating heterogeneous application over the internet. Facilitate just-in-time integration and allow businesses to establish new partnership easily, irrespective of their programming language or operating system.

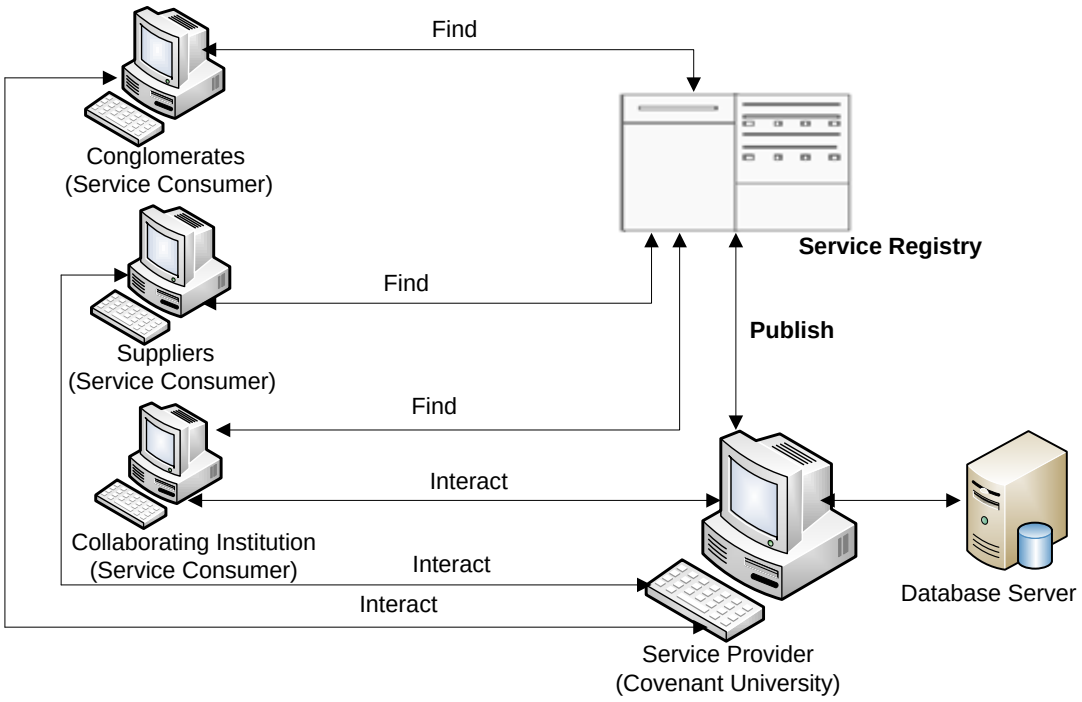

Figure 3.2: Context Diagram of the E-procurement Framework

3.3 SYSTEM DESIGN

3.3.1 USE CASE DIAGRAM

The use case is used to model the behavior of the system that is visible externally. The use case specifies the interaction between the system and the users of the system to achieve a particular goal.

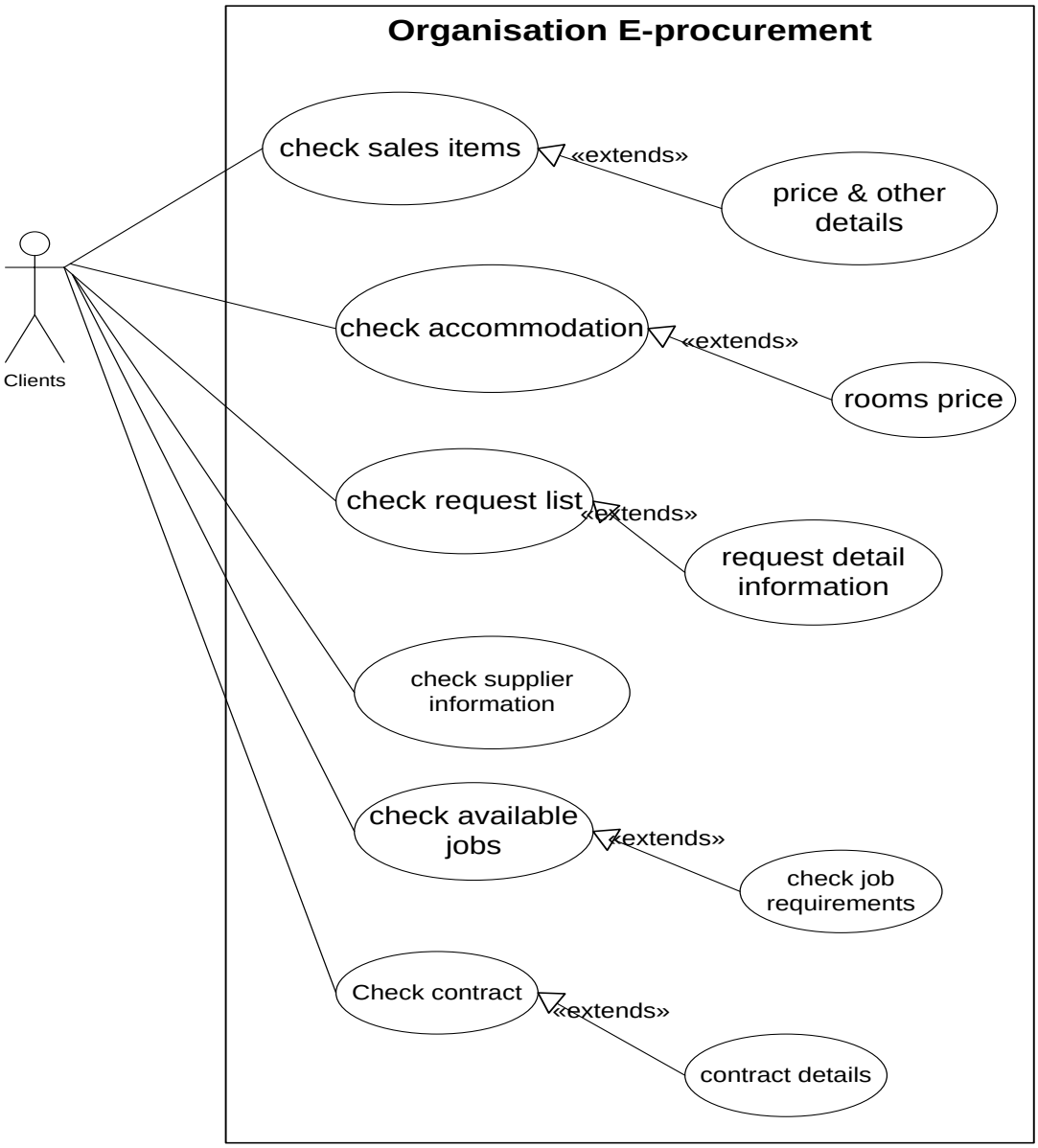

Figure 3.3: Use Case diagram for the web service

3.3.2 CLASS STRUCTURE OF THE E-PROCUREMENT FRAMEWORK

The web service layer is basically a very small layer of the application. The web services methods are exposed to the client but the implementation of those methods are internal to the implementation classes. The advantage of this architecture is that the implementation of the web service can be changed in future while keeping the web service interface stable. The E-procurement web service is a single web service that deals with managing

and retrieving of procurement information. The implementation class for each of the web methods exposed by the web service calls the database access class, wraps the corresponding SQL statements that controls the retrieval of data from the database and compose of corresponding classes that get input and return output to the users.

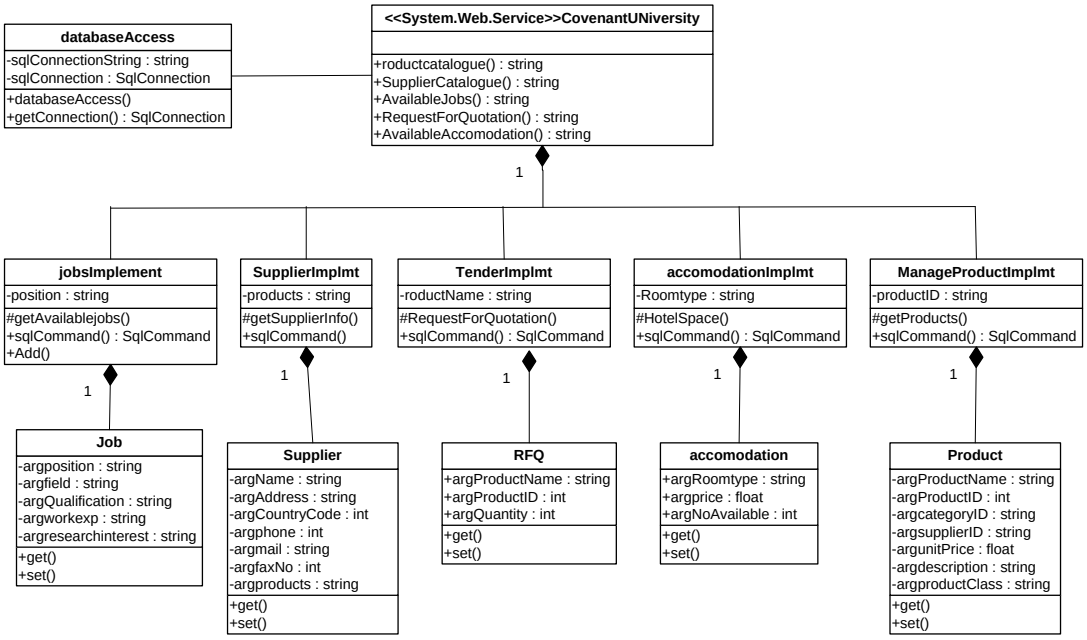

Figure 3.4: Class Structure of the Web Service Layer of the E-procurement Framework

3.3.3 DESIGN PATTERN OF THE E-PROCUREMENT FRAMEWORK

A model is a graphical representation of the functionality, or behavior of the system. Systems model play important role in system development [27]. It helps to give a pictorial representation of reality with respect to functionality, or behavior of the system that will satisfy the needs of clients or users. Modelling is an activity carried out with the aim of producing a correct, complete and consistent representation of the Real World – or more precisely that part of the Real World which is of interest to the designer of the target Information System. Basically, there are two types of model: *Logical model* and *Physical model* [29]**.**

In modelling the proposed system, the Facade pattern is used to illustrate how the system is physically and technically implemented. Facade pattern is a design pattern approach

used to provide a simplified view of the system when it involves manipulating many different classes.

The Façade class is an interface for an entire subsystem. It is a high level layer that makes the subsystem easier to use.

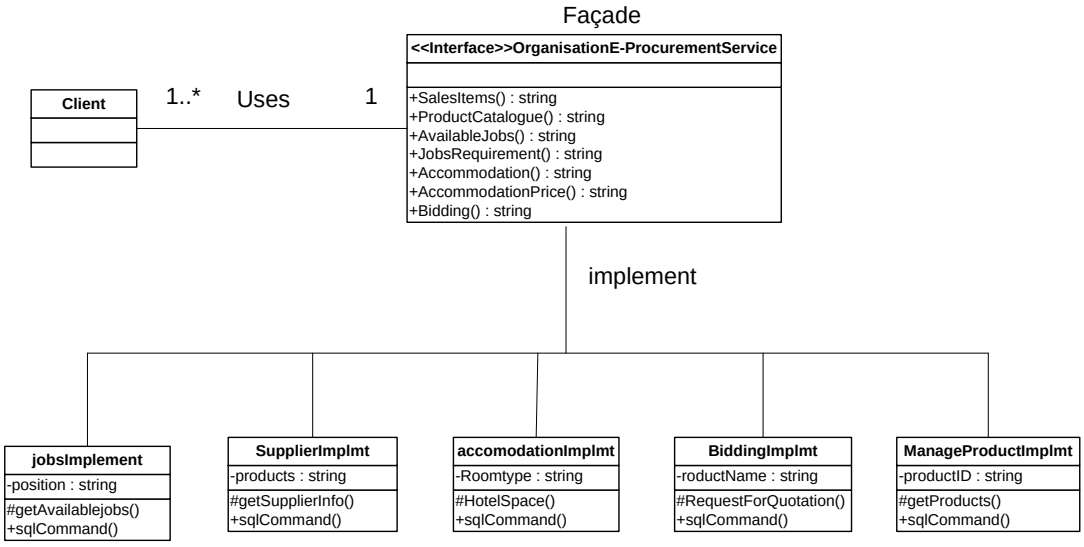

Figure 3.5: Façade pattern of the web service layer of the E-procurement framework

3.3.4 SEQUENCE DIAGRAM

Sequence diagram models the interaction between objects in the system. It depicts how objects interact with each other via messages in the execution of a use case or operation. They illustrate sequential representation of message passing between objects

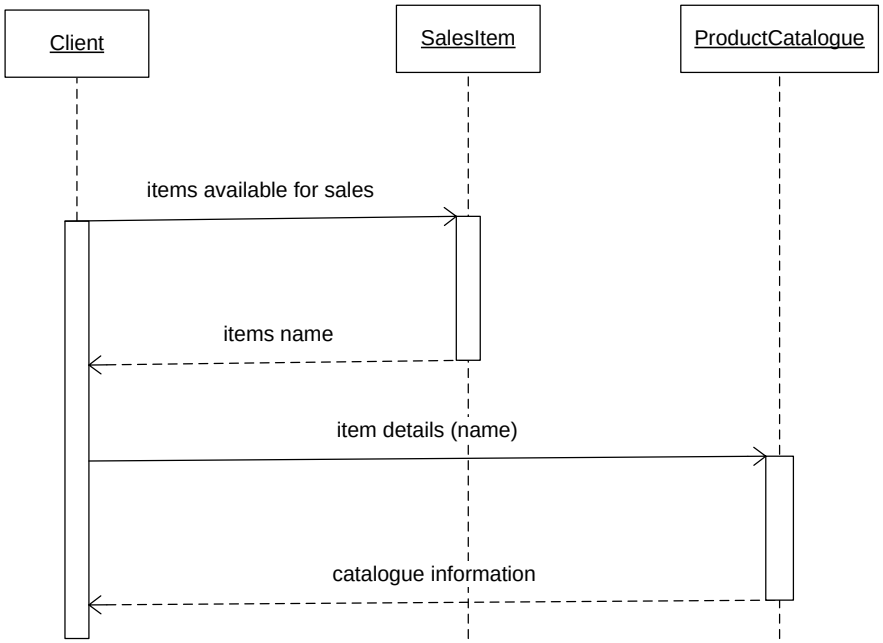

Figure 3.6: Invocation of the Sales Item Information Service

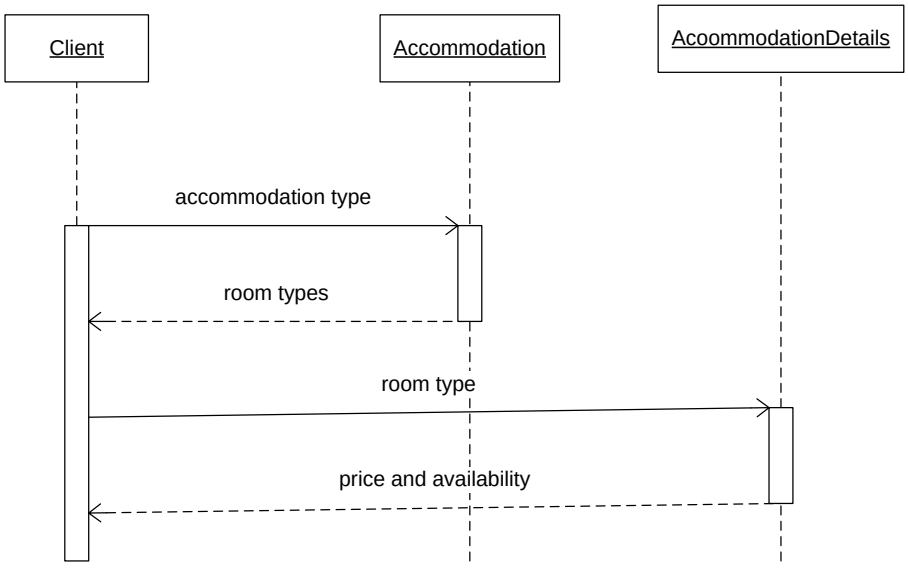

Figure 3.7: Invocation of the Accommodation Information Service

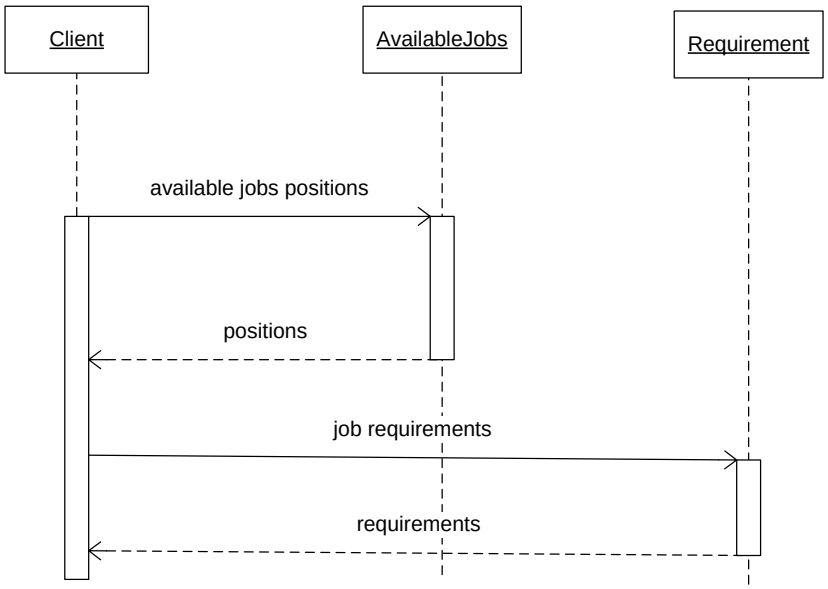

Figure 3.8: Invocation of Jobs Information Service

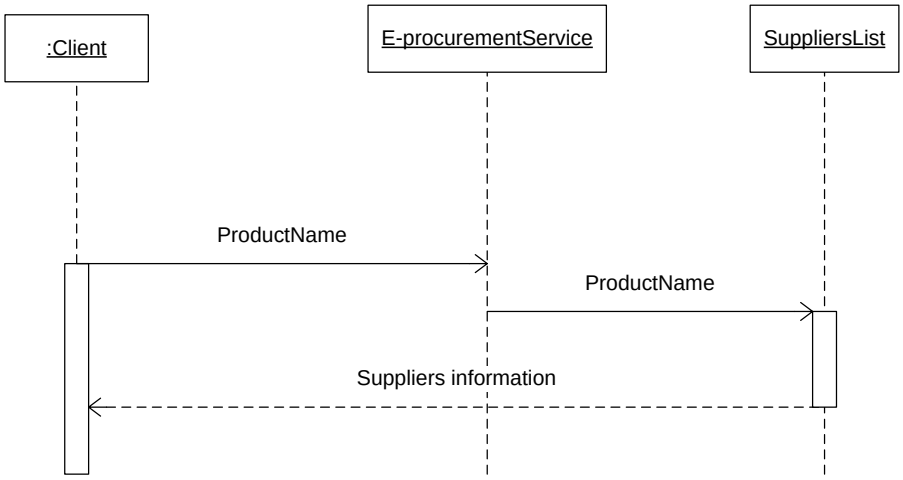

Figure 3.9: Invocation of Suppliers' information Service

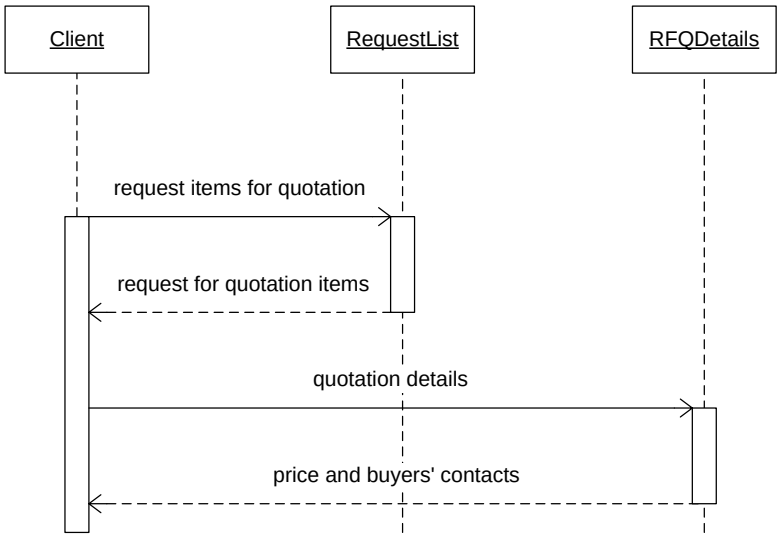

Figure 3.10: Invocation of Purchase Request Information Service

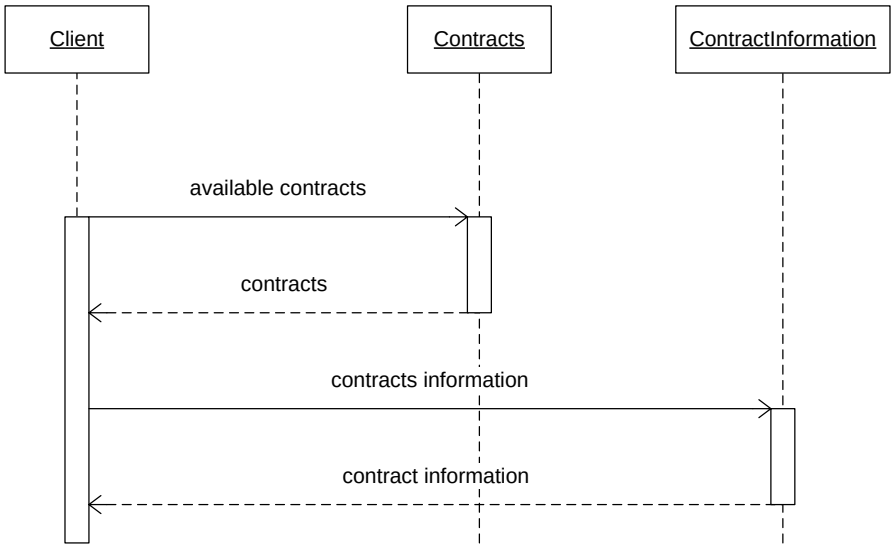

Figure 3.11: Invocation of the Available Contracts Method

3.4 DATABASE DESIGN

A database is a collection of information stored in a computer in a systematic way, such that a computer program can consult it to retrieve information. A database management

system is needed to access and process stored data. For the implementation of this work, a relational database Microsoft SQL Server 2005 was used as the database management system.

A database named cuservicesdb was created in Microsoft SQL Server Management Studio with six tables to facilitate an efficient query system

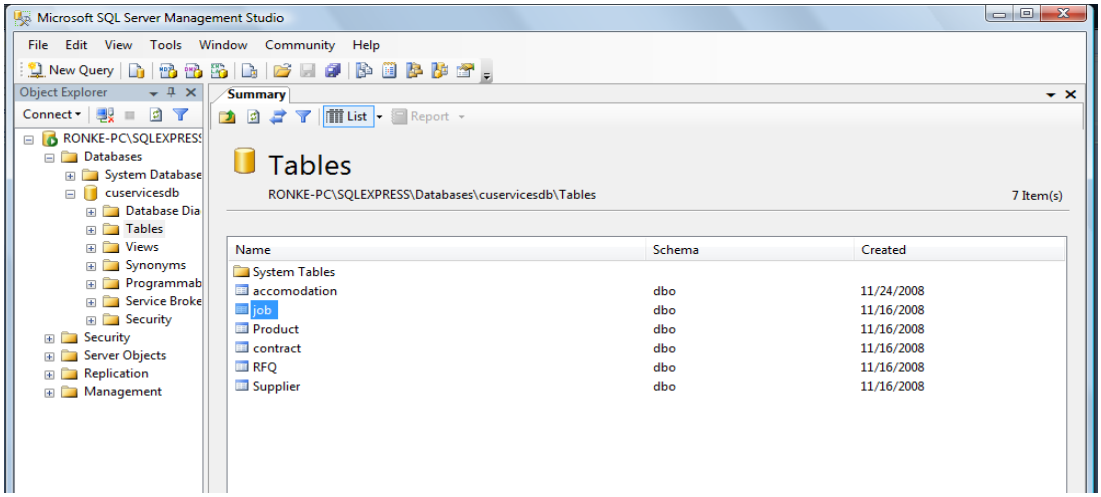

Figure 3.12: Diagrammatic representation of the cuservicesdb containing six tables

3.4.1 DESCRIPTION OF TABLES

1. **Accommodation Table**

   As shown in figure 3.14 below accommodation table is made up of three (3) fields; Roomtype, price and NoAvailable. This table is used to store information pertaining to the type of rooms available at the quest house.

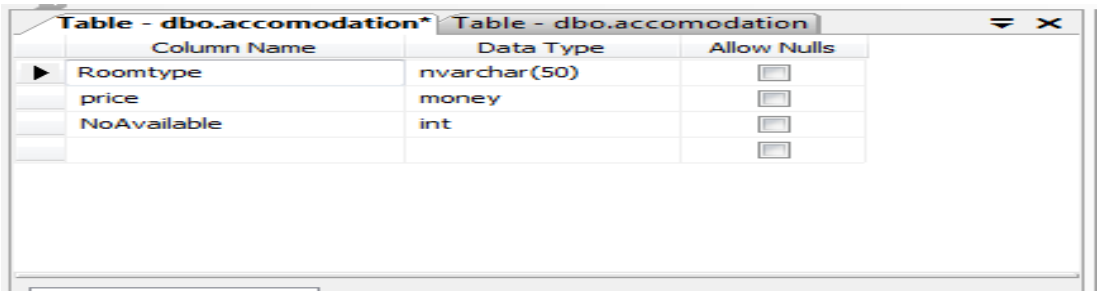

Figure 3.13: Diagrammatic representation of the accommodation table

2 Supplier Table

The table gives a detailed contact of the supplier or business partners which Covenant University which to expose through their web service

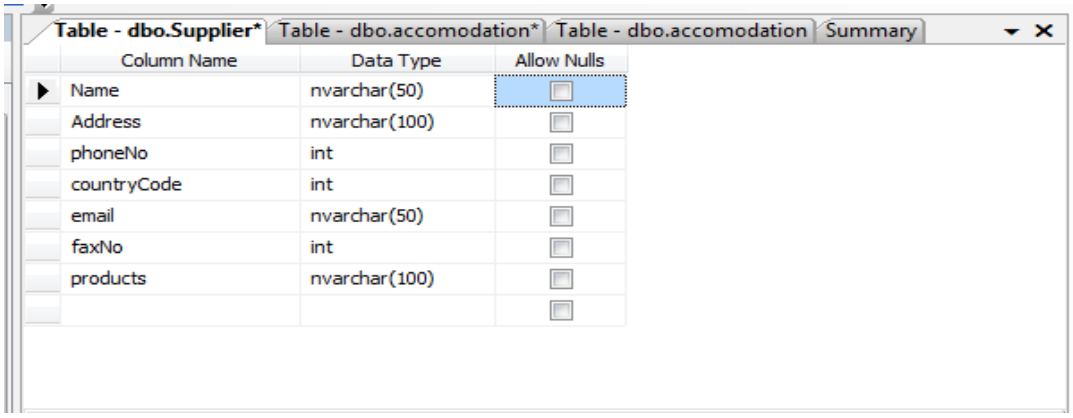

Figure 3.14: Diagrammatic representation of the supplier table

3 Product Table

The table Product gives a detailed description of the items available for sales within Covenant University.

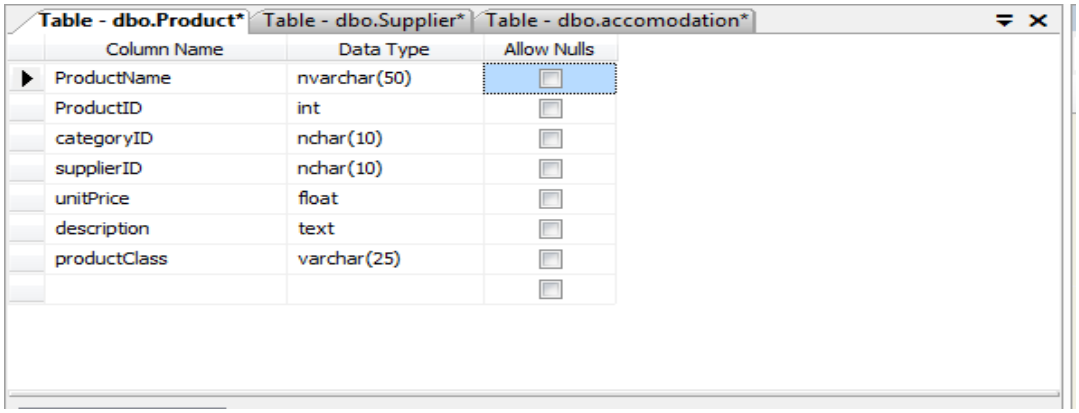

Figure 3.15: Diagrammatic representation of the Product table

4 Job Table

The table Job stored information of the available job vacancy for recruitment.

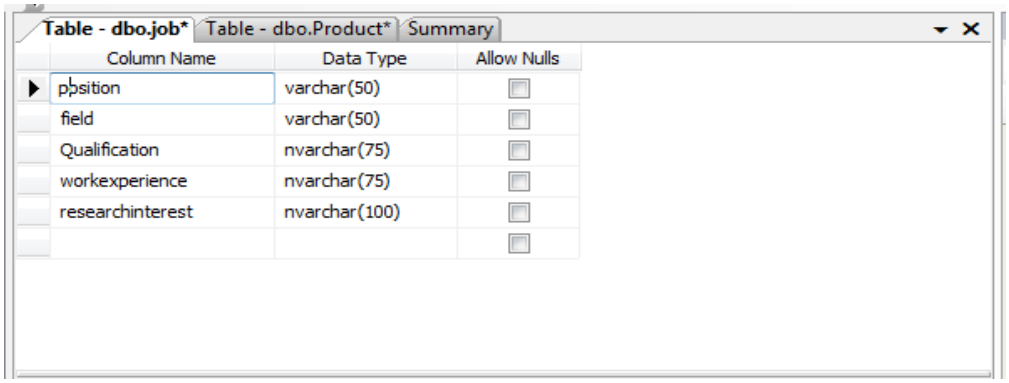

Figure 3.16: Diagrammatic representation of the Job table

CHAPTER FOUR

IMPLEMENTATION

4.1 INTRODUCTION

After achieving the design of the proposed system, this chapter discusses the implmentation procedure of the proposed system. The system implementation of the SOA-Based Framework was achieved using web service technology. The web service provides service interfaces which enable clients to interact with the provider's server in a more general way than web browsers. Coupled with the implementation of the web service, two ASP.Net web site applications were developed to test the real life invocation of web service methods.

4.2 IMPLEMENTATION DETAILS

This section describes in details the web service application called "Organisation E_procurement".

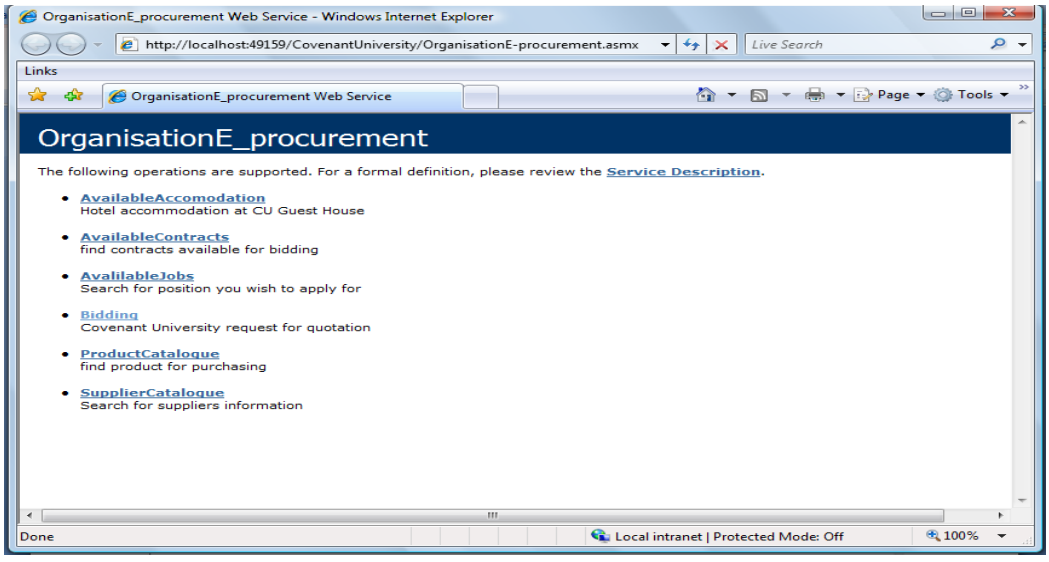

Figure 4.1: A view of the web method interfaces

The web service figure 4.1 comprises of six web method interfaces:

1. AvailableAccommodation,
2. AvailableContracts,
3. AvailableJobs,
4. Bidding,
5. ProductCatalogue, and
6. SupplierCatalogue for publishing different classification of information as identified within the Covenant University context. The first line of the web service page gives the hyperlink to the Service Description (WSDL file). The various links (method interfaces) helps to navigate to the method invocation page with which one can test the method using HTTP POST protocol.

4.2.1 SERVICE DESCRIPTION

Service description allows the client to communicate with the services. It gives description of the message and response format that clients need to understand to bind-to and interact with web service as internet applicationst. The Web Service Description Language (WSDL) is an XML file consisting of five parts, wrapped in the <definitions> XML element:

- The <Type>: defines all the data types used by the service. It gives both the data type of the input and output variable
- The <Message>: defines the input and the output parameter of the web service and also help remote clients to know the kind of access method to be used in communicating with the web service.
- The <PortType>: ties the access methods to the messages declared in the <message> section.
- The <bindings>: the binging section declares the protocols (SOAP) used to access the web methods. It also defines the encoding used to send data over the wire.
- The <Service>: gives the name of the web service, *OrganisationE_procurement,* its methods and the URL of the web service *CovenantUniversity.com*

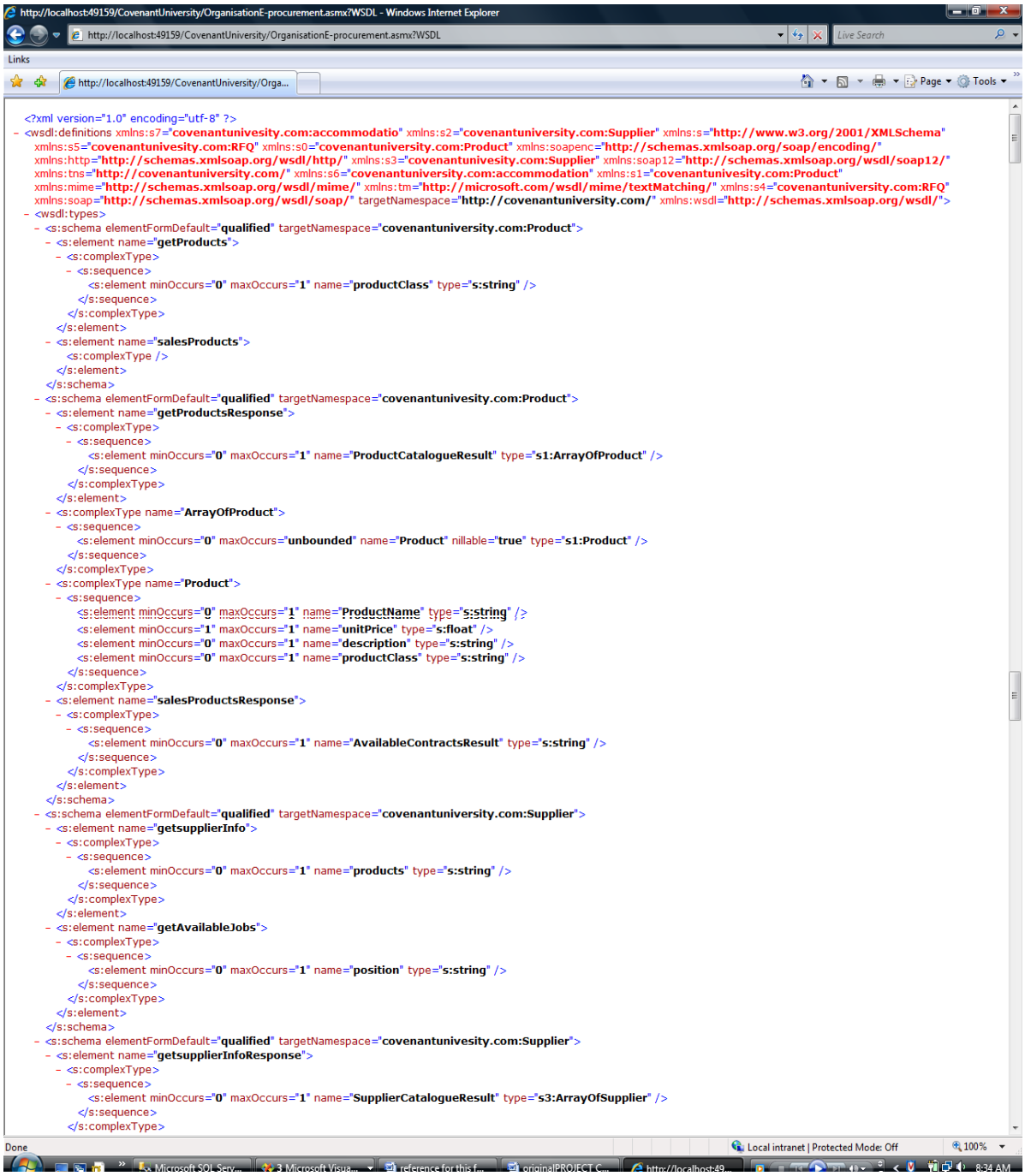

Figure 4.2: WSDL description of the OrganisationE_procurement web service.

(See Appendix for detailed description)

4.2.2 TESTING WEB SERVICE WITH A REMOTE APPLICATION

Before web service methods can be consumed by another application, the service provider must have published the WSDL in either a public registry or private registry as desired by the service provider. Then an interested service consumer can find the WSDL and use the information in it bind to the web service via the Net without bordering to know the service provider.

To test web services with web application on the standalone machine, Visual Studio .Net 2.0 framework comes with the capability to add web reference for applications on the same localhost. After successful completion of the "Add Web Reference" process the WSDL file and two other files with extension "disco" and "discomap" are added to the web application and therefore, we can invoke the web service method with a click of a button.

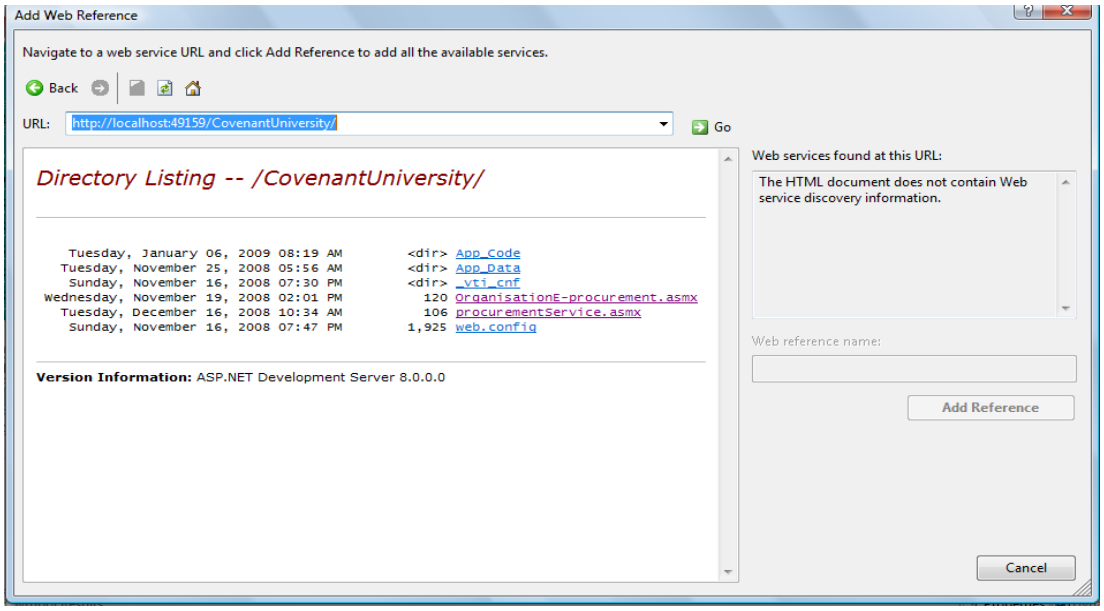

Figure 4.3: Adding Web Reference to Service Client Application

4.2.3 DESCRIPTION OF THE WEB SERVICE METHODS WITH CLIENT APPLICATION INTERACTION

1. AvailableAccommodation

The AvailableAccommodation method was used to publish accommodation information at the University Guest House. Figure 4.4 shows how the method can be invoked from

the service interface and SOAP format of the request and response of the web method. figure 4.5 shows the XML formatted text of the server response that is passed through the network when the method is invoking.

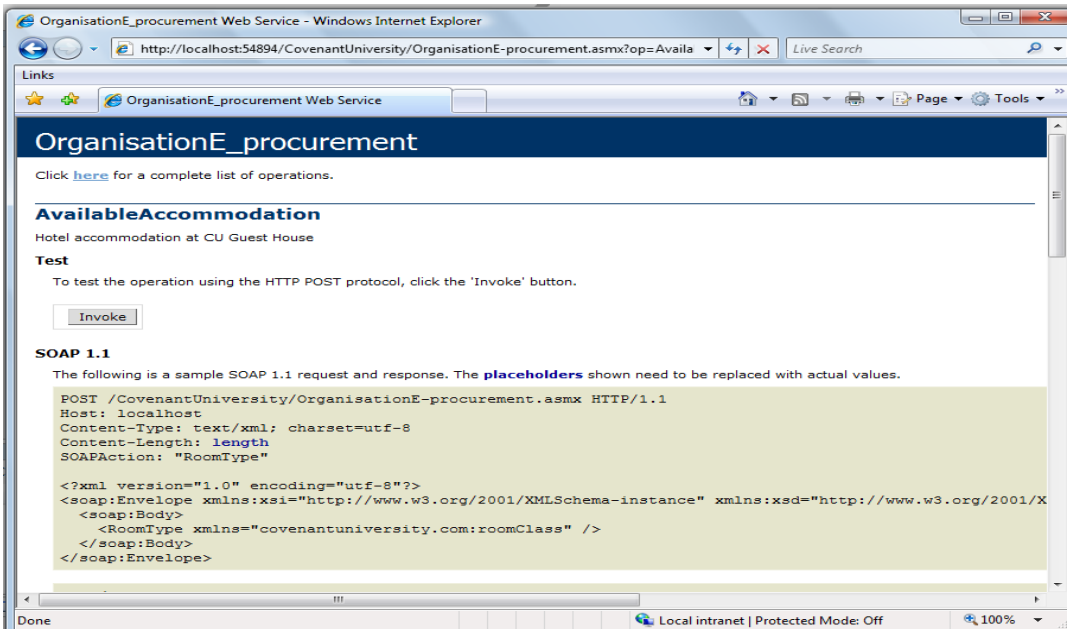

Figure 4.4: Invoking the AvailableAccommodation Method

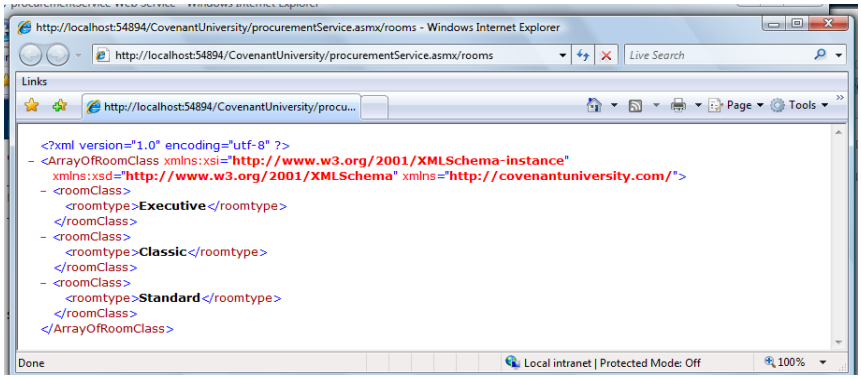

Figure 4.5: XML formatted response of invoking AvailableAccoomodation Method

Figure 4.6 depicts remote web application interation with the web service. When clients interact with web services method, the output comes in form of normal text though XML it is conveyed through XML tag.

When a user click the "Accommodation Available" buttom to invoke the AvailableAccommodation method, the Combo box is load. Then the user can select from the various Room types and click on "View price and Availability" button to see the price for the sellected room type.

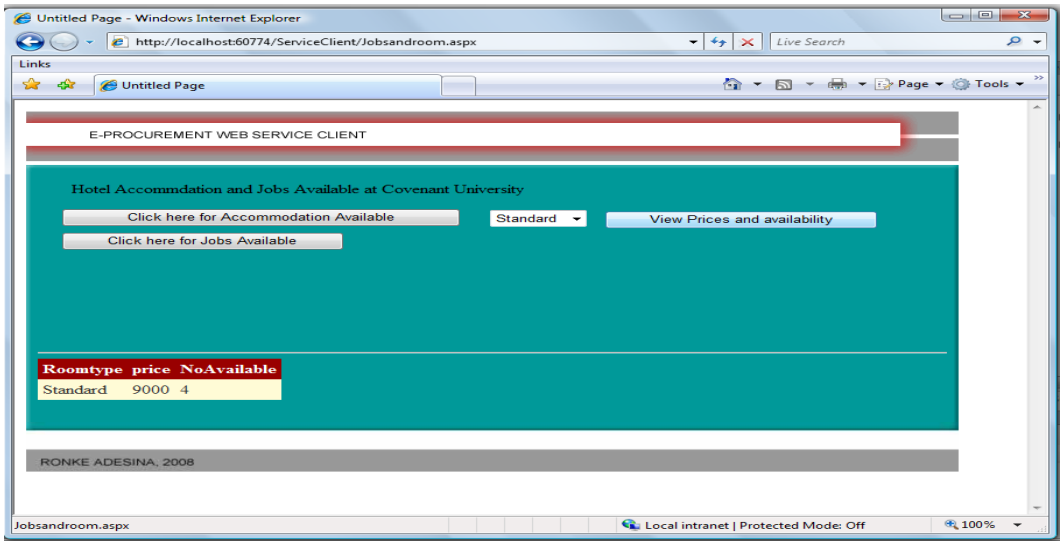

Figure 4.6: Client invocation of AvailableAccommdation Method

2. AvailableJobs

The AvailableJobs method is used to publish recruitment information. Relevant recruitment information such as Academic qualification, research interest, work experience can on available Position can be gotten be invoking the method.

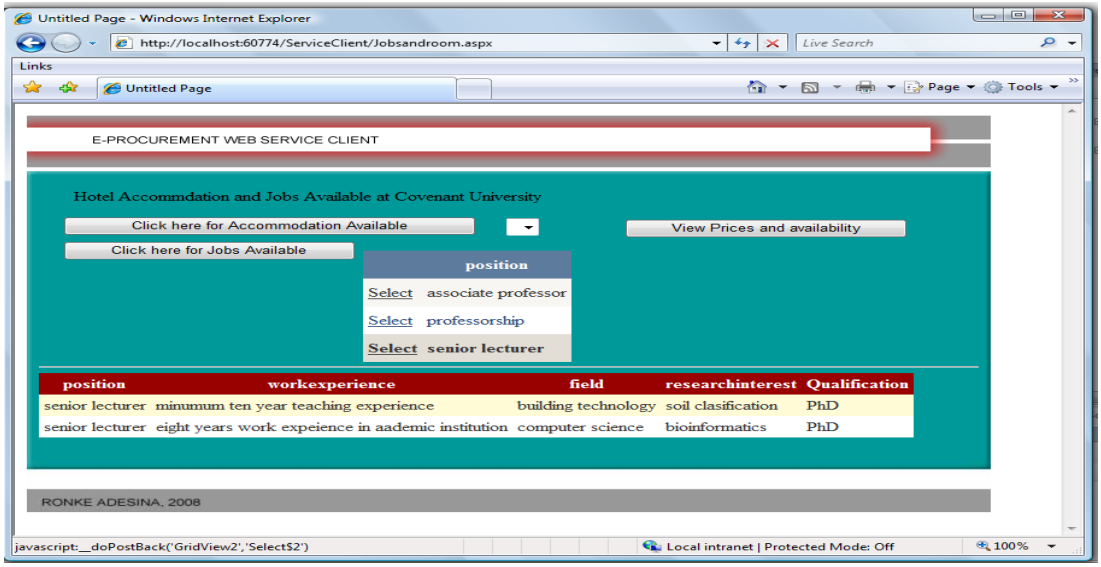

Figure 4.7: Client Invocation of *AvailableJobs* Method

Clicking on the button “Click here for Jobs Available” displays different postions for available job opportunity. Users can select to access the jobs criteria.

3. Bidding

The method was developed to source for tender. It publish information such as quantity, specification and name of goods suppliers can tender to supply.

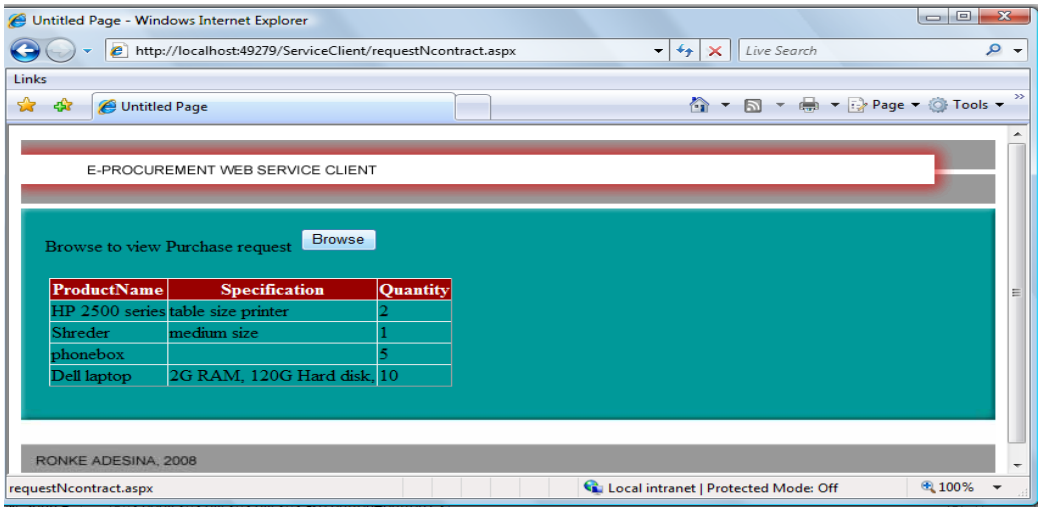

Figure 4.8: Client Invocation of the Bidding Method

4. ProductCatalogue

The ProductCatalogue web method provides a catalogue of items available for sales within Covenant University. Invoking the method “ProductCatalogue” (figure 4.9) returns various classes of items available for sale. Selecting any of the items gives more information concerning the particular product class

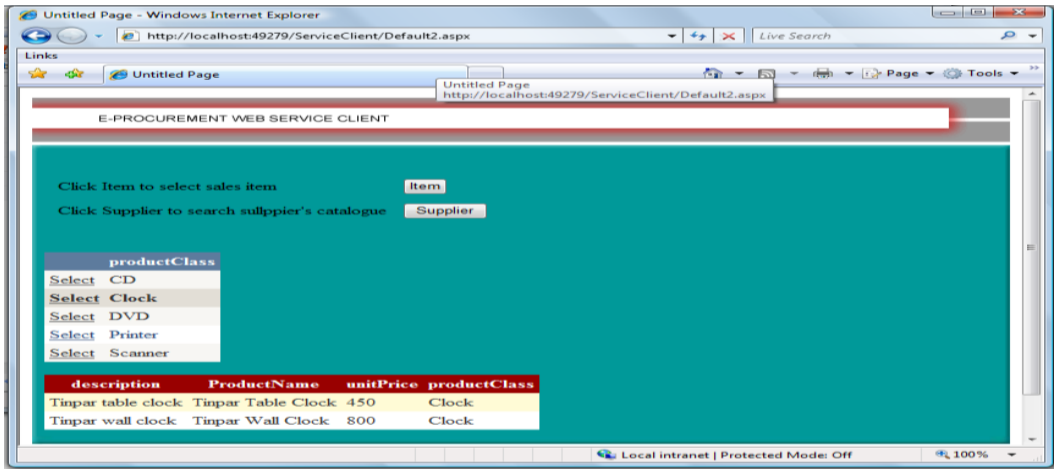

Figure 4.9: Client Invocation of the *ProductCatalogue* Method

5. SupplierCatalogue

The web method "SupplierCatalogue" interact with the database to provide information about businesses involving in supply of goods to Covenant University. The method returns information including Name, Address, and Phone number of suppliers of any product that is selected. Figure 4.10 shows client interaction with "SupplierCatalogue" method.

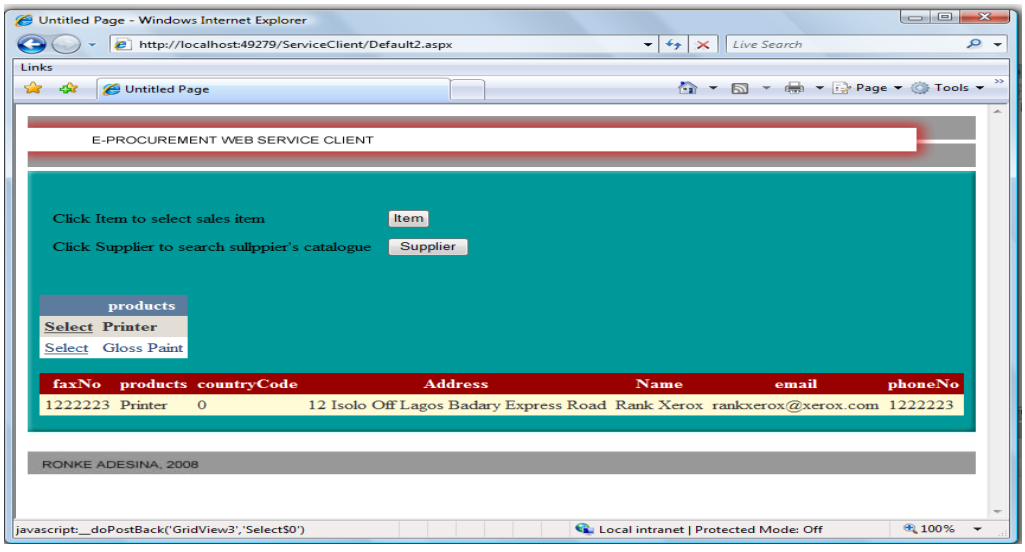

Figure 4.10: Client invocation of *SupplierCatalogue* Method

CHAPTER FIVE

CONCLUSION AND RECOMMENDATION

5.1 SUMMARY

In this work, we have been able to achieve the stated aim and objectives of the study which are as follows: 1) we identified the requirements for efficient B2B oriented E-procurement framework which are: Harmonious communication, reduced processing time, reduced processing cost, expand revenue, accuracy of business and transaction data, shortened systems development life cycle and reliability; 2) we developed an SOA-based framework for E-procurement; and 3) we designed and implemented an E-procurement system prototype that is based on the proposed SOA framework.

In implementing this concept, we used Covenant University as a case study; we identified the various units that can take advantage of E-procurement system within the institution. A single access point for revealing procurement information is provided so as to achieve transparency, uniformity and convergence of concerns. The framework proposed provides a distributed computing technology for revealing the business services of applications on the Internet using standard XML protocols and formats. This is to achieve interoperability and application level integration with other institutions or business entities partnering with Covenant University.

This framework can be applied in any multi-tiered organisation or government institutions. The E-procurement Framework can be put to use in linking disparate applications and services for the purpose of mergers, acquisitions, and virtual integration.

5.2 CONCLUSION

This study exploits the concept of SOA in achieving integration and interoperability in the procurement model of business organisations. The various existing tools for interoperability in business were explored with weakness associated with each one identified. The advantage of web service technology in achieving inter-organisation

integration, platform independence interoperability and loose coupled integration over the proprietary interoperability tools (CORBA, DCOM, and RMI) was discussed. In conclusion, we presented a model of interoperable SOA-based E-procurement framework for multi-tiered business organisation, since integration and interoperability is a requirement for survival in E-business.

5.3 FUTURE WORK

In future work, an evaluation of the proposed E-procurement framework will be undertaken. This will entail an evaluation of several aspects of the E-procurement framework so as to ascertain the performance and users' acceptance of the system. In carrying out the evaluation, qualified functions such as: overall operation of the system, system flexibility, ease of startup, compatibility with other systems, and collaborative tools availability would be assessed.

Furthermore, we will look at enhancing the architecture with enterprise-class messaging system such as ESB. This is to ensure data and process level integration through the use of loosely coupled services.

REFERENCES

1 Panayiotou N.A., Gayialis S. P., and Tatsiopoulos I.P., "An E-procurement system for governmental purchasing", International Journal of production economics, Vol 90 No. 1, pp. 79 – 102, 2003.

2 Natis Y. V., "Service-Oriented Architecture Scenario", Gartner Publication, www.gartner.com/resources/114300/114358/114358.pdf, 2003.

3 Office of the Government UK, "E-procurement Interoperability Requirement", www.govtalk.gov.uk/documents/OGC-eProc-v4-0.pdf, 2004.

4 Cheung C.F., Wang W.M., Victor Lo and Lee W.B., "An agent-oriented and knowledge-based system for strategic E-procurement", Journal of Expert Systems, Vol. 21, No 1. pp. 11-21, 2004.

5 Economic and Social Commission, Asia and the Pacific, "E-procurement", www.unescap.org/publications/detail.asp?id=1157, 2006.

6 Department of Communication Information and Arts, Australian Government, "From paper to procurement", www.dbcde.gov.au/data/assets/pdf file/0017/20447/ecatalogue28-4.pdf, 2004.

7 Endrei M., Ang J., Arsanjani A., Chua S., Comte P., Krogdahl P., Luo M., and Newling T., "Patterns: Service-Oriented Architecture and Web Services" www.ibm.com/redbook, 2004.

8 Gueritz A., "Understanding the Benefits of E-procurement", Sixhills Consulting Ltd & Authors, http://www.sixhills-consulting.com, 2001.

9 Khosrow-Pour M., "Encyclopedia of E-commerce, E-Government and Mobile Commerce", Information Resources Management Association, USA, 2006.

10 Sheldon F.T, Jerath K., Pilskalns O., Kwon Y., Kim W., and Chung H., "Case Study: B2B E-commerce System Specification and Implementation Employing Use-Case Diagrams, Digital Signatures and XML", Multimedia Software Engineering, Proceedings, pp. 106 – 113, 2002.

11 David Sprott, "Service Oriented Architecture: An Introduction for Managers", *http://www*.roadmap.cbdiforum.com/reports/soa/managersintro.php, 2004.

12 Stevens M., "Service-Oriented Architecture Introduction, Part 2", http:// softwaredev.earthweb.com/msnet/article/ 0,,10527_1014371,00.html, 2004.

13 Sprott D., and Wilkes L., "Understanding Service-Oriented Architecture" http://www.developer.com, 2004.

14 Keen M., Acharya A., Bishop S., Hopkins A., Milinski S., Nott C., Robinson R., Adams J., Verschueren P., "Patterns: Implementing SOA Using an Enterprise Service Bus", www.ibm.com/redbooks, 2004.

15 Fiorano Software, Inc., "Service Oriented Architecture Implementation Frameworks", www.fiorano.com.

16 David B., Haas H., McCabe F., Newcomer E., Champion M., Ferris C., Orchard D., "Web Services Architecture. W3C Working Group Note 11", http://www.w3.org/TR/2004/NOTE-ws-arch-20040211, 2004.

17 Michael Champion, "Web Services Architecture W3C Working Draft 14 2002" http://www.w3.org/TR/2002/WD-ws-arch-20021114, 2002.

18 Berners-Lee T., "Web Services Program Integration across Application and Organization boundaries" www.w3.org/DesignIssues/WebServices.html, 2003.

19 Nickull D., Reitman L., and Ward J., "Service Oriented Architecture (SOA) and Specialized Messaging Patterns", http://www.adobe.com/enterprise/pdfs/ Services_Oriented_ Architecture_from_ Adobe.pdf, 2007.

20 Turtschi A., Werry J., Hack G., Albahari J., Meng-Lee W., "C# .NET Web Developer's Guide". Syngress Publishing, Inc., USA, 2002.

21 Adebiyi A. A., "Support Tools for Interoperability of Information Systems and Databases Using XML". M.Sc Dissertation, Department of Computer and Information Science, Covenant University, 2006.

22 Coulouris G., Dollimore J., Kindberg T., "Distributed Systems Concepts and Design", Addison Wesley, 4th Edition, England, 2005.

23 Anderson S., Chapman M., Goodner M., Mackinaw P., and Rekasius R., "Supply Chain Management Use Case Model Version: 1.0" http://www.wsi.org/SampleApplications/SupplyChainManagement/2003 - 04/SCMUseCases1.0-BdAD.pdf, 2003.

24 Normén F., "Using-web-services-in-a-3-tier-architecture", weblogs.asp.net/ fredriknormen/archive/2008/11/05/using-web-services-in-a-3-tier architecture.aspxmembers.tripod.com/ChannuKambalyal/NTier Architecture.pdf. 2008.

25 WS-I Attachments Sample Application 1.0 EA5 http://java.sun.com/ webservices/docs/1.6/wsi-sampleapp/index.html.

26 Chapman M., Goodner M., Lund B., McKee B., Rekasius R., "Supply Chain Management Sample Application Architecture" http://www.wsi.org/Sample Applications/SupplyChainManagement/2003-04/SCMArchitecture1.0-BdAD.pdf, 2003.

27 Whitten J. L., Bentley L. D., Dittman K. C., "Systems Analysis and Design Methods", McGraw Hill, 6th Edition, New York, 2004.

28 Bennett S., McRobb S., and Farmer R., "Object-Oriented Systems Analysis and Design using WML" McGraw Hill, 2nd Edition, London, 2001.

29 Knowles C., Mohr S., Palermo J.M. Siegers P., and Singh D., "Professional ASP .Net XML with C#", Wrox Press Ltd, USA, 2002.

30 Carol B., and Doake J., "Software System Development: A Gentle Introduction", McGraw Hill, 3rd Edition.