ADAMA SCIENCE AND TECHNOLOGY UNIVERSITY
SCHOOL OF ELECTRICAL ENGINEERING AND COMPUTING

Integrating City Services Using Distributed Topics Messaging Architecture: The Case for Debre Markos City

A THESIS SUBMITTED IN THE PARTIAL FULFILLMENT OF THE REQUIREMENTS OF THE DEGREE OF MASTER OF SCIENCE IN SOFTWARE ENGINEERING

By
Tilaye Habtu Tsega

November, 2016

ADAMA SCIENCE AND TECHNOLOGY UNIVERSITY
SCHOOL OF ELECTRICAL ENGINEERING AND COMPUTING

Integrating City Services Using Distributed Topics Messaging Architecture: The Case for Debre Markos City

By

Tilaye Habtu Tsega

Advisor

Frezewd Lemma (PhD)

Name and Signature of the board of the examiners for approval

Name	Signature	Date
Mr. Dereje **Chair Person, Examining Board**		
Dr. Mesfin Abebe **Internal Examiner**		
Dr.Tibebe Beshah **External Examiner**		

Acknowledgment

First of all, I would like thank Almighty God who makes things possible and helps me in every aspect throughout my study.

Next, I would like to express my sincere gratitude to my advisor Dr. Frezewd Lemma for the continuous support for my thesis, for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped me in all the time of the research and writing of the thesis.

Besides my advisor, I would like to thank participants during data collection who have willingly shared their precious time during the process of interviewing. I would like to thank my loved ones, who have supported me throughout the entire process, both may keeping me harmonious and helping me putting pieces together. I would be grateful forever for your love.

Last but not list, I would like to thank my families, teachers, friends and others who helped me for the accomplishment of this thesis. Thank you all.

Declaration

The thesis is my original work and has not been presented for a degree in any other university or conferences and all sources of materials used have been acknowledged.

Tilaye Habtu

November, 2016

The thesis has been submitted for examination with my approval as university advisor

Frezewd Lemma (PhD)

November, 2016

Table of Contents

List of Figures

List of Tables

List of Acronyms

API	-	Application Programming Interface
CMS	-	City Management System
EAI	-	Enterprise Application Integration
E-City	-	Electronic City
ESB	-	Enterprise Service Bus
ICT	-	Information Communication Technology
IT	-	Information Technology
ITAA	-	Information Technology Association of America
J2EE	-	Java 2 Enterprise Edition
JMS	-	Java Message Service
JNT	-	Jinni Network Technology
JTS	-	Java Transaction Service
MDB	-	Message Driven Bean
MOM	-	Message Oriented Middleware
NGO	-	None Governmental Organizations
QOS	-	Quality Of Service
RPC	-	Remote Procedure Call
SMME	-	Small And Medium Enterprises
SOA	-	Service Oriented Architecture
TP	-	Transmission Protocol
UNDP	-	United Nations Development Program
URL	-	Universal Resource Locator
W3C	-	World Wide Web Consortium
WS	-	Web Service
WSDL	-	Web Service Development Language
XML	-	eXtensible Markup Language

ABSTRACT

Electronic governance is an important issue in managing a city. In a connected society, the needs for service integration have been increasing day to day. The advancements of ICT in today's world led to cities and city managers to take advantages of it. Cities consists of different information systems and sources for performing activities which needs to be communicated and integrated with each other, but these information systems are developed from a variety of technology platforms. This is a challenge for creating communication and integration among them. Service oriented architecture is a suitable paradigm to overcome this service integrity and interoperability challenges in a city services.

In this research work we have studied a variety of problems facing in service integration and focusing the benefits of service oriented architecture and developed an architecture to alleviate those challenges for efficient access and accurate delivery of services for consumers. Therefore we proposed a seven layered service oriented architecture for service integrity in Debre Markos city. The proposed service oriented architecture is examined using a case study scenario of two applications exchanging information based on publish-subscribe messaging model in combinations of distributed topics and durable subscribers architecture paradigm.

***Keywords**: Electronic city, service oriented architecture, Java message service, service integrity,*

CHAPTER ONE

1. INTRODUCTION

1.1.Background

Information technology (IT) created a lot of changes in all aspects of social and economic of human living and its impact of communities with an incredible speed from traditional and industrial societies to an information society.

The ITAA (Information Technology Association of America) is defining working fields in scope of studying, design, development, implementation, management and support of computer based systems [1]. Nowadays, using contributions of information technology hopefully launches societies known as information society.

The field to an information society has developed within subjects such as e-city (electronic city).

Electronic city is defined as a city in which most of its activities are performed through internet and electronic systems and its purpose is citizen's access to all organizations, urban areas and other required information in a sustainable, trustworthy and confidential manner in all 24 hours of the day [2].

On the other hand an electronic city is a program that focuses on modernization and redevelopment of the engineering networks to establish a public access of information at a certain place [3] . Rapid growth of cities and it's exceeding from managerial bases and abilities have hugely challenged them in providing well improved services. Researches shows information in the field of city management. The development of electronic city, systematization of a framework for the architecture of electronic city is one of the most basic and most fundamental actions that consists of basic components such as transforming from current situation to a desired one that are general polices and lines of electronic city development [4].

One of the main steps of e-city growth is integration which makes establishing an intelligent and efficient relationship with users in order to answering a variety of questions, problems and their needs possible. For the integration of information systems in e-city, several ways have been proposed. Service oriented architecture (SOA) and distributed object oriented computing technologies are the most important approaches used in this regard [1].

In this research work we have studied different organizations service delivery system and their services to provide integrated services on a certain place in Debre Markos city. To do so we have developed a service oriented architecture having layers and studying the challenges of information systems by concentrating on the benefits of service oriented architecture and, we have proposed a new service oriented architecture for Debre Markos city service integration and interoperability system to overcome the challenges take place and to provide accurate and efficient services to consumers on the need basis.

1.2. Basic Concepts of Service Oriented Architecture

1.2.1. Features of Service Oriented Architecture

A service-oriented architecture (SOA) is an architectural pattern in computer software design in which application components provide services to other components via a communication channels typically over the network. The principles of service-orientation are independent of any vendor, product or technology [5]. One of the concepts used in service orientation system is service reusability.

A service is an autonomous unit of functionality, like retrieving an online transactions for example, retrieving bank statement [6]. Based on this definition, a service is a discretely accessible operation. However, in the Web Services Definition Language (WSDL), a service is an interface definition that may contain several distinct services or operations. And elsewhere, the term service is used for a component that is captured behind an interface.

Services can be combined to provide the functionality of a large software application [7]. SOA makes it easier for software components on computers connected over a network to work together. Every computer can run any number of services, and each service is built in a way that ensures that the service can exchange information with any other service in the network without human intervention and without the need to make changes to the given program itself.

1.2.2. Benefits of Service Oriented Architecture

Agility, flexibility, and business automation have become essential for enterprises to survive [8]. SOA is a strong candidate paradigm for the realization of the agility, flexibility, and automation of the business processes that span large distributed systems. It is an approach that can support systems to maintain scalability and flexibility while growing. Companies that need customizable solutions or use IT for competitive value, companies seeking to leverage IT capabilities for

business advantage, these are companies that should care about SOA. The SOA supports the realization of these strategic goals that enables business and IT to collaborate in order to achieve the following objectives:

- To provide greater flexibility in planned applications
- To provide faster time to value from IT
- To enable for developing modernized strategic applications
- To lower the lifetime cost of applications or infrastructure
- Targets service reuse as a goal to adapt products or capabilities to the market faster.

There are lots of benefits of SOA. We can classify these benefits in to two fundamental groups, technical and business benefits.

Technical Benefits: SOA provides basic technical benefits like, providing efficient development process, facilitating the concept of software reuse, improved interoperability and independence of development technology/platform.

Business Benefits: SOA provides some basic business benefits such as providing business agility, evolutional approaches and adequate business infrastructure.

The following figure 1-1 shows SOA at technical and business perspective.

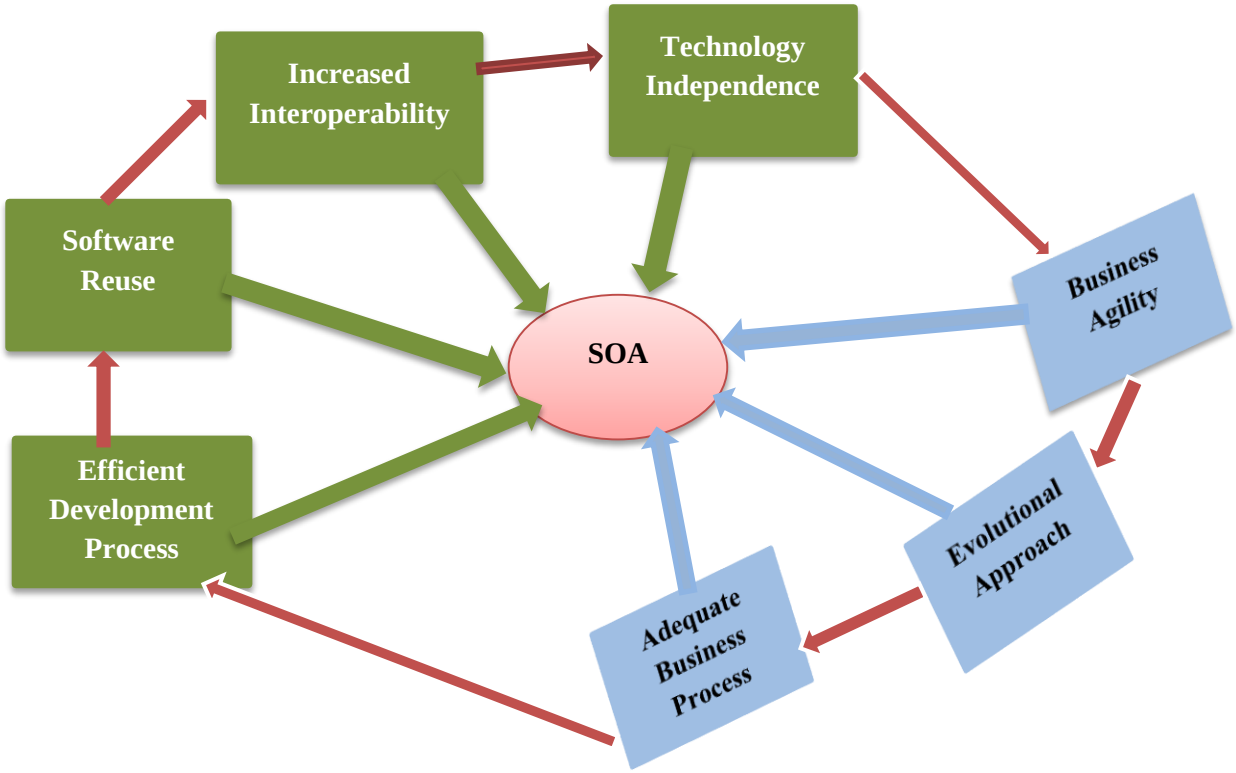

Figure 1- 1 SOA benefits at the technical and business dimensions

1.3.Motivation

Developing an effective service integrity provides a well improved and organized services. Such services need to be managed by commonly agreed business objectives and standards that are managed by a group consisting of business and service integrator agents. Service integration is the integration of discrete IT service elements into a coherent set of end to end services.

In Debre Markos city there are many organizations delivering their services manually using papers. As we gathered the data in different organizations found in the city for example Debre markos city administration mayor office they are providing services to consumers using different techniques like, TV Park, pamphlets, newspapers. Debre Markos city economic and finance office also they are accessing and delivering organizational services manually using papers. Such a system consumes time, resources and money.

This motivates us to develop automated architecture which integrates these different services found in each organization to deliver services on a certain common place. This reduces time for an employee and the consumers.

To design such a system service oriented architecture is a suitable architecture which provides many distinct concepts and methodologies that needs to be defined and explained in order to understand the SOA offerings in an accurate way and build a proficient architecture that satisfy the needs of the consumers. This enables the system to provide accurate and efficient flow of information between a variety systems.

1.4.Statement of the Problem

The need to integrate heterogeneous data sources is a challenge that is faced in almost any software engineering discipline. In Debre Markos city there are organizations providing services to consumers using manual methods of information disseminations. A much more dynamic, adaptive and agile approach is needed to support the rapidly changing requirements of different consumers' in Debre Markos city. Within different enterprises, current solutions like, data warehousing are no longer suitable for the real-time and large enterprises. These conditions have led enterprises to strongly agree to implement Service Oriented Architecture (SOA) to address the agile and dynamic nature of their customers. SOA enables efficient application integration. Probably more and more systems are developed using a model driven approach in order to handle with the increasingly changing requirements and intended use of these systems. Furthermore, the recent trend towards the acceptance of the need for collaboration referred as "The Wisdom of Crowds" [9] and

”Collective Intelligence” have led us to believe new approaches should be discovered to address the increased need for service integration.

It is evident that more research work is needed to explore these new techniques and trends in order to address the increased and improved service integrity requirements. The lack of a scalable methodology and platform that allows the integration of a very large amount of diverse data sources by a very large number of stakeholders made us question the efficiency and effectiveness of these approaches for real world situation.

The lack of support for collaboration and growth in existing systems has motivated us to propose a new approach and methodology for service integration.

Our main aim is dealing with the basic layers of the city architecture and problems of service integration in Debre Markos city and focusing on the benefits of service oriented architecture. Therefore it is important to develop a new service oriented architecture to provide the public to involve in urban affairs and improved delivery of city services and overcoming the challenges of service integrity. Another challenge that motivates us to design an architecture in addition to service integrity, service interoperability among different organizations work together another important issue that has to be addressed.

Finally this study work should answer the research questions like:

- The absence of proper architecture for a city is a challenge for the integrity and interoperability of a variety of services found in the city. If so, how can we develop an architecture for a city using service oriented architecture reliable service integrity and delivery using publish-subscribe messaging model.

1.5. Objectives of the Study

1.5.1. General Objective

The general objective of this research work is designing service oriented architecture for integrating different services and develop a prototype for the architecture to provide and test efficient, accurate and reliable services delivery in Debre Markos city.

1.5.2. Specific Objective

To achieve the general objective of the study, the following specific objectives are identified:

- To review literatures on existing service integration architectures and techniques with respect to service oriented architecture.

- To design a new service oriented architecture which facilitates efficient service delivery system in a city.
- To identify and recommend future research directions for further investigation on the benefits and limitations of services integration and service oriented architecture in a city or urban areas.

1.6. Scope and Limitation of the Study

The main concern of the study is to examine the current architecture of a city to support business organizations to provide services in efficient and effective ways to serve their customers properly. The scope of the study covers Debre Markos city governmental and non-governmental organizations and to compare with existing service integration approach in the city. The research work is limited to providing integrated service delivery system in Debre Markos city basically considering urban areas which have access to computers and internet.

1.7. Research Methodology

To achieve the solutions to the problems which are listed out in the research work the following research methodologies have been used.

1.7.1. Interview

We have conducted a random interview questions with Debre Markos city administration and communications affairs officials, Debre Markos city economic and finance office officials and citizens and collected important information about current strategies for service delivery system they are using as well as current governmental and non-governmental organizations interlinkage and their customer's access to their resources and services possibility.

1.7.2. Literature Review

We have reviewed related journals, books and sources from the web and examined whether or not related works have been done before and if there are, we used them to support our idea. Reviewing different literatures enables us to have better understanding and knowledge about the study work, to identify the gaps found and to provide well improved and advanced solutions to the problem identified under this thesis.

1.7.3. Document Analysis

Sample documents and forms of information exchange and management system between organizations and to the consumers is identified to check whether a similarity or a difference is

available between them so, that we can determine what kind of information integration and delivery system is required in the organization.

This research has been developed mainly in the following research design method.

- Data gathered from different organization from debre markos city to assess service integrity and interoperability approaches used between these organizations.
- Having the above information about the ways ho these organizations are interacting and providing different services to customers and identified the gaps and proposed a suitable service oriented architecture.
- Then we developed a prototype for this architecture to test the reliability and the performance of service integrity architecture using distributed topics architecture with durable subscribers.
- Finally, evaluation of the prototype has been done.

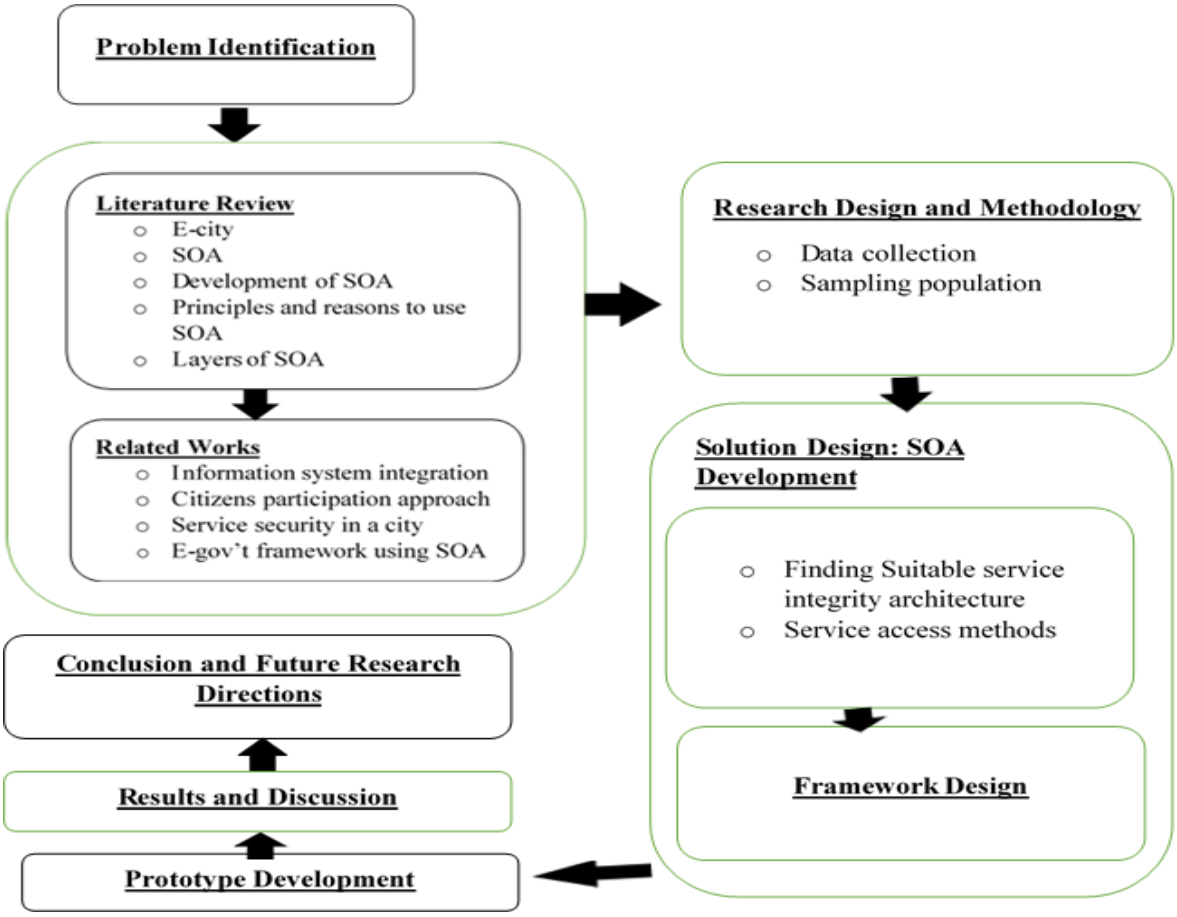

Figure 1- 2 Research methodology procedure

1.8. Significance of the Study

Service integration and efficient delivery of government services from one organization to another and from government to citizens is an important issue for the development of the city as well as the country as a whole. Service integration enables consumers to access and use the right information on time with a cost effective means by the comfort of their homes. Developing this architecture reduces cost of development for the organization, reduces access time of both the organizations and the consumers. Therefore it is important to create strong relationship between the organizations and the people and tourists coming to the city.

1.9. Structure of the Thesis

The remainder of this thesis work is organized as follows:

Chapter two introduce the background knowledge about service oriented architecture focusing on evolution of SOA, SOA entities, characteristics of SOA, features and benefits of SOA, types of services in SOA, layers of SOA and related works associated to our problem. Chapter three presents methodology and solution design and development. Chapter four discusses about prototype development and demonstrations and the last chapter presents contribution, conclusions and future research directions of the study**.**

CHAPTER TWO

2. LITERATURE REVIEW AND RELATED WORKS

2.1. Overview

Finding background information on the area around the problem is a key procedure in doing any research work. Accordingly, we have carried out a study of the important scientific background knowledge and related literatures about service oriented architecture and service integration approaches. This chapter gives a brief description of the background knowledge about service oriented architecture basically focusing the research domains like city, components and benefits of connected city services, SOA, developments of SOA, entities of SOA, characteristics and types of services in SOA.

2.2. Basic Concepts of a City

A city is defined as a large and permanent human settlement. Cities generally have complex systems for sanitation, utilities, land usage, housing, and transportation. The concentration of development greatly facilitates interaction between people and businesses, benefiting both parties in the process, but it is also challenging to manage urban growth [10].

2.2.1. Components of a City

There are many common components which exist in all cities. We can define these components according to their type and category [11].

Buildings: buildings consists of representing, symbolizing religious and absolute power, symbolizing civic and regal power, symbolizing military/police power, symbolizing legal power, symbolizing financial and administrative, symbolizing knowledge and technology of society.

Areas: professionals or administrators accommodation**,** storage goods-warehouse/distribution**,** workers accommodation**,** markets**,** industry.

Systems: public service administration system**,** legal justice system**,** military or police system**,** roads or transport system**,** food/produce/market/distribution system**,** water & sanitation system**,** power system**,** public standard of living**,** health system**,** education system**,** public welfare & entertainment system and others.

2.2.2. The Role of ICT for E-Services

Information and Communication Technologies (ICTs) can play a vital role in sustainable human development and poverty reduction [12]. ICT is a powerful facilitator of development goals because of the way in which it improves communication and the exchange of knowledge and information necessary for the development processes. ICT is pervasive and which impacts the full range of human activity and will become one of the main enablers in the pursuit of poverty mitigation and wealth creation in developed and developing countries. As accelerator, driver, multiplier and innovator, ICTs are powerful tools in the huge scaling up and interlink-age of development and outcomes [13].

ICTs also play a critical role in facilitating the flow of information and knowledge between government and citizens and changing the way in which governments and citizens interact. According to the United Nations Development Program (UNDP) the challenge for all countries is to create and develop a system of governance that promotes, supports and sustains human development [14]. Governments in many parts of the world have made huge ICT investments aimed at improving governance processes.

Electronic governance is considered as the ICT provided route to achieving good governance since it integrates people, processes, information, and technology in the service of governance initiatives. The estimated benefits of such public sector restructurings have been recognized as an increase in the efficiency of government operations, strengthening democracy, enhancing transparency, and providing better services to citizens and businesses [12].

Generally, e-government is the use of ICT to promote more efficient and effective government, facilitate more accessible government services, allow greater public access to information, and make government more responsible to citizens. E-government has emerged beyond electronic service delivery and is part of the ongoing restructuring and transformation of government enabling participatory governance and partnerships to improve efficiency and effectiveness [15].

E-government is about transforming government services to be more citizen-centered. E-government successes require changing how government works, how it deals with information, how officials view their works and interact with the public. E-government is also classified into different sectorial areas such as e-health, e-education, e-banking, e-insurance and SMME (Small and Medium Enterprises).

Achieving e-government success also requires active partnerships between government, citizens and the private sector. The e-government process needs continuous input and comment from the customers the public, businesses and officials who use e-government services. Their ideas and feedback are important to make e-government to function properly. If e-government is implemented properly it opens an opportunity for public participation.

2.2.3. Benefits of Cities with Advanced Technology

Cities provide the potential of a better, more sustainable lifestyle to their consumers in the following ways [16].

Efficient Service Usage

Technology solutions across the fields of energy, transport, and waste led to direct economic and environmental benefits. Examples include customized energy consumption through smart metering, micro-grids, and dynamic pricing are some of the technologies facilitating efficient usage of energy.

Transparent and Connected Public Services

Efficiency in public and citizen services was determined by e-governance initiatives in the early days. Over time, the focus has shifted to interconnected delivery of government services, which enables for an improved citizen involvement in every social and economic aspects. For example integrated record management system allows for holistic health services across locations. Ready access to services and those services performance standards for all individuals serves the cause of transparency and equity in city governance.

Increased Safety and Security

Smart cities tend to deploy integrated public safety and security solutions (remote monitoring, smart cameras, pattern recognition), resulting in safe and secure settings for their citizens.

A Better Lifestyle

These next-generation cities combine and extend their objectives to create stronger links between government, education, and industry by using ICT. In order to attract and retain talent, these cities also provide world class services in terms of schools, housing, landscaping, and retail and entertainment channels.

Created Managed Service Capabilities

As discussed before, governments and developers demand immediate solutions and are increasingly exploring innovative business models that allow them to transfer the capital expenditure investments and risk to vendors.

Establish Partnerships

A wide range of decision makers and influencers involved in building a city, ICT users need to create suitable partnerships, both on the supply side to develop capabilities and on the demand side to gain access and the right to use the services.

Adopt a Different Operating Model

ICT providers are used to short turnaround times for product sales or service provisioning with immediate realization of revenues. Money flows in a city build-out, however, are significantly different from those in the traditional sales model to which ICT players are adapted. These require a long sales cycle that involves significant pushing of governments to influence up-front design of the city's ICT main plan.

2.3. Introduction to Service Oriented Architecture

SOA describes an open, agile, extensible, federated, compose-able architecture consists of autonomous, QoS (Quality of Service) capable, vendor diverse, interoperable, discoverable, and potentially reusable services, implemented as Web services [8].

Service Oriented Architecture (SOA) can be defined as a paradigm or an architectural style. It is not a framework or product that can be bought and used. It is more of a thinking approach which helps in designing interoperable software architectures according to enterprise business requests. SOA aims to combine architectural style and methodologies to achieve interoperability among heterogeneous and homogenous systems by creating reusable services. It concentrates on the problem domain more than the underlying technology used to implement it.

Service Oriented Architecture (SOA) provides a design framework to integrate disparate applications so that their functionality can be exposed and made available as services on a network. Moreover, SOA divides monolithic applications into different service sets, making functionality modular to implement. There are different implementation models and techniques for SOA but the most common is through the use of standardized Web Services which make interoperability easy

to implement among disparate systems. The following figure 2-1 shows the basic components of SOA and the relationship between them.

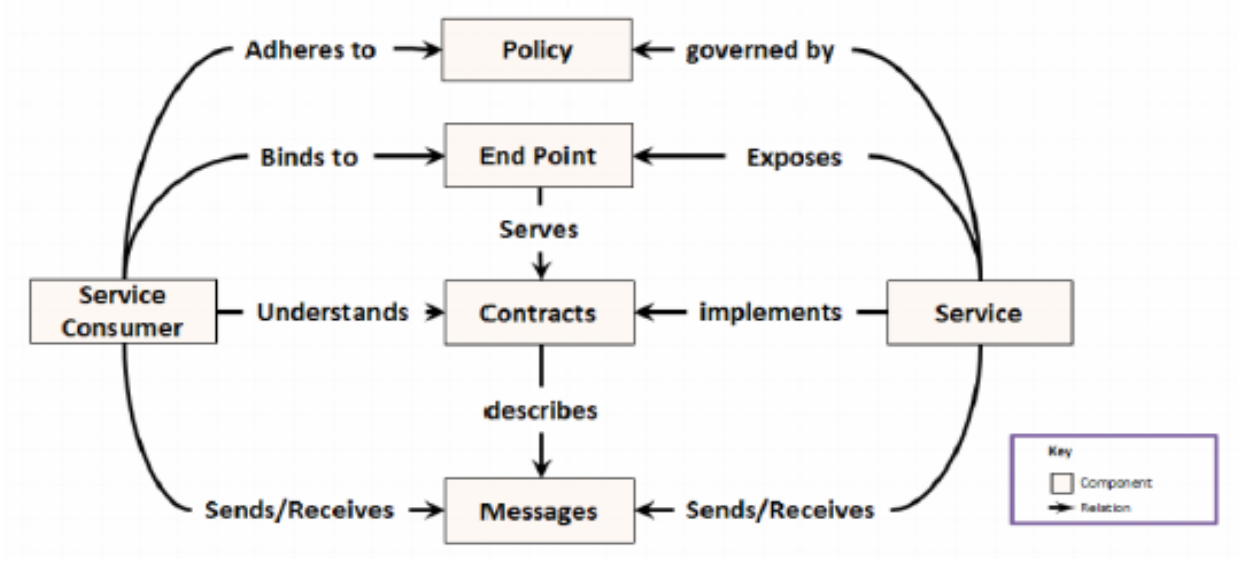

Figure 2- 1 Components of SOA and their relationship

2.4.Development of SOA

Alexander P. is a former analyst defined the term SOA while he was teaching middleware architecture. He came up with SOA term because the traditional client/server architecture had lost its touch and people started referring to distributed computing as involving a desktop PC which ran the frontend logic and another computer running and maintaining backend operations. To avoid confusion among meanings of client/server applications and SOA which originated as misconceptions, Pasik concerned about "server orientation", as he recommended developers to design SOA based business applications [17].

The concept of service-orientation describes the development of different IT patterns and technologies. Moreover, SOA itself is in the phase of development and this makes it difficult to precisely define SOA. The following are the technologies from which SOA has been developed [18].

2.4.1. Object Orientation

Object orientation is a concept of describing how distributed solutions would be constructed. Object orientation had its own set of rules and principles, which helped to ensure consistency among applications running on different environments. The main principle which came out of

object orientation was objects being the units of solution logic, and a wide-range of solutions would communicate and relate with each other using these objects.

Service-orientation can be considered as an advanced form of object orientation, and the principles and patterns of object-oriented analysis and design have been inherited by service-orientation, for its communication and implementation. For instance, the principles like, service reusability, service abstraction, and service composability are basic concepts of object-orientation which are being used in service oriented model [19].

2.4.2. Enterprise Application Integration

When enterprises started to interact with each other more frequently, integration became an important concept. Different integration techniques were developed, and the first technique was point-to-point integration technique. This technique functioned for small integrations, but as soon as the enterprise grew, it became too complex to manage. The familiar problems occurred with this technique were lack of scalability, extensibility and no support for interoperability.

Enterprise Application Integration (EAI) platforms were introduced to improve the problems associated with point-to-point integration technique.

This paradigm used adapter, broker component and orchestration engines to achieve application integration and business processing. EAI had its own set of limitations. The biggest problems with EAI were high scalability cost. Due to these problems, enterprises started looking forward to other solutions which were vendor diverse and had low scalability cost. With the emergence of web services as an open framework, enterprises moved towards service orientation which gave them much more flexibility and control over their integration architecture and platform.

A lot of innovations were developed in the time of development of EAI which are being used within SOA. One such innovation is the broker component, which allows services with different schemas to interoperate with each other. Orchestration engine is another such inherited component. These inherited components of EAI map to several service oriented principles, like service abstraction, service statelessness, service loose coupling and service composability [20].

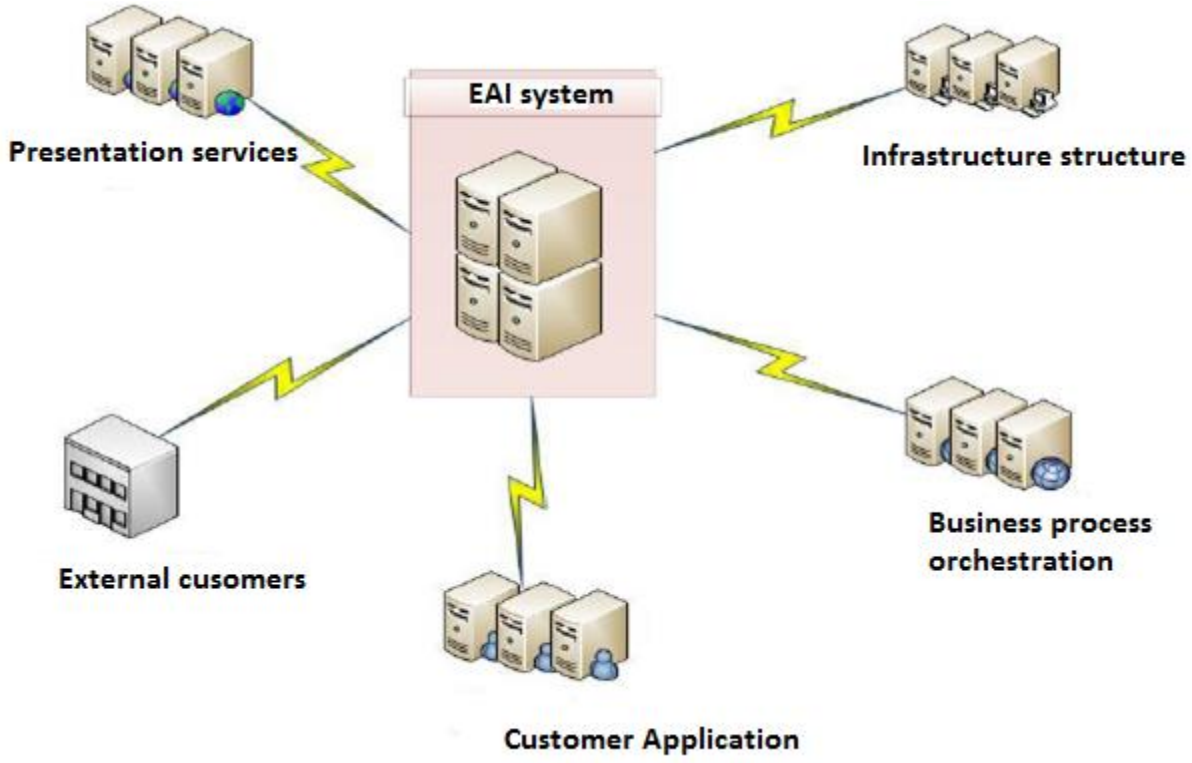

Figure 2- 2 Enterprise Application Integration

2.4.3. Web Services

Web services have become the main implementation technique for SOA. This is because web services are standardized across vendors, which allows easy interoperability among disparate systems. Most of the vendors are now using web services as their main technology for the implementation of SOA solutions.

As SOA is a concept, it has been influenced by different vendors and frameworks, one such framework is the web services framework, which has also influenced SOA in shaping up several of its principles including service abstraction, service loose coupling, and service composability [21].

2.4.4. Business Process Management

Running business processes smoothly and in a correct order is essential for an enterprise. The emphasis of business process management is on restructuring business processes, by making them adaptive to change and improving their efficiency. Business process layer is one of the basic layers within service oriented architecture. It provides the capability of service composition and organization.

One of the major goals of Service oriented architecture is to automate business processes, which are highly adaptive to change. This goal can be achieved by abstracting business logic into one layer, and having services to reuse this logic without having to reconstruct it again. Service logic reusability is a key factor within Service Oriented Architecture and business process management fulfills that functionality.

2.5.Basic Entities of SOA

SOA consists of three basic entities within its architecture: Service Consumer, Service Provider and Service Registry and the interaction among them. The interaction between them is described as: service consumer requests the service provider for a service, the provider searches the registry for the requested service which matches the needs of the consumers, and if the service is available the service information is sent to the consumer [22]. Figure 2-3 shows a model for the interaction among SOA entities.

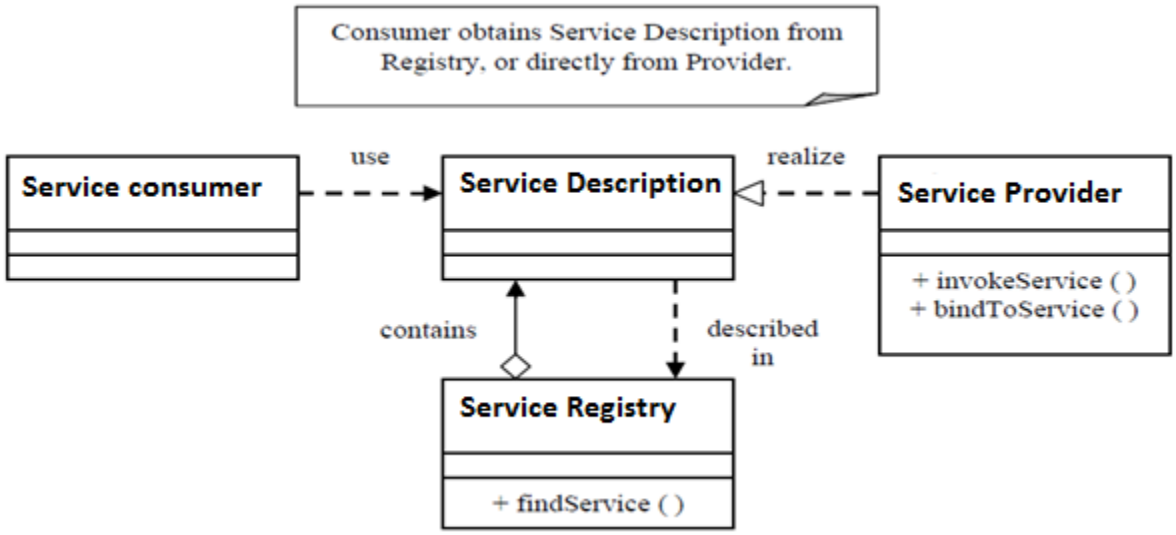

Figure 2- 3 Basic components of service oriented architecture

2.5.1. Service Consumer

Service consumer is an entity which consumes the service functionality provided by a service. It calls different functions within a service for different functionality. The consumer can be a software module, a service or an application. The consumer can either directly invoke the service if its location is known or can access the service location through registry.

2.5.2. Service Provider

Service provider is the core service; it accepts and executes requests which comes from the consumers. It consists of the details about service description and implementation. The service provider can be a mainframe system, a component or any other type of system which executes the consumer's requests.

2.5.3. Service Registry

Service registry is a network directory which contains addresses of all the available services. Its main purpose is to store and publish service contracts of different service providers, and when requested provider provides these contracts to interested service consumers.

2.5.4. Service Contract

A service contract is a specification or description of the way a service consumer would be interacting with a service provider. It holds information about the exchanged message formats, quality of service pre and post conditions which need to be fulfilled before the service can be executed.

2.5.5. Service Proxy

Service proxies are provided by service providers to facilitate service consumers by allowing them to call functions of the service using an API which is exposed via proxy. This API is written in the local language of the consumer. The service proxy can enhance performance by providing caching of remote references and data. This proxy is optional in SOA and services can also be called directly by the consumers.

2.5.6. Service Lease

The service lease is the amount of time for the validity of a service contract. It is approved and managed by the registry. The lease is significant for the services which require maintenance of state information and about the bindings between the consumer and provider. In addition, using service lease increases loose coupling between service provider and consumer.

2.6. Characteristics of SOA

Software architectures have some specific characteristics and principles that needs to be followed in order to fully use their capabilities, [20], [23]. Service oriented architecture has the following characteristics and principles:

2.6.1. Services are Discoverable and Dynamically Bound

SOA provides a way to discover services dynamically at run-time. The service consumer asks the registry for the required service and receives the essential information to execute the service at runtime. Service interfaces are discovered dynamically and messages are also constructed at runtime which leads to no compile time dependency of being bound to the service. The only thing that needs to be updated is the service contracts, if they changed.

2.6.2. Services are Self-Contained and Modular

Services are usually self-contained and modular. A service has a set of functional interfaces which jointly performs business oriented tasks. These interfaces relate to each other to form a module and have enough information to function without being authenticated or being dependent on other software applications or modules.

2.6.3. Services are Interoperable

One of the main features of service oriented architecture is interoperability, which is the ability to communicate with different systems without being dependent on any specific platform or language. Every software module or architecture has some proprietary structure which is restricted to its own domain and is considered tightly coupled. Service based architecture achieves interoperability by supporting protocols and data formats of the service and potential consumers.

2.6.4. Services are Loosely Coupled

Coupling refers to the level of dependencies among software modules. Coupling can be categorized as either loosely coupled or tightly coupled. Loosely coupled modules have well-defined dependencies and are more flexible. Most of the software architectures try to provide modules which are as much loosely coupled as possible. On the other hand, tightly coupled software systems are difficult to organize as they have lots of unknown requirements for each module within the software structure or architecture.

Service oriented architecture emphasis on development of loosely coupled services, meaning there should be minimal dependencies between the consumers and providers. In SOA, loose coupling is accomplished by the use of registry, which contains the binding and contract information of every service. Contracts are the main concern for the service consumer and they need no other specific information to call the service. However, tight coupling remains at the implementation level i.e. at interface definition or binding to a specific protocol.

2.6.5. Services have a Network Addressable Interface

A service should have a network address interface where its interface is published. This is one of the main design principles of service oriented architecture. The consumer should be able to invoke a service across a network in a distributed way; this makes a service reusable and available to different consumers. Services can also be accessed through a local interface without any involvement of network; this is possible when both provider and consumer are on the same machine. This is basically done to improve performance.

2.6.6. Services have Coarse-Grained Interfaces

The concept of granularity with respect to services can be defined in two ways. Firstly, scope of the implementation domain of the service. Secondly, scope of interface implementation for each method within the service. In short, it can be seen as the implementation of interfaces within the system. There are levels of interface granularity, if the interface provides all the business functionality, it is considered as a coarse-grained interface. On the other hand, if the interface implements part of the business functionality, it is called fine-grained interface. A service can have granularity for all its interface methods or for some particular interface methods. A service oriented system by default supports coarse-grained interfaces with different granularity levels for the service interface. Meaning, objects within a service can be fine-grained, but these objects remain within the physical structure of the service as shown in Figure 2-4.

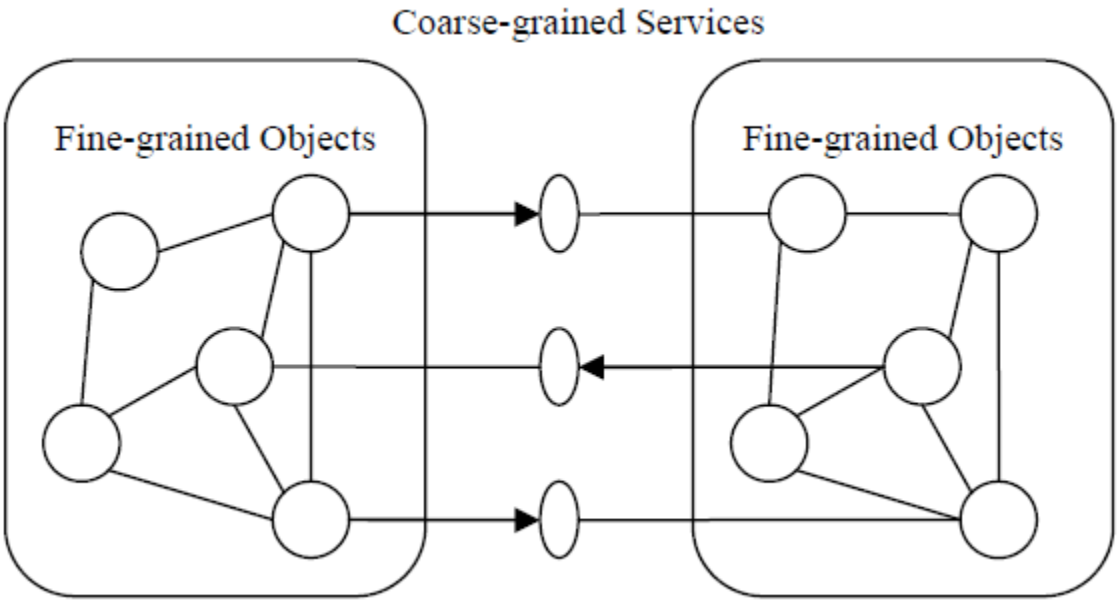

Figure 2- 4 Service granularity

2.6.7. Services are Location Transparent

Location transparency is one of the main characteristic which SOA promotes. Location transparency enables services to move from one location to another without affecting the consumer of the service. Consumer has no knowledge of the location of the service until it looks for it at run-time in the registry. The consumer has dependency only on the service contract, this makes it easier to change service implementation location.

2.6.8. Services can be composed into new Applications

Reusability of services is also one of the key characteristics of SOA. This enables development of new applications using existing services. Composition of services is an effective design approach which concentrates on reusability of service functionality without having any knowledge of its future use.

2.6.9. SOA Supports Self-Healing

A self-healing system is one which has the ability to recover to normal state in case an error occurs. Applications within a SOA based system should possess the ability of self-healing, as within a service based system, services are usually combined to execute business functionality. For a SOA based system to execute efficiently and properly it should always support self-healing properties.

2.7.Reasons to use SOA

SOA helps an enterprise to benefit both in its business and IT area. From the business perspective, SOA makes it possible to develop dynamic applications, which resolve a number of business problems that are essential for the growth of an enterprise [24]. Some of the benefits of SOA concerned with business development are:

2.7.1. Improvement in Business Decisions

SOA helps an enterprise to integrate different business oriented information into one composite business application. This dramatically increases information accuracy and provides decision makers access to a lot of information. Different departmental information can be mapped onto a single web page, enabling enterprises to have a better understanding of their business costs and tradeoffs of decisions being made during daily business operations. Information being readily available also makes it possible for an enterprise to identify and rectify problems quickly.

2.7.2. Improved Employee Productivity

SOA enables employees to be more productive by providing them easy and efficient access to systems and information, which in turn improves business processes. SOA removes the IT platform integration restrictions and limitations, enabling employees to focus more on the important business processes. Moreover, clients can access their required information in any view which could be web, desktop or mobile application; this really enhances productivity.

2.7.3. Easy Integration of Consumer and Supplier Systems

SOA's main benefit is easy integration of different systems and applications. Enterprises can easily merge with each other without any major change in their current running IT systems. Integration with other partner enterprises and restructuring of different processes within the enterprises is easily performed. Customer or partner stratification increases as they get access to dynamic applications and business services, all at one place.

SOA greatly helps in defining supply chain processes for specific business tasks which are independent of their underlying IT architecture, this leads to better arrangement of business processes within enterprise. SOA also contributed in documenting different business processes and models, which helps in optimizing and capturing change within these business processes.

From an IT departments perspective, service oriented architecture offers a simplified framework for the creation and management of integrated systems and applications. It is a technique to map IT processes with business models and their changing needs.

2.7.4. More Productive, Flexible Applications

SOA allows exiting IT systems including legacy systems and applications to integrate without any need for adding any custom code or updating these systems to newer versions. This increases the productivity and success of the enterprise business. SOA also allows enterprises to develop dynamic and composite applications which provide information from different systems irrespective of their underlying environment or platform and programming language. Furthermore, as services are not dependent on their underlying platform, solutions can be designed with greater flexibility.

2.7.5. Faster, Cost Effective Application Development

Services are standardized which allows developers to make them as reusable services that can be arranged into composite services and applications to fulfill business change requests. This reduces the cost of solution, development and testing time and speeds up the process of implementing new requirements within the business system. Moreover, the use of one standardized development model increases efficiency and simplifies application development, maintenance and testing.

2.7.6. More Manageable and Secure Application

SOA solutions provide a standardized infrastructure for developing applications which are secure, can be monitored and have services with standardized contracts. SOA makes it very easy to add and map service capabilities to new and existing business processes. Services can be added independently within a service oriented solution, which helps in removing the need of developing a new application. Existing IT applications just need to include new services to start providing the new functionality. This saves cost for the enterprise, of developing new applications. Moreover, SOA provides a strong authentication and authorization model for all services, and as services are loosely coupled, the overall security is increased.

2.8.Types of Services in SOA

There are several types of services in SOA, which are divided into two major categories: Business Services and Infrastructure Services.

Business Services

Business services are services that perform specific business functions and are required for the successful completion of a business process. They might also be called application services since they are used to develop composite services and business applications that automate business processes. For example, a retail enterprise might have an inventory service, customer management service, and shipping service in its repository of business services.

Business services can be further divided into entity services, capability services, activity services, and process services. Entity services are the data centric components of the enterprise system and are responsible for exposing the information stored in backend databases. They are the action centric components that implement business capabilities. Whereas capability services are coarse-grained and provide generic business capability, activity services are more fine-grained and provide specific business capability.

Infrastructure Services

Infrastructure services are part of a centrally managed infrastructure component such as an Enterprise Service Bus (ESB). Infrastructure services provide the technical functionality necessary for the implementation of business processes in SOA, but do not directly add business value. Examples of infrastructure services include Software as a service (SAAS) integration services, authentication services, event logging services, and exception handling services.

Service Oriented Development

Services are the advancement of components in which multiple component interfaces form into a single interface to perform a specific function. A service is an abstract resource with the capability of performing a task [25]. Services have the potential reflection of business functions as well as technical task definitions.

Service based development advances component based development in a way that services involve development of distributed, cooperating components realized in different programming languages, but also service orientation supports the features such as platform independence, dynamic discovery of services and improved level of reusable application modules.

Service orientation is as well evolved service-based application development in which services is the natural realization of Internet networking and World Wide Web technologies. These technologies change the application structure to allow more dynamic and open software

construction and solve interoperability problems by applying standard developments that is acceptable from many software organizations and vendors.

Services are designed and developed to support the following characteristics:

- Each service defines a specific business function and can match to real-life activities
- A service may have various procedures and operations
- Services interact with other services and system components in a loosely coupled, message oriented environment to accomplish business goals
- Services has clearly defined interfaces and can be used by many different other services and applications
- Services do not need to be in a distributed environment

The following figure 2-8 shows service based development in the context of components and objects.

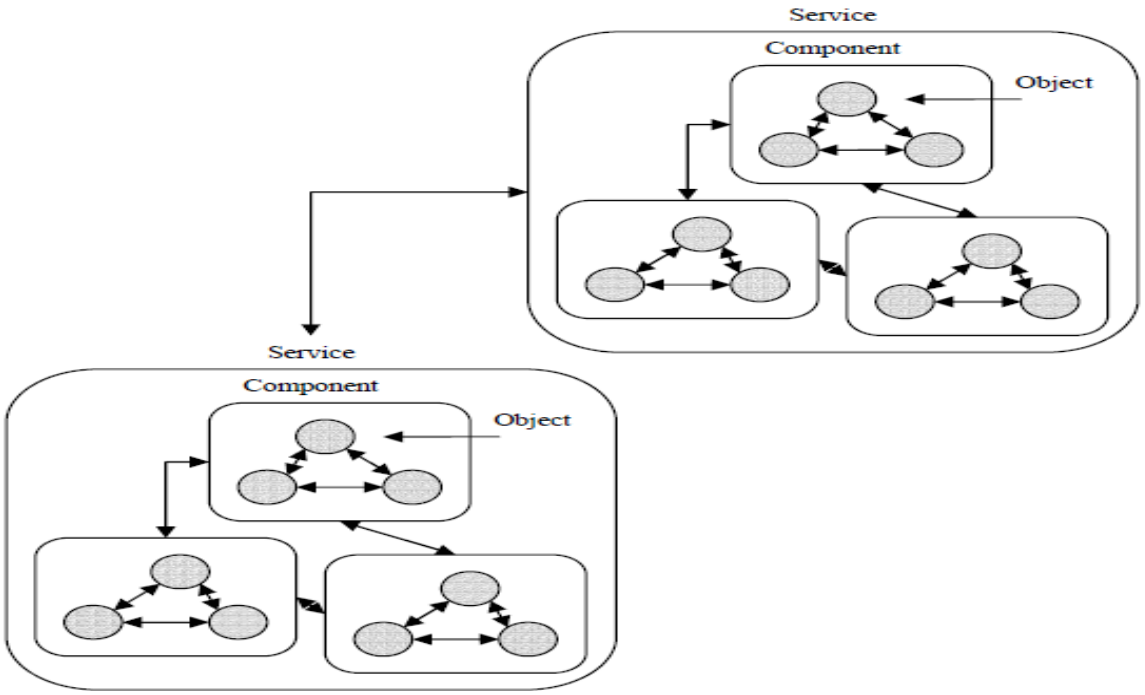

Figure 2- 5 Service based development

Software development has passed through certain phases and architectural models until to enable the development of platform independent, reusable services. Each architectural model provides a better means of dealing with software difficulties than the previous approaches in order to accomplish better software structures that fit real world requirements.

The early architecture of software is based on structured design, which has rigid rules and limited support to enable robust and sophisticated application development. Object oriented technologies result in flexible software development that supports encapsulation of business logic through more coarser-grained functions and classes; however, the tangible benefits of robust application development are gained through the progression of components and services. The following table 1-1 describes the characteristics and features of software architectural models.

Table 1- 1 Comparison of architectural development models

Structured Development	**Object Oriented Development**	**Component Based Development**	**Service Based Development**
Very fine system structuring	Small grain system structuring	Medium grain system structuring	Coarse-grain system structuring
Low reusability	Low reusability	Medium reusability	High reusability
Tight coupling	Tight coupling	Loose coupling	Loose coupling
Have compile time dependencies	Have compile time dependencies	Have compile time dependencies	Have only run time dependencies
Intra-application communication scope	Building blocks are individual classes	Building blocks consist of several classes (components)	Building blocks consist of components
	Encapsulation, Inheritance, Polymorphism	Interactivity, connectivity, and exchangeability of components	Published interface Definition
	Functionality is described by class declarations	Functionality is described by interface declarations	Functionality is described by network addressable component interface declaration
	Dynamic but large number of connected object		Dynamically discoverable distributed services
			Inter-enterprise communication scope

Basic Layers of SOA

SOA has the following five basic abstract layers: enterprise layer, process layer, service layer, component layer and object layer.

Object Layer: is the bottom layer and consists of legacy systems, including custom-built applications and databases that contain business functionalities. These legacy enterprise objects can be leveraged to build composite services.

Component Layer: The second layer next to the object layer consisting of enterprise components. These components are responsible for realizing the functionality of services.

Service Layer: is the middle layer which is where exposed services (both individual and composite services) carrying out business functions reside. This layer acts as a bridge between the lower-level layers (the object layer and component layer) and the higher-level layers (the process layer and enterprise layer). Enterprise components can be exposed as services in this layer, making reuse a real possibility.

Process Layer: The fourth layer where the services exposed in the service layer are combined through service organization or service composition to create a single application that realizes and automates a business process.

Enterprise Layer: Also known as presentation layer is the last layer which is the end-user's point of access to the composite enterprise application, typically over the Internet. As the layered abstraction above makes clear, services play a vital role in SOA. For enterprises facing integration challenges and looking to increase business agility in meeting customer demands and to cut IT costs, SOA doesn't have to be difficult to understand. As long as business leaders understand what services are (and their different types) and how each layer contributes to building new applications, SOA becomes less foreign as a concept and instead becomes a mission-critical IT goal waiting to be realized.

2.9.Related Works

Several research works has been done related to service oriented architecture design and service integration approaches of different information systems in a city like, [28], [29], [30] and [31]. These research works vary in different aspects like, the software architecture they follow, the issues they addressed, the application domain and the platforms they utilize different services in different organizations. In this chapter, four of the related works are presented with an emphasis on their

primary goals, the issues addressed, the gaps we have identified and the comparison of each work to our work.

2.9.1. A Novel SOA for Integration of Information Systems in an E- City

According to Sajjadhashemi, S.Y [26], service oriented architecture is used for integration of information system and challenges in e-cities so that most of the activities are performed through internet and electronic systems for the purpose of sustainable, trustworthy and confidential access of offices, urban areas and other required information of citizens.

Accordingly, they specified various information systems to achieve these goals which needs to cooperate and interact to each other to provide reliable and efficient services to consumers. Creating interactions among these information systems were the main challenges faced while producing the suggested architecture in an e-city. They classified the main challenges of information systems for the development of e-city into three key groups of problems [33].

- The lack of interaction and integration of all organizational information systems to call and use of common software components independently from the operating system, platform, technology, and sides communication protocols.
- The difference of terminology and viewpoints of information technology experts and business experts brings the lack of a common understanding of information systems which needs to be productive.
- Lack the ability of information technology to adapt with the business changes speed with respect to needs of organizations as to constant changing of processes and services.

To overcome these challenge they have proposed an architecture consisting of five layers, these are user oriented service layer, application oriented service layer, security service layer, data service layer and information infrastructure layer.

The main target of this paper is to study the layers of the e-city and challenges regarding to integration and interaction aspect and designed an architecture having the above layers but doesn't deal with how these layers integrated with each other.

Figure 2-6 An architecture for the integration of information systems in an e-city [26]

2.9.2. A Novel SOA for E-City with the Increase of Citizens Participation Approach

According to Hashemi, S. [27] proposed an architecture for the increase of citizen's participation approach in which one of the key issues in the success of e-city. They proposed service oriented architecture because service oriented architecture is an appropriate choice for increasing citizens participation in urban affairs and for creating interoperability between different organizations. Finally they proposed a six layer service oriented architecture including: user oriented service layer, strategies for collaboration layer, security service layer, application oriented service layer, data service layer, and information infrastructure service layer.

This paper deals with similar challenges of e-city layers specified on the first paper [32] and focused on the new layer called citizens participation strategy to increase active participation of the consumers of a city but it needs to define interaction methodology and security aspects in detail.

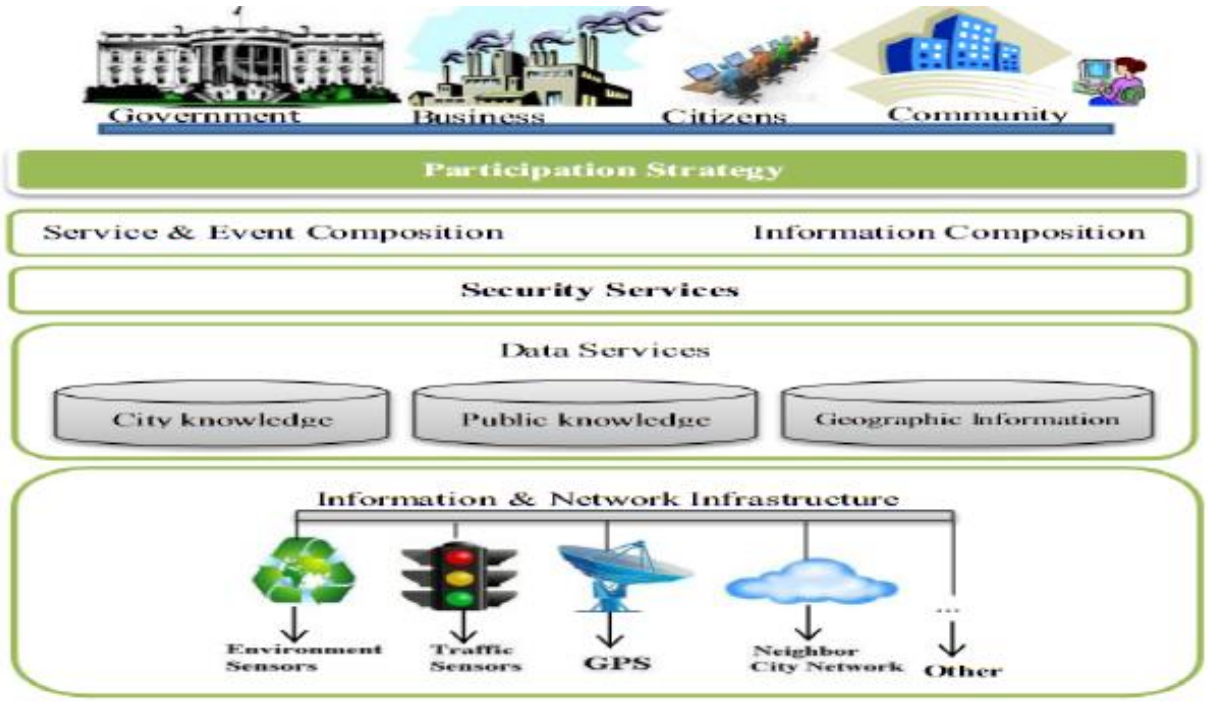

Figure 2-7 The architecture for the increase of citizen's participation in an e-city [27]

2.9.3. A Novel SOA to Secure Services in E-City

Sajjad Hashemil S.Y [28] proposed Service oriented architecture to secure services in e-city having seven layers for delivering secured information service including: stakeholder's layer, participation strategies layer, channel of communication layer, electronic data server layer and information and network infrastructure layer. This architecture provides independent services during calls by various stakeholders, reuse or service and service composition.

This architecture provides trust on people because it hides internal complexities of a system using security mechanisms and provides independent services for users but doesn't include interaction models and implementation details about the proposed system.

Figure 2-8 An architecture for secure services in e-city [28]

2.9.4. A Framework for an E-Government Based on SOA for Jordan

According to [29] e-government is growing faster that needs to full attention from the government and demands collaboration and facilitation between private sectors and Non-Government Organizations (NGOs). For the successful achievement of the e-government applications, governments have to provide services to citizens, businesses and government agencies. In Jordan, e-government applications are limited to an informative objective; they basically provide information. Additionally, it is found that the traditional peer-to-peer integration of applications will result in a tightly coupled system that decreases the agility and expansion of the e-government

system. In this paper they proposed a new integration method based on the web service of the Service Oriented Architecture for the various e-government systems.

They developed a stage model framework for e-government interoperability based on Service Oriented Architecture. The architecture is being evaluated using case study in the context of implementing environmental license web service in the Jordanian ministry of environment. The framework consists of layers client layer, presentation layer, application layer, data layer.

CHAPTER THREE

3. RESEARCH DESIGN AND METHODOLOGY

3.1. Overview

Identifying a suitable research design and methodology related to our study is the key concept considered before performing any research work. In this research we have stated suitable research design and associated methodologies. We have also discussed the study area in which we collected the data for our research, nature of the study, research type and approaches, sampling techniques, data analysis and final results of the study.

3.2. Study Area

The study is conducted mainly in Debre Markos city focusing on governmental and private organizations service integration and service delivery system.

3.3. Nature of the Study

The nature of the study is focused on describing and finding the possible requirements which are basic for designing service oriented based architecture for service integration and interoperability for efficient and accurate delivery of information for service consumers in the city. The nature of the study is formulative and targets to analyze organizations service delivery and integration system and how they process data from one organization to another as well as to citizens for serving the people and organizations to improve the life of the people and for the development of the city in general.

3.4. Research Design Methods

The emphasis of the research work is to identify feelings, thoughts and ideas about the current trends of service integration systems and conduct a case study scenario. For our research formulative research approach is suitable.

For our research qualitative approaches are used to gather the required primary and secondary data. We conducted open ended interview questions to Debre Markos city administration staff officials, employees and examined different documents used in different organizations for serving their customers.

3.5. Sampling population

Sampling technique are techniques used to clearly define the target population. A small, but carefully chosen sample can be used to represent the population. The sample reflects the characteristics of the population from which it is drawn.

3.5.1. Source Population

The sampling units of the study is Debre Markos city administration service integration and service delivery systems. Here we have collected necessary data from Debre Markos city administration mayor office and communication affairs office. We found that 26 governmental and 19 non-governmental organizations in Debre Markos city. In addition to this based on the 2008 census the population of Debre Markos city is 107,433, from this 50036 are males and 57397 are females. Managing these large number of organizations and the people is not an easy task, therefore some kind of architecture and governing mechanism is required. So, we proposed a service oriented architecture for integrating different services of a city to simplify the tasks of managing the city and the delivery of information for different organizations and to citizens.

3.5.2. Target Population

The target population of the study is higher officials of Debre Markos city administration office, employees of different organizations of Debre Markos city, citizens and tourists and invited guests.

3.5.3. Sample Size

A total of 6 respondents were involved in the actual study of the research work, 2 of them are from Debre Markos city administration mayor office, 2 of them are from communication affairs office, and 2 of them are from economic and finance office.

3.6. Validation

The city administration is responsible for administering different departments and service consumers found in the city but there are many challenges such as management overhead and service wastage. To reduce these challenges we need to have a mechanism to handle these large population based on their needs. This mechanism is designing an architecture deployed at the center and users access it possibly. Therefore, we designed an service oriented architecture using distributed topics messaging approach.

CHAPTER FOUR

4. SOLUTION DESIGN: SERVICE ORIENTED ARCHITECTURE DEVELOPMENT

4.1. Overview

As we discussed in the previous chapters the main target for this research work is to design a solution model for service integration in for Debre Markos city using services oriented architecture. In this chapter, e designed a service oriented architecture by adopting the existing service oriented architecture layers and the new layer called service integrity layer for service integration and interoperability between disparate organizations in the city. Therefore in this chapter, we discussed about service integrity,

4.2. Service Integrity

In short, the term integrated services is defined as to examples of joined-up social services, for the mutual benefits of service users or providers. A more detailed definition can be taken from the health literature: "integration is a coherent set of methods and models on the funding, administrative, organizational, service delivery and clinical levels designed to create connectivity, alignment and collaboration within and between different sectors" [30].

Services can be integrated either horizontally or vertically. Vertical integration refers to "bringing together of different levels in the care hierarchy" [31]. In health care, for instance, this could mean integrating the hospital, clinical and community based health services to ensure the level of care. Our focus here is horizontal integration, which brings previously separated services together, professions and organizations across different sectors to better serve service users with complex needs [32].

Integrated services can be delivered in many forms, depending on the extent of interaction, and the scope of support. Integration of services can happen via cooperation or communication among service providers, collaboration among professionals across different sectors, the physical or virtual collocation of complementary services, or a mixture of these.

4.2.1. Collocation, Collaboration and Cooperation

The concept of service integration covers collocation, collaboration, and cooperation. Each term defines integration at different stages.

Collocation: refers to getting all agencies in one location such as: legal services, educational services, health services, housing services, social services or city management services as indicated in figure 3-1. Getting all the services in one location can reduce the travel and time costs associated with access for service users [33]. Collocation also makes for easier accessibility between agencies that can help to promote collaboration among groups of service providers and professionals.

Collaboration: provides a higher level of integration than collocation. It refers to organizations working together through information sharing and training, and creating a network of organizations to improve service user capability. Collaboration is a fundamental process for minimizing the gaps in services for service users. By sharing knowledge, agencies and professionals can improve the information distribution process to other services offered by the center [34]. The more knowledge professionals have about the different services, the improved needs based recommendations are available to service users.

Cooperation: The highest level of integration is achieved through cooperation. Cooperation is defined as professionals communicating and working together. For example, within a certain organization small number of professionals from different organizations are discussing about the service delivery issue on a service user's state [35]. Effective cooperation, through good communication is a principal issue for improving service user's outcomes. When professionals work well together, costs can be reduced, services will not be duplicated, identification and response to service users' needs can occur more quickly**.**

4.2.2. Suitable Model for Service Integrity

To structure the discussion of integrated services in this research work, it is necessary to define a model for integrated services and associated terminology. In developing a model of service integration, and importantly to assess it, it is also necessary to acknowledge the type of goals achieved by integrating services. In this work, the goal for service integration for users is to enhance quality of support and quality of life, service user satisfaction and system efficiency for service users with complex, long term problems cutting across multiple services, providers and settings [36].

Figure 4-1 demonstrates essential concept of integrated services delivery system. The model represents a one-stop Centre where service users can access the service they need in one place. This can be physical, or virtual, and can be explicitly defined as integrated or not. In some cases this will include services that the user is entitled for, or in need of, but previously has been uninformed of the possibilities of access.

Figure 4- 1 A basic model for service integrity [36]

The integrated services delivery model consists of a number of distinct elements, which needs the following explanations:

The purpose of the model is to deliver output in the form of service user outcomes. The term service user is used to define the consumers of public services provided. It is used refer an individual or groups of consumers.

In this integrated model, a service user has entered the integrated setting following the assessment of their needs from employees in City Management System (CMS), or via one of the service agencies.

The various service agencies are all single service providers. As we can see from the above figure the five example organizations included in the model are: legal services, educational services,

housing services, health care services and social service organizations. The term providers is used in this research work to define individual service providers, or a group of integrated service providers. The number and specialization of the organizations may change depending on the situations and the needs of service user requests.

The outside circle represents the connections among the service providers. The circle can be used to represent the various forms of integration between service providers. Arrows within the integrated model, going back and forth to city management system, represents the potential for service providers to share information through and with the CMS.

At the Centre of the model, the city management system can take different forms, from an organization or individual, with an overview role of the interventions for the service user, such as a case worker assigned to represent the service user in the delivery process. The CMS, organization or individual, can actively manage the integration whether it is simple cooperation, collaboration, collocated service, or a mix of all of these. The CMS may also take the form of an information exchange system, designed as an information sharing core at the Centre of a service network.

4.2.3. Interoperability and Integrity Challenges in SOA

As the level of e-government maturity increases the need for interoperability will increase [23].The following challenges will take place in service integrity and interoperability in SOA. There are three types of integrity challenges.

Technical Interoperability

Technical interoperability describes the technical issues of computer systems. It includes issues on platforms and frameworks. Frameworks are complex and provide conceptual differences to working approaches; e.g. understanding and relying on classes in an object-oriented system. In addition, at times frameworks are duplicative and contradicting with multiple levels.

Organizational Interoperability

Organizational interoperability is concerned with organizational processes and cooperation of organizations. The processes are not flexible enough and adaptive to be integrated and be interoperable. Here the requirements of decentralized agencies have to meet the central needs on coordination. The higher level managers play a vital role. Leadership and strategic direction of management are cited as the most important factors for corporate adoption of web technology.

Semantic Interoperability

Interoperability or integration efforts are about making information from one system syntactically and semantically accessible to another system. Syntax problems involve format and structure the system. Semantics is important technical issue that is almost invisible outside technical circles. Such differences normally make it more difficult to make systems work together. The differences can be minimized if systems are designed using approved data formats. Semantics is related to the understanding and integrity of the information.

Semantic interoperability includes both the data interpretation, by means of XML schemas, and the knowledge representation and exploitation. Semantic interoperability is an enterprise capability derived from the application of special technologies that relate, interpret, and classify the implicit meanings of digital content, which in turn drive business process, enterprise knowledge, business rules and software application interoperability.

4.3. Getting Access to the Services

The services are deployed in the provider's data store and accessed through the web by all registered users. Any client/customer needing the service is going to register and access the service based on his/her interest and needs. For accessing the service provided by the provider, the consumer can use a variety of devices. For example PCs, laptops, smartphones and so on because the proposed service oriented system supports platform independence, thus it becomes cost effective and efficient. Figure 4-2 shows the interaction of the consumers through the cloud service.

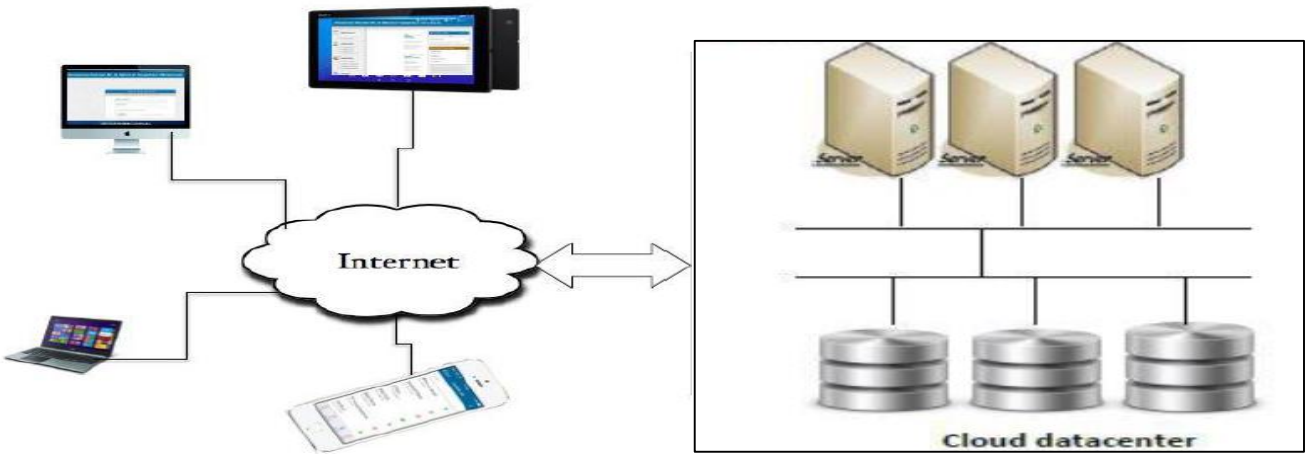

Figure 4- 2 High level overview of accessing the services to the cloud

4.4. A Proposed Service Oriented Architecture

As we discussed in the previous chapters Debre Markos city consists of different organizations with different information sources for delivering to the users but, these organizations are using manual approaches of data handling and dissemination techniques which is not reliable and efficient method. Therefore, it needs to develop a new system which handles this large amount of data called service oriented based service integrity and interoperability architecture.

In this section we discussed the service oriented architecture for the proposed system under study and brief description of the components of the architecture. Therefore we did adapting the architecture to ease the communication between the layers and focused on the new layer service integrity layer of the architecture. In this research work we identified the layers of the city architecture and designed an architecture by providing important layers basically service integrity layer which is not stated in the above papers. This layer deals with techniques of integrity by linking different services provided in the city. This layer facilitates the interaction between different services by providing messaging, transformations, routing, and protocol conversion activities. Therefore our main intention is focused on service integration layer related to the issue messaging. Figure 4-3 illustrates the proposed service oriented architecture for Debre Markos city.

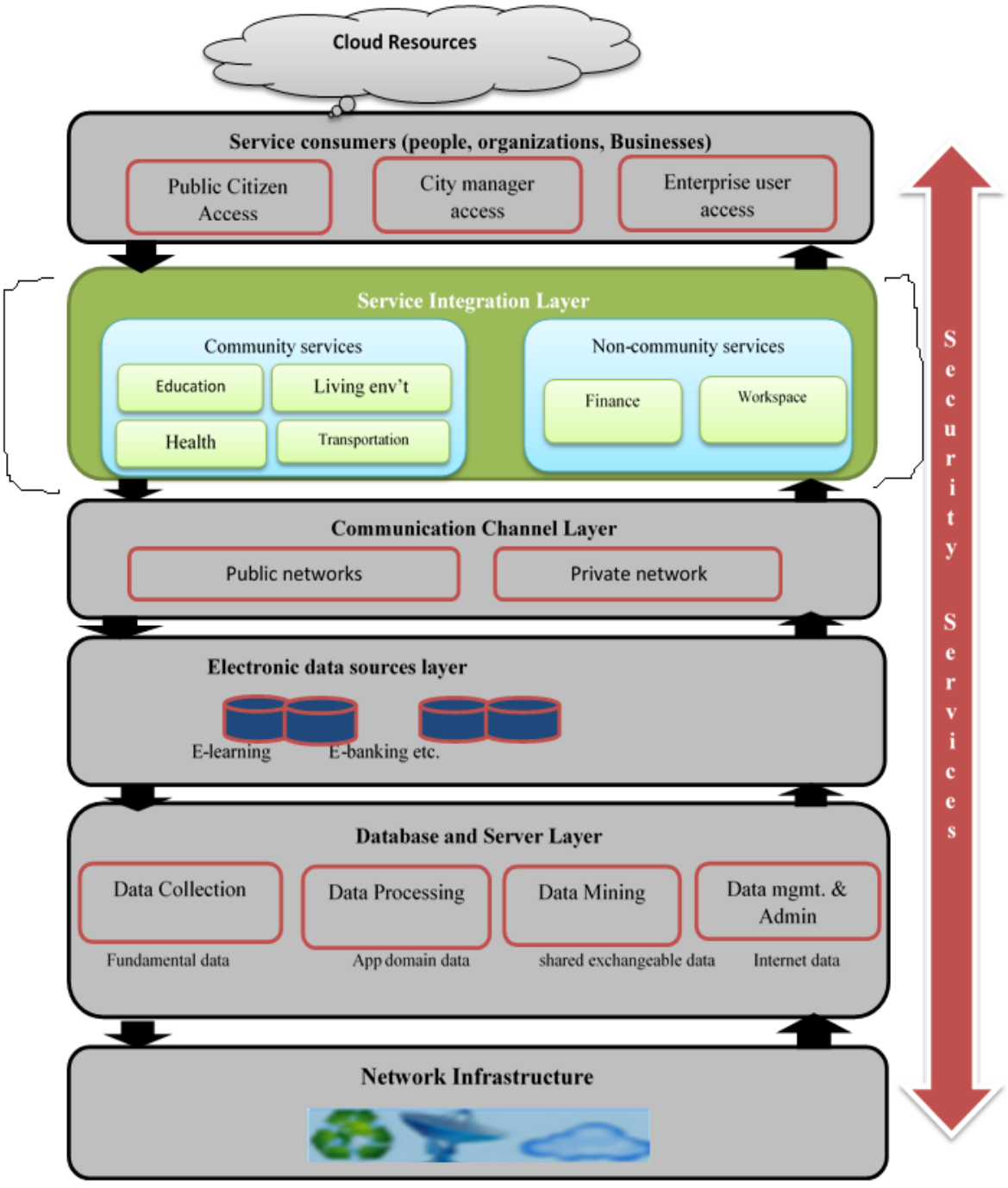

Figure 4- 3 The proposed service oriented architecture for service integrity in a city [26]

4.4.1. Basic Components of the Architecture

Service consumer's layer: This layer is used as an interface between the users and the system which enables channel-independent access to services and business processes provided by various applications and platforms. It provides the capabilities required to deliver IT functionalities and data to end users and used as an entry point for consumers. It also provides the ability to integrate services from within the other layers, and the ability to transform and personalize information.
This layer helps users to easily identify and use of the services, enables them to access data electronically, and enables the user to choose the best option provided by the city, to facilitate an opportunity for active participation in urban affairs and it reduces time and the cost of accessing the services provided by the city. The services can be grouped into three groups: public citizen access, city manager access, and enterprise user access.

Service Integration layer: This layer is a key facilitator for the architecture because it provides the potential to facilitate transformation, routing, and protocol conversion to transport service requests which comes from the service requester to the correct service provider. Thus, it supports the capabilities required for enabling activities such as routing, protocol support and conversion, messaging or interaction mechanism, support for diverse platforms, adapters, service interaction, service enablement, service virtualization, service messaging, message processing, and transformation. From these capabilities of service integration layer we focused on the service messaging system to test the reliability and efficiency of the system.
These services can be grouped in to two groups, community services and non-community services.

- **Community Services:** These are services that are provided directly to citizens and visitors of the city, like education, healthcare, living environments, transportation and so on.
- **Non-community Services**: These are services which consists of other services retrieved through the use of ICT and which will have an impact on sustainability of the city. For example the finance services include for instance the development of smart business environment, while a workplace service relates to improve work efficiency, and significant information services to encourage people's choices.

Communication Channel Layer: This layer is used as a connecting line or a pathway for accessing services provided by the city that includes a variety of browsers and telecommunication equipment's connected to the internet. It provides proper platform and infrastructure support, best

quality services and shortest possible time range, enable proper coverage and extension, qualitative benefits like, security, reliability, and it also supports multiple communication protocols.

Electronic Data Sources Layer: This layer offers a collection of electronic data sources such as e-learning, e-banking, e-governance, e-finance, e-insurance and integration of necessary infrastructure for application and data transfer to ultimate goal in the network. This layer is responsible for providing the possibility of reusing the data sources, accessing the data by different channels, dynamic and flexible features of the data, combining and offering new services, ability to easily identify and discovery of services and the ability to hide the internal implementation and coding from the users.

Data base and Servers Layer: This layer is used to organize and manage the flow of information and services of data systems, city data and allocate them to the requested organization.

Network Infrastructure Integration Layer: This layer is used as a bridge to the upper and lower layers. It provides city information services by receiving raw data from the environment and delivers to the upper layer to be processed.

Security Services Layer: This layer keeps services to be secure and reliable. The service users or applicants depending on the type of service and concurrent with the requested service will request required level security as a separate service. For example some applications require strong security than the others, for instance accounting transactions require higher level of security than managerial services because accounting transactions consists of sensitive data that needs strong security. Therefore security plays a vital role in this architecture.

CHAPTER FIVE

5. PROTOTYPE IMPLEMENTATION AND VALIDATION

5.1. Overview

This chapter explains a brief description of implementation technologies and prototype design for the proposed service oriented architecture using some messaging protocols to increase the performance and scalability, reduced resource consumption and flexibility of the suggested architecture.

5.1.1. Description of Implementation Technologies

Several technologies can be used for achieving the issues of service integrity and interoperability problems. Some of the basic technologies are discussed below:

Service Oriented Architecture

SOA is an architectural pattern whose goal is to achieve loose coupling among interacting software agents. A service is a piece of work performed by a service provider to achieve desired outcomes for a service consumer. SOA is mainly focused on the issues of interoperability. It provides additional systems and covering the existing systems to function together.

Service oriented environment is concerned on the following key principles:

- SOA is not just architecture of services seen from a technology perspective, but the set of rules like, policies, practices, and frameworks by which we can ensure the right services are provided and consumed.
- SOA is an important architectural style to implement two or more processes for a service provider and service consumer.
- Instead of discovering individual services by developers business process bus is used as a starting point for guiding them to an ordered set.
- Organizations are turning their dimensions to SOA based on Web Service technologies to make existing applications, components, and data available for reuse and to simplify the consumption of reusable resources [37].

Web Services

The W3C Web Services Architecture Working Group defined, web services supports direct interactions with other software agents using XML messages exchange through Internet protocols.

Web services need semantic-driven descriptions for discovery, negotiation and composition. Web Services have entered the research issues of many research communities and are being proposed as the means for remote interoperable access of components and software systems [37].

Jini Network Technology (JNT) is the first service oriented technology [23]. Jini is a lightweight environment for dynamically discovering and using services on a network. Its main objective is to enable devices such as printers to dynamically connect to the network and register their available services.

Therefore, the need to use services as a software development unit led to development of more set of technologies for implementing service oriented architecture. However, there are challenges of SOA related with service interoperability, integration complexities and conveying to industry principles, influenced to service based application development.

J2EE

The Java 2 Platform Enterprise Edition (J2EE) is a java technology which supports the design, development, and deployment of component based distributed applications [24].

The J2EE specification proposes several services and each of these services has different functionalities which can be used in building service oriented application development. In this case Clients access these services by provided APIs.

5.1.2. Java Message Service

Messaging systems provide reliable transmission of information between different applications across a variety of heterogeneous computer networks and systems. These systems have the characteristics of allowing interacting programs to run at different times and with hiding network complexities. Java Message Service (JMS) is a Java enabled development interface which supports standardized asynchronous message interaction and offers the capability of simulating synchronous request/response communication mode.

JMS provides Java applications with a standard and consistent interface to the messaging services of a MOM (Message Oriented Middleware) provider or a messaging server [38]. JMS enabled applications provide integrating abilities for legacy systems and distributed applications in heterogeneous environments. The architecture of the JMS system allows queuing of messages with guaranteed, timeliness delivery and ensuring offline applications to process messages later when they are capable of receiving them.

A message is a collection of data in which one application needs to send to another application, usually on another machine [25]. Messaging systems provide a method to accumulate incoming messages on a queue, which lets the receiver to process messages and receive the response back at another time. The difficulty in this technique is to have a reliable transportation of messages between sender and receiver.

JMS has a set of rules to manage message communication for reliable and stable message transportation with supporting features of message persistence, message acknowledgement, and administration of the message consumer's performance, which may disconnect and reconnect to JMS provider.

The following figure 5-1 shows message exchange between message Sender, provider and consumer.

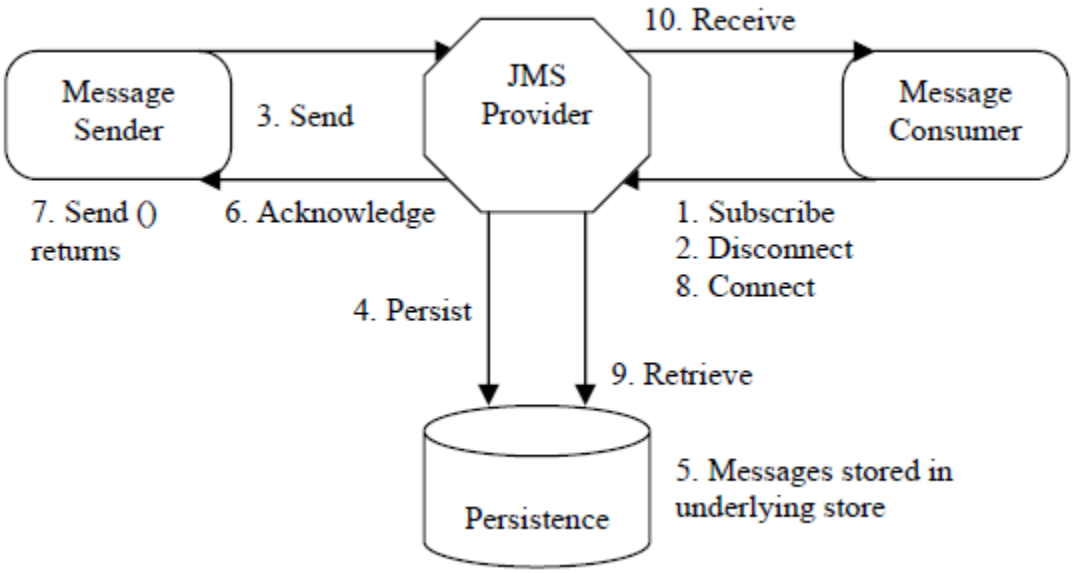

Figure 5- 1 Conceptual Model for JMS

JMS supports different types of messages, such as byte message, object message, and text message. The JMS specification [10] defines a programming interface which leaves the implementation details to message provider, which is located at each J2EE compliant application server. Messaging in general achieves loose coupling as the sender and consumer interact through the messaging transport which requires separate transactions. The provider guarantees the delivering of messages to its destination, however the processing of the message is independent from sender.

5.1.2.1. Messaging Models

There are two basic types of messaging models: point-to-point and publish-subscribe messaging models.

5.1.2.1.1. Point-To-Point

In this type of JMS model, the message is produced by the sender, which is then placed on a queue, addressed to a single targeted receiver. The consumer reads the incoming messages from queue for processing.

- Each message has only one consumer
- A sender and a receiver of a message have no timing dependencies. That means, the receiver can fetch the message whether or not it was running when the client sent the message.
- The receiver sends back an acknowledgement for the successful processing of a message.

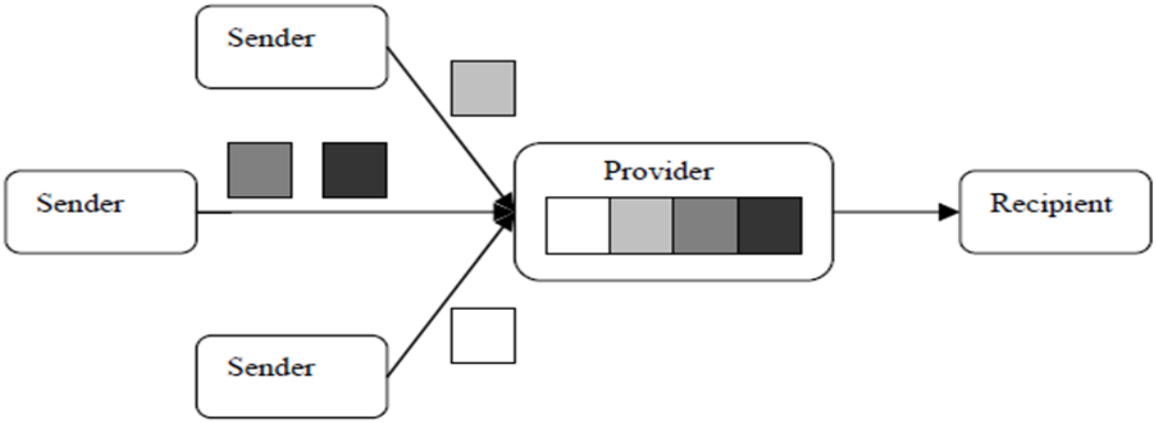

Figure 5- 2 Point to point messaging model

5.1.2.1.2. Publish-Subscribe

In this type of model, the senders publish messages, which are then put on a topic. Multiple recipients can subscribe to the topic and deliver a copy of the particular message.

- Each message may have multiple consumers.
- Publishers and subscribers have a timing dependency.

A client that subscribes to a topic can consume only messages published after the client has created a subscription, and the subscriber must continue to be active in order for it to consume messages.

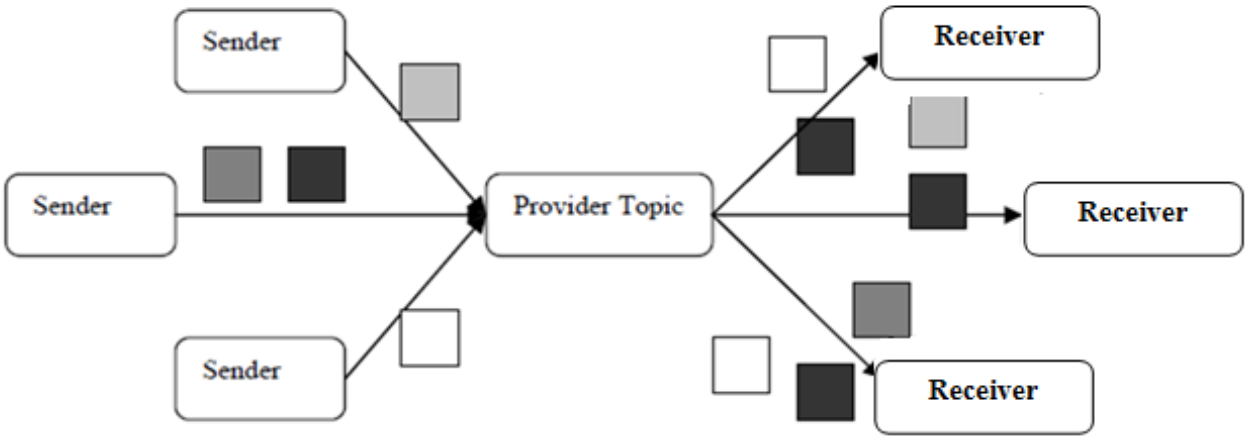

Figure 5- 3 publish-subscribe messaging model

From the abovementioned models we choose the publish/subscribe type of model, because the publish/subscribe model provides access to multiple consumers from a single provider and leverages the performance of communication. Therefore, the Publish-subscribe messaging model is suitable for our architecture used for distribution of messages for heterogeneous receivers.

5.2. Case Study Scenario

Consider Debre Markos city administration has a single JMS message distributed to multiple consumers having guarantee of messages to be received by the consumers even if any of the consumers getting down during message delivery.

As we discussed before an application may have multiple connections/consumers participating to receive the message, during this time it should maintain high availability and scalability to ensure production rates are higher than consumption rates.

Every consumer application should receive exactly one copy of each message regardless of the number of connections/consumers involved in the messaging system.

To resolve this issue we can use two types of messaging approaches:

- Distributed topics or replicated architecture and,
- Partitioned topics architecture

Distributed topics messaging approach allows the delivery of the same message to multiple consumers or connections along the queue. Every consumer receives a single message unless some

kind of problem is occurred. If the problem takes place the message is going to be placed on the queue and can be resent later based on the delivery configuration set. If any application is offline the message will remain on the queue. Which is one of the limitations of distributed topics. Some of the limitations of this approach are:

- There is no guarantee of message delivery, meaning the offline clients may not get the message.
- It reduces performance and consumption of space for duplicating the same message to all the consumers.
- If the number of applications increases, it requires more queues, which needs code changes.

The second approach is best suitable for our scenario, because it resolves the limitations of distributed topics by using the concept of durable subscriptions.

The combination of distributed topics with durable subscriptions allows the same message is to be delivered to many consumers with lower latency, lower overhead (no extra copies of the message) and less management costs (configuration simplicity).

After the development of WebLogic server version 10.3.4 the combined effects of partitioned distributed topics and shared subscriptions enable more than one applications to use the same client ID and receive one copy of the message.

5.3.Developing Suitable Messaging Approach

Publish-subscribe messaging is an improved technique of messaging model having unique features provided by oracle WebLogic server for Java Message Service. These features goes beyond JMS specification and provide more robustness to java messaging system, brings parallelism to the Publish-subscribe messaging and providing the competing consumers pattern found in point-to-point (queues) model.

Competing Consumer's Pattern: Is defined as multiple consumers are created to receive messages from a single point. When the message is released by this single channel one of the consumers could receive that message. The messaging system determines which consumer receives the message but the consumers compete each other to be the receiver. Once the consumer receives the message, it acknowledges to the rest of its applications to help process message. This is done in point-to-point messaging pattern but in case of publish-subscribe pattern more copies of each message are distributed.

Oracle WebLogic server also provides performance, scalability and high availability to the Publish-subscribe messaging pattern.

The original distributed topic architecture was known as replicated which is defined as: "All physical members receive each message sent. If the message arrives at one of the physical topic members, a copy of that message is delivered to the other members of that uniform distributed topic. A subscription to any one particular member will get a copy of any message sent to the uniform distributed topic logical name or to any particular uniform distributed topic member". By the above definition distributed topic architecture, the message is replicated to all the members of the distributed topic so that all members receive the messages sent. There is no way to prevent messages sent to all members and only a single connection is permitted to use the same client ID on the entire cluster and the subscription name can't be shared. Distributed topics/replicated architectures have their own limitations like:

- The need for parallelism
- Less availability and
- Ability to control how many copies of the same message are received by the application.

After the release of WebLogic server 10.3.4 new features of publish-subscribe are formulated. Some of the basic features are as follows:

Partitioned Topics: partitioned topics give the same scalability present in distributed queues to distributed topics, allowing messages to be spread among multiple managed servers. This increases scalability, because it allows messages to be processed in parallel and also allows them to be spread among all available servers. Another important feature made available by partitioned topics is the ability to control how many copies of each message an application may receive, independent of how many connections and subscriptions it may have on the same topic stream.

No Forwarding: Partitioned topics do not forward messages to the other physical members. Oracle WebLogic Server documentation defines partitioned topics as the following:

The physical member receiving the message is the only member of the uniform distributed topic that is aware of the message. When a message is published to the logical name of a Partitioned uniform distributed topic, it will only arrive on one particular physical topic member. Once a message arrives on a physical topic member, the message is not forwarded to the rest of the members of the uniform distributed destination, and subscribers on other physical topic members do not get a copy of that message.

Therefore it prevents producing large amount of sent message copies distributed among all the members i.e. the limitations of distributed topics architecture.

Sharing the same client ID and subscription name among different subscribers: Multiple subscribers can share the same client ID and subscription name, guaranteeing high availability and scalability for message consumption and processing (parallel consumption) for durable and nondurable subscriptions.

Ability to control how many messages an application will receive: having the above mentioned techniques, we can control the number of message copies of a single message an application will receive. It can be one message copy per instance of the application or just one copy per application regardless of how many connections of the application are available. Message-driven beans (MDBs) have been updated and new activation configuration properties have been produced to control such actions.

The above new feature provide the following benefits:

- **Architectural Flexibility**: we can easily add more subscribers and ability of cluster expansion if message production supersedes message consumption without any code change
- **Configuration Simplicity**: MDBs can be easily configured to support different models of subscription.
- **Less resource Usage**: Not forwarding messages to all the cluster members, as happens when using the replicated model, ensures fewer resources.
- **Performance and Scalability**: With the new options, you can consume messages in parallel much the same as you could do with queues. You can also expand your cluster the way you want and subscribers will adapt to the new architecture.
- **High Availability**: Multiple physical subscribers can share the same subscription, there's guaranteed support for high availability, for the distributed topic itself and for the topic subscribers.

5.4.Prototype Implementation

To demonstrate the implementation of the above case scenario, suppose we have two subscribers (MDBs) and one client application with durable subscription approach to receive the one copy of each message published on the topic. We have two distributed topics having the three above listed

subscribers placed on a cluster and two JMS server's server A and server B as shown below in figure 5-4.

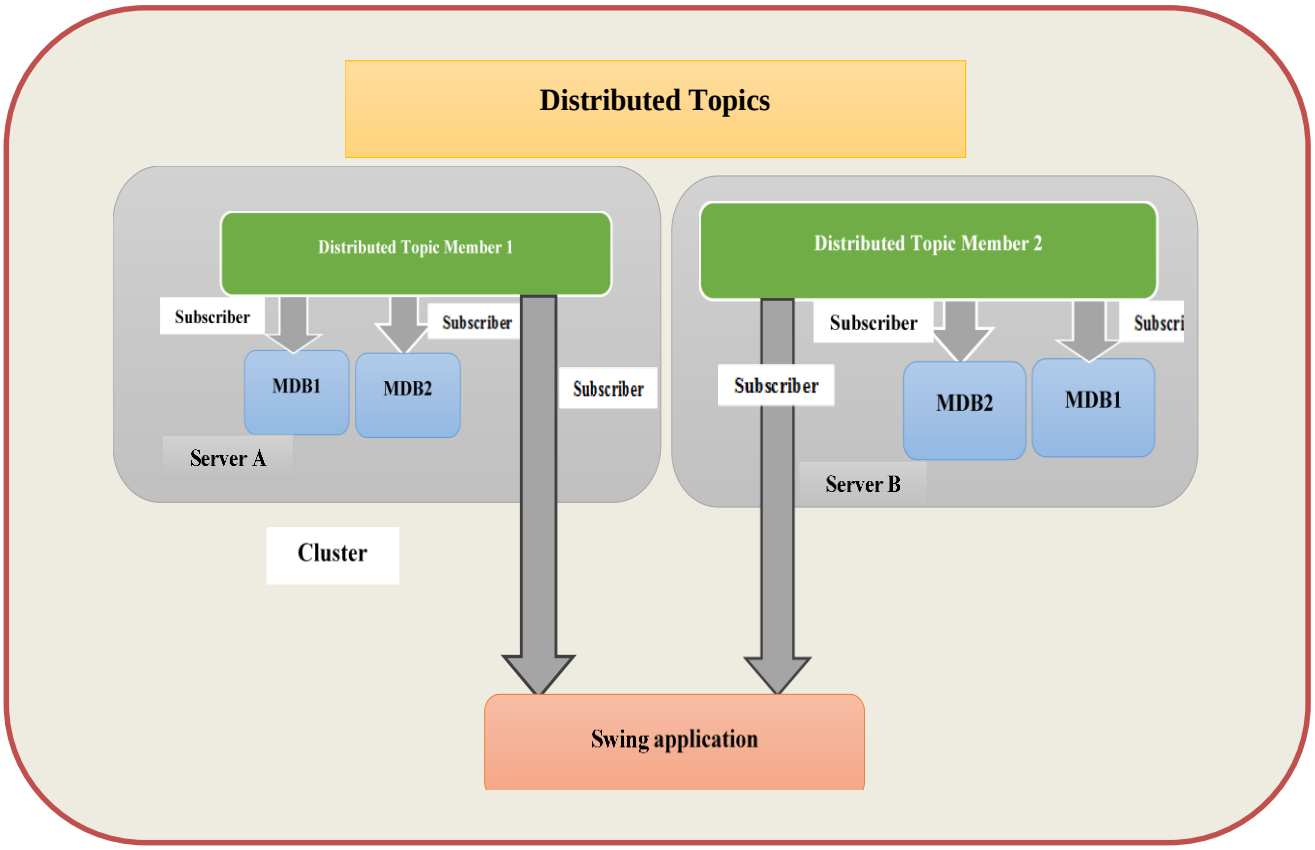

Figure 5- 4 Conceptual model for the distributed topics with durable subscriber's case study scenario

After designing the scenario diagrammatically the next task is selecting a suitable implementation tool for checking the performance and reliability of the proposed architecture. The best suitable implementation software is oracle WebLogic server 12.1.2(12c) integrated on java eclipse Kepler using partitioned distributed topics approach. This tool consists of built-in JMS plugins to implement message processing. Before performing the implementation it is better to define some of the terms used in different JMS system resource configurations.

JMS System Module Configuration on WebLogic Server

JMS system module is a JMS resource that displays the general information about the JMS module and its resources. It also enables the users to configure new resources and for accessing existing resources in the administration console page.

Table 5- 1 Basic components of the JMS system module

Name	Description
Name	The name of JMS system module. For example in our case myTopic.
Descriptor File Name	The name of the JMS module descriptor. i.e. JMS/myTopic-jms.xml
Type	The type of the JMS system resource, such as a Queue, Topic, or Connection Factory. Example ShareableConnectionFactory
JNDI Name	The JNDI name of system resource i.e. JMS. myTopic
Sub-Deployment	A method by which system resource are grouped and targeted to a server resource (like JMS server, server instance, or cluster sub-deployment) i.e. ClusterSD
Targets	The place where the system resource is targeted i.e. AdminServer or JMSServer-1 or JMSServer-2

The first task is configuring the JMS system module by focusing on partitioned distributed topics in oracle WebLogic server 12c. The following figure 5-5 shows the system module for creating distributed topics.

Figure 5- 5 creating JMS system module

When we are implementing partitioned distributed topics in WebLogic server using administration console we must select forwarding policy as partitioned. The following figure 5-6 shows the forwarding policy as partitioned.

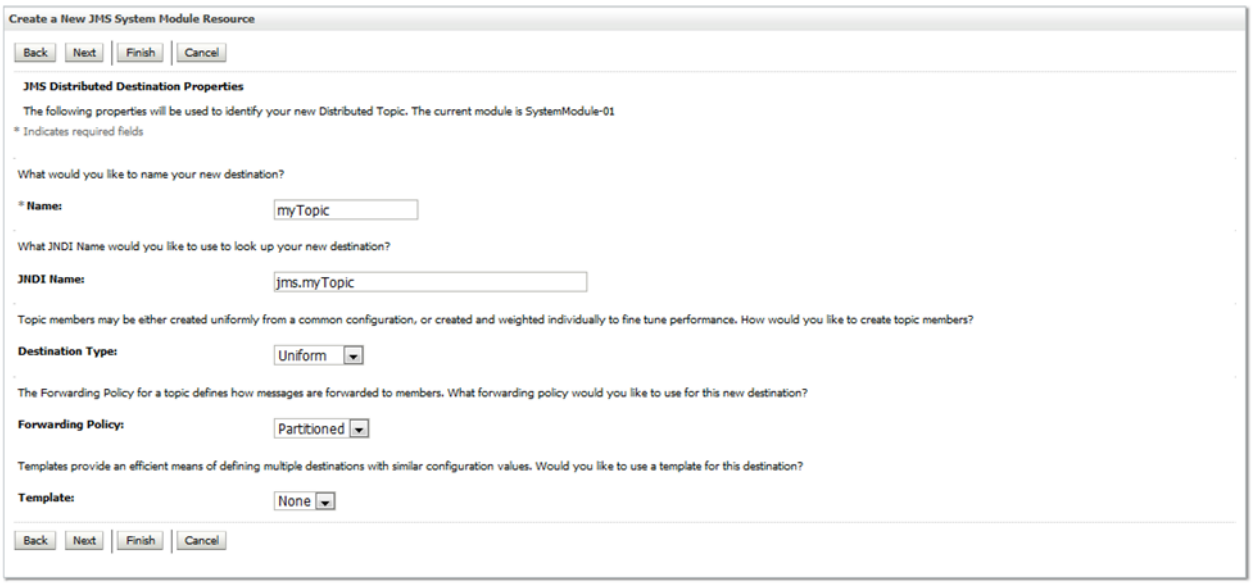

Figure 5- 6 Selecting the partitioned forwarding policy

After selecting the partitioned forwarding policy the next step is targeting the distributed topic sub deployments. The sub deployments also targeted to individual JMS servers distributed to the cluster. This situation can be depicted as follows in figure 5-7.

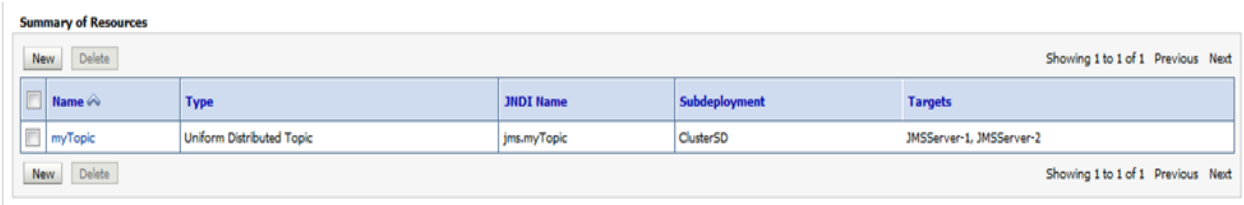

Figure 5- 7 Targeting to sub deployments

The next step is configuring connection factory to make ready to support our requests. This connection factory should allow multiple subscriptions with the same client ID and subscription name (See figure 5-8 below).

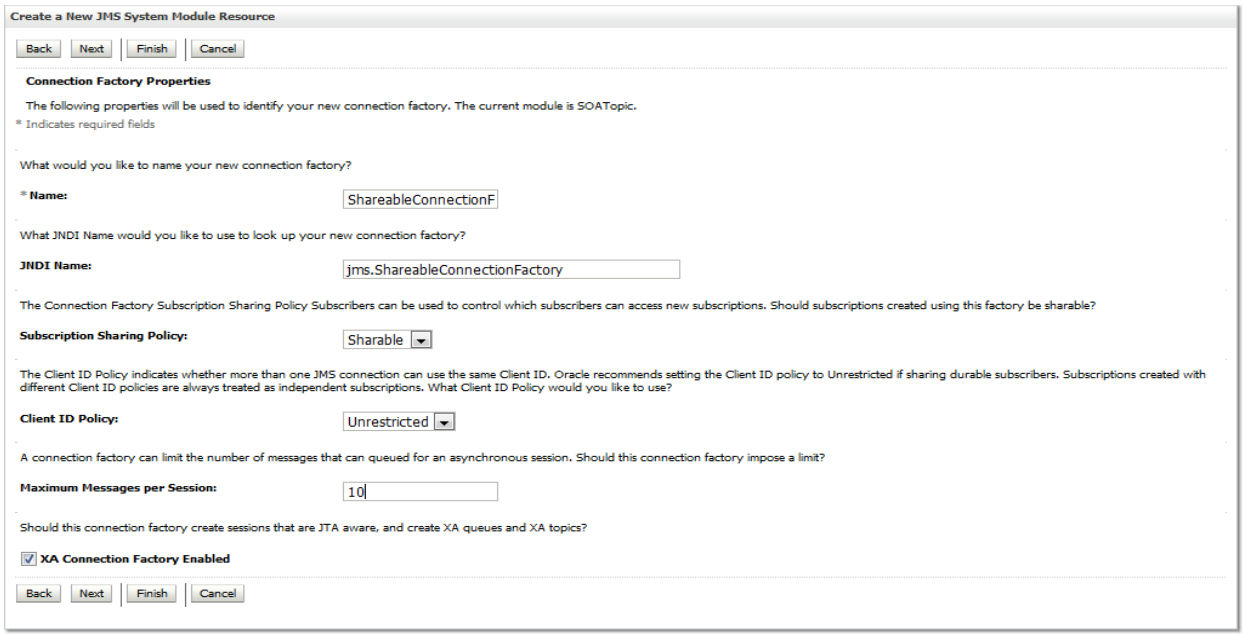

Figure 5- 8 Configuration of connection factory

5.4.1. Implementation and Coding

Based on our case study scenario we have three subscribers two message driven beans(i.e MDB1 and MDB2) and producer swing application. Both applications are identical and composed of a single message driven bean. The message driven bean receives the message and print the body of the message to the standard output device along with the name of the application receiving the message. This method makes easy to check and prove that the same message is going to be received by multiple applications.

To do so we used eclipse platform and created an enterprise aplication project consisting of two modules EJBModule01 and EJBModule02 having classes.

The following code represents an EJB code for application01 receiving the text message and displays an output.

Message Driven Bean (MDB1) EJBModule01 Code

```
package com.debremarkos1;
// imports here
@MessageDriven(activationConfig = {
  @ActivationConfigProperty(propertyName = "destinationType", propertyValue = "javax.jms.Topic"),
  @ActivationConfigProperty(propertyName = "connectionFactoryJndiName", propertyValue = "jms.ShareableConnectionFactory"),
  @ActivationConfigProperty(propertyName = "topicMessagesDistributionMode", propertyValue = "One-Copy-Per-Application"),
  @ActivationConfigProperty(propertyName = "subscriptionDurability", propertyValue = "Durable") }, mappedName = "jms.myTopic")
public class App01Subscriber implements MessageListener {
public void onMessage(Message message) {
  try {
    TextMessage textMessage = (TextMessage) message;
    String text = textMessage.getText();
    System.out.println("Message '" + text + "' received by Application 01");
  } catch (JMSException e) {
    throw new EJBException(e);
  }}}
```

we can also write the same code ith MDB1 for MDB2 ("Message '" + text + "' received by Application 02");

As we can see from the above configuration code there are four basic configurations made using @ActivationConfigProperty. These are :

Destinationtype: Defines the destination type as javax.jms.Topic

Connectionfactory jndiname: Describes the connection factory used by the message driven bean as jms.ShareableConnectionFactory. In this case as the connection factory created, it lets the client ID to be unrestricted and the subscription name to be shared .

Topic messages distributionmode: We declared that One-Copy-Per-MDB, which means that each application will receive only one copy of every message sent to that particular topic, regardless of the number of application instances available. In this case different MDBs in the same aplication can receive one copy of each message.

Subscription durability: We defined the message subscription as durable subscription to keep messages in case the receiver is not active at the moment the message is published.

Message publisher/producer

Producer is an application which is responsible for inputting messages into the topic. The class shown below is the Swing client code for a simple producer that has the responsibility of taking the text input from a text box and sending it as a JMS text message.

If you run this code it will display a button to accept the text message to send to the consumer client application/subscriber as shown below in figure 5-9.

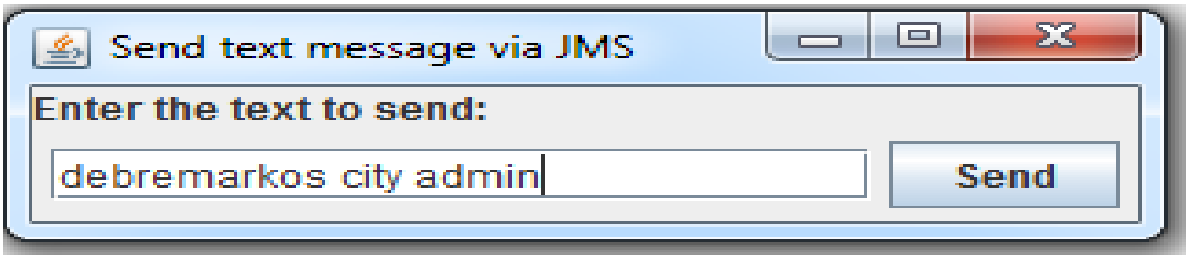

Figure 5- 9 Message producer client application

Therefore the two applications are deployed so, we can access the services from the oracle weblogic server administration console page.If you open the JMS module(SystemModule-01) each message driven beab(MDB1 and MDB2) will have a defualt of 16 consumers as shown in figure 5-10 .

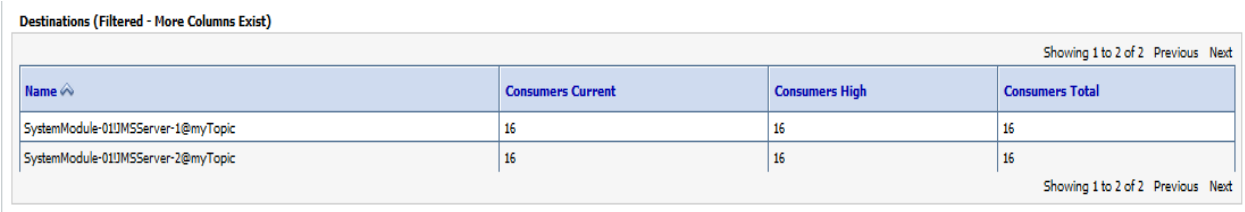

Figure 5- 10 Consumers' view for a single subscriber

As mentioned earlier there are two client applications having a defualt of 16 consuers,so that we can have a total of 32 consumers in each managed servers in the distributed topic(See figure 5-11 below).

Destinations (Filtered - More Columns Exist)

Name	Consumers Current	Consumers High
SystemModule-01\JMSServer-1@myTopic	32	32
SystemModule-01\JMSServer-2@myTopic	32	32

Figure 5- 11 Consumers' view for two subscribers

Durable subscription was created in each member of the distributed topic with the same subscription name. Because the topic is partitioned, messages will be spread among the members and won't be replicated. The consumers are responsible for consuming messages from the member to which they are connected. This allows for parallel processing, improves scalability, and enables to the one-message-per application concept as illustrated in figure 5-12.

Durable Subscribers (Filtered - More Columns Exist)

Delete | Show Messages — Showing 1 to 4 of 4 Previous Next

Uniform Distributed Topic Members	Client ID	Client ID Policy	Subscription Name	Active	Last Message Received Time	Idle Time in Seconds	Messages Current Count	Messages Pending Coun
SystemModule-01\JMSServer-1@myTopic	App01Subscriber	Unrestricted	App01Subscriber	true	11/1/16 1:36:38 AM PDT	11966	0	0
SystemModule-01\JMSServer-1@myTopic	App02Subscriber	Unrestricted	App02Subscriber	true	11/1/16 4:42:29 AM PDT	815	0	0
SystemModule-01\JMSServer-2@myTopic	App01Subscriber	Unrestricted	App01Subscriber	true	11/1/16 1:35:37 AM PDT	12027	0	0
SystemModule-01\JMSServer-2@myTopic	App02Subscriber	Unrestricted	App02Subscriber	true	11/1/16 4:42:20 AM PDT	824	0	0

Delete | Show Messages — Showing 1 to 4 of 4 Previous Next

Figure 5- 12 Consumers' view for durable subscribers

The other step is testing the sender and producer codes for sending a text message to the consumer applications (app01 and app02) in the managed servers found in the cluster. Let's take four message tests (message test1, message test 2, message test3 and message test 4). It results the following output.

Server 1

Message 'message test 2' received by Application 01

Message 'message test 2' received by Application 02

Message 'message test 3' received by Application 01

Message ‘message test 3’ received by Application 02

Server 2

Message ‘message test 1’ received by Application 01

Message ‘message test 1’ received by Application 02

Message ‘message test 4’ received by Application 01

Message ‘message test 4’ received by Application 02

As we see from the above output displayed the messages are divided per the managed server for load balancing between the managed servers. This is the result of the connection factory which plays a great role in balancing the load among the managed servers.

5.4.2. Results and Discussions

If we have an initial number of sixteen consumers of each two applications (MDB1 and MDB2), we can find the statistics of the consumers that each member has thirty two consumers.Then the durable subscription is created in each distributed topic with the same subscription name. Hence, the topic become partitioned and the messages are going to be distributed among the memebers and they would not be replicated. This is done by the help of shared connection factory. Therefore the consumers are able to consume the message on which they are connected with a system. This allows a system the ability to perform parallel processing, improved scalability and considering a single message delivered per application concept. In addition to these the system enables the service offering organization the ability to track whether the messages are delivered to the targeted consumer or not because it displays the consumers getting down during message delivery time. And also it helps the consumers to get messages that are received during offline state. Therefore there is no message loss.

CHAPTER SIX

6. CONCLUSION AND RECOMMENDATIONS

6.1. Conclusion

Service-oriented architecture (SOA) is an architectural pattern in computer software design in which application components provide services to other components through a communication medium usually over the network. The principles of service-orientation are independent of any vendor, product or technology. Since, service integrity and delivery system is one of the research issues in Ethiopia which is not yet addressed appropriately.

In this research work we have studied service integrity and designing service oriented architecture to Debre Markos city for integrating heterogeneous services to provide accurate and efficient services and delivering important data and information to service users on time.

The major intent of this research work is that developing a new service oriented architecture mainly focusing on service integrity and delivery system which enables city managers to integrate different services effectively and efficiently in an organizations. The proposed SOA based service integration architecture has some advanced features and improvements done in [26].This architecture provides improved service delivery system using publish/subscribe messaging model. The Publish-subscribe messaging model provides message delivery to multiple consumers through a topic even if the consumers are offline by the time, they will receive the message when they become active state.

The main objective of developing service oriented city architecture is used for facilitating access of information for consumers in a stable, confidential and efficient manner in all 24 hours of a day. It also facilitates public participation and interaction of citizens with the government and private sectors in urban affairs is one of the key indicator for the development of a city. Service oriented architecture is a suitable paradigm for service integrity and delivery system. In this research work we proposed in study that a seven layer architecture consisting of service consumer's layer, service integration layer, communication channel layer, electronic data sources layer, security services layer, data base and servers layer and network infrastructures layer. This architecture provide improved service integration and delivery system in Debre Markos city.

6.2.Recommendations

Our newly proposed architecture provides reliable and efficient service integrity using publish-subscribe messaging model using partitioned distributed topics with durable subscribers for Debre Markos city which is applicable for other cities of Ethiopia. We recommend that in addition to the interaction and message exchange system, it is important to study the possibilities of protocol conversion, the effects of the number of users in the efficiency of the system as a future work.

REFERENCES

[1] Freeman, P. and Aspary, W, "The Supply of Information Technology Workers in the United States," Published by Jossey-Bass, A Wiley Company, Washington D.C., 1999.

[2] E. Asgharizadeh, M. AjalliGeshlajoughi, and S. R. SafaviMirmahalleh, "‖ Identification of the Electronic City Application Obstacles in Iran," International Journal of Human and Social, pp. 15-163, 2009..

[3] EvgenDuravkin,E, "Using SOA for development of information system (Smart city)‖,," Modern Problems of Radio Engineering, Telecommunications and Computer Science (TCSET), pp. 258 – 258, 12 April 2010.

[4] M.R. Zynoddini,, "an Enterprise Architecture Framework for Electronic City," Master Thesis, Payame Noor University, Iran, 2011.

[5] chapter 1:service oriented architecture, "Msdn.microsoft.com. Retrieved on december 1, 2016".

[6] What is SOA?, "opengroup. Retrieved 2015-08-18".

[7] England, E. and Lester, H. (2005)., "Integrated Mental Health Services in England: A Policy Paradox?," International Journal of Integrated Care, (5).

[8] Derek T. Sanders, J. A. Hamilton, Jr and Richard A. MacDonald, ""Supporting a service-oriented architecture"," in in Proceedings of the 2008 Spring simulation multi conference,ISBN:1-56555-319-5.

[9] Sun Microsystems, "Jini Network Technology,http://www.sun.com/software/jini/".

[10] Jeff A. Estefan, "Exploring Open Software Standards for Enterprise e-business Computing," August 28, 2000, IBM International Technical Support Organization.

[11] Sun Microsystems, "Java Message Service Specification," Version 1.1 http://java.sun.com/products/jms/docs.html.

[12] G. S. S. D. Harter, ""Sustainable Urbanization The Role of ICT in City Development"," Booz & Company, 2014.

[13] Flak, L.S., Olsen, D.H. and Wolcott, P (2005), "Local E-Government in Norway"," Scandinavian Journal of Information Systems, Vols. Vol 17, No.2, pp. 41 – 48..

[14] The Working Group on E-Government in the Developing World, "Roadmap for E-government in the Developing World, 10 Questions E-Government Leaders Should Ask Themselves," April 2002.

[15] Thomas Erl, "Service-Oriented Architecture (SOA): Concepts, Technology, And Design," in Prentice Hall Ptr, 760p. 2005.

[16] Smart GIS/IT, "The city of Cape Town in SouthAfrica has pursued a "smart city" goal through the integration of GIS with it system," GEOconnexion International Magazine, 2007.

[17] Thomas Erl, (2008), "SOA Principles of Service Design," ISBN: 0-13-234482-3.

[18] Thomas Erl, (2004), Service-Oriented Architecture:, "A Field Guide to Integrating XML and Web Services," Prentice Hall PTR, Upper Saddle River, NJ, USA.

[19] Valipour, M.H. Amirzafari, B. Maleki, K.N. Daneshpour, N, ""A brief survey of software architecture concepts and service oriented architecture"," in in Proceedings of 2nd IEEE International Conference on Computer Science and Information Technology, ICCSIT'09, ISBN: 978-1-4244-4519-6.

[20] Linthicum David S., "Enterprise Application Integration [Book]. - [s.l.]:," Addison Wesley, First Edition November 05, 1999.

[21] "Top Reasons to Implement SOA," July 2013.

[22] S. T. M. E. S. M. James McGovern, " Java Web Services Architectre," Morgan Kaufmann Publishers, 2013.

[23] Microsoft Corporation (2006), "Enabling "Real World SOA" through the Microsoft Platform".

[24] World Wide Web Consortium (W3C), "Web Services Architecture," pp. http://www.w3.org/TR/ws-arch/, 11 February 2004.

[25] Arcieri, F., et al., (2002), "Experiences and issues in the realization of e-government services," in in : Proc. of the 12th International Workshop on Research Issues in data engineering, Engineering E-commerce / E- Business Systems (RIDE‘02), IEEEE, New York, 143-148.

[26] SajjadHashemi, S. Y. (DECEMBER 2012)., "A Novel Service Oriented Architecture for Integration of Information Systems in Electronic city," INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY RESEARCH(IJITR),vol. VOLUME 1(ISSUE 11)..

[27] Hashemi1, S. (November 2013)., "A NOVEL SOA FOR E-CITY WITH THE INCREASE OF CITIZENS PARTICIPATION APPROACH," International Journal of Information Technology, Modeling and Computing (IJITMC),, vol. volume 1..

[28] Sajjad Hashemi1, S. Y. (December 2013)., "A NOVEL SERVICE ORIENTED ARCHITECTURE (SOA) TO SECURE SERVICES IN E-CITY," International Journal of Security, Privacy and Trust Management (IJSPTM), vol. voluume 2 No. 6..

[29] R. A. O. K. Zakaria I. Saleh, "A Framework for an E-government Based on Service Oriented Architecture for Jordan," I.J. Information Engineering and Electronic Busines, Vols. 3, 1-10, 2013, 3, 1-10.

[30] Munday, B. (2007), "Integrated Social Services in Europe," Council of Europe Publishing.

[31] Sloper, P. (2004)., "Facilitators and Barriers for Co-ordinated Multi-Agency Services," Child: Care, Health & Development, pp. (30)6, pp 571-580..

[32] Rosenheck, R. A., Resnick, S. G., and Morrissey, J. P. (2003), "Closing Service System Gaps for Homeless Clients with Dual Diagnosis,Integrated Teams and Interagency Cooperation," The Journal of Mental Health Policy and Economics, (6).

[33] Kodner, D. L. and Spreeuwenberg, C. (2002)., "Integrated Care: Meaning, Logic, Applications, and Implications-a Discussion Paper," International Journal of Integrated Care, (2).

[34] Kodner, D. L. and Spreeuwenberg, C. (2002)., "Integrated Care: Meaning, Logic, Applications, and Implications-a Discussion Paper," International Journal of Integrated Care, (2).

[35] United Nations ICT Task Force (2005), "Information and Communication Technologies and the Millennium Development Goals," in UN ICT Task Force, New York..

[36] UN Millennium Project (2005) Innovation, "Applying Knowledge in Development.," UN Task Force on Science, Technology, and Innovation, New York.

[37] Bieber, Carpenter, Stevens, "Jini Technology Architectural Overview," in SunMicrosystems, 2001.

[38] Sinrod, Eric J. (2004-06-30)., ""A look at the pros and cons of e-government"," USA Today. Retrieved 2008-03-01.

[39] Berntzen, L., Olsen, M.G, "Benchmarking e-Government – A Comparative Review of Three International Benchmarking Studies," in This paper appears in: Digital Society, ICDS _09. Third International Conference on, pp. 77 – 82,13 February 2009.

[40] M. Atashak, P. Mahzadeh, "E-government Status in Iran (TAKFA Plan Case Study)," World Applied Sciences Journal, pp. 12-20, 2008.

[41] G. Booch, "Object Oriented Design with Applications," Benjamin-Cummings Publishing, 1990.

[42] C. SzyperskiComponent Software, "Beyond Object-Oriented Programming," Addision-Wesley, 1998.

[43] Dirk Slama, Karl Banke, Dirk Krafzig, "Service Oriented Architecture: Inventory of Distributed Computing Concepts," in Prentice Hall PTR, Dec. 10, 2004.

[44] Sun Microsystems, "Java 2 Platform Enterprise Edition Specification," Version 5.0http://java.sun.com/j2ee/.

[45] Workflow Management Coalition, "the Workflow Reference Model, Document Number WFMC-TC00-1003," http://www.wfmc.org/, January-1995.

[46] United Nations Development Programme, "Governance for Sustainable Human Development," A Policy Document', [Online], Available, http://magnet.undp.org/policy/chapter1.htm [03 July 2006]..

[47] Thomas Erl, (2005), "Service-Oriented Architecture: Concepts, Technology, and Design," in Prentice Hall PTR,ISBN: 0-13-185858-0.

[48] Nicolai M.Josuttis, (2007, "SOA in Practice, O'Reilly Media," in Inc., ISBN: 13: 978-0-596-52955-0.

[49] Bieber, Carpenter, Stevens, (2001), "Jini Technology Architectural Overview," in Sun Microsystems.

APPENDICES

A. Interview Questions

We conducted the following basic interview questions during data collection with Debre Markos city administration officials and employees. We conducted the following interview to check the existing strategies and methods of service integration in debre markos city and identify the gaps and problems, and to design suitable service oriented architecture deployed at the center of the city on the provider's data center deliver services in our case message delivery and exchange using distributed topics architecture.

1. Does your organization have access to ICT infrastructure/technology infrastructure or not?
2. If you are equipped with ICT infrastructure what kind of activities are you performing using computers and internet access?
3. What kind of data manipulation approaches suitable for your organization are you using?
    - Manual/traditional? or
    - Computerized/automated approach?

If you are using the manual approach:

- Is this system enough to handle every activities properly? If not what do you feel if you have a chance to use automated system having different desktop computers, mobiles and free internet access?
- What are the problems that limit not to use such a computerized system?
    - Resource cost like software and infrastructure cost?
    - Limited skilled personnel?

If the problem is skill have you tried to outsource a system? I.e. putting an agreement with other companies to do automated system for your organization.

- Can you show me how to do you handle your organizations data?
- Do you have documents which gives a brief description about the system?

If you are using computerized system:

- Is your system handling every necessary requests?
- Can you explain the major functions of the system?
- Can you show me how it works?

B. A List of Sample Data Forms of Different Organizations in Debre Markos City

Different organizations use different data forms for the sake of data exchange between different organizations in the city. We gathered these data forms to study how these organizations exchange information. Therefore they following manual approaches of information exchange using papers as shown below. It needs suitable service integrity architecture so that we proposed SOA based architecture with an implementation prototype. The following forms shows that the current state of data exchange between different organizations in debre markos city using paper.

Debre Markos city administration mayor office documented money ordering form

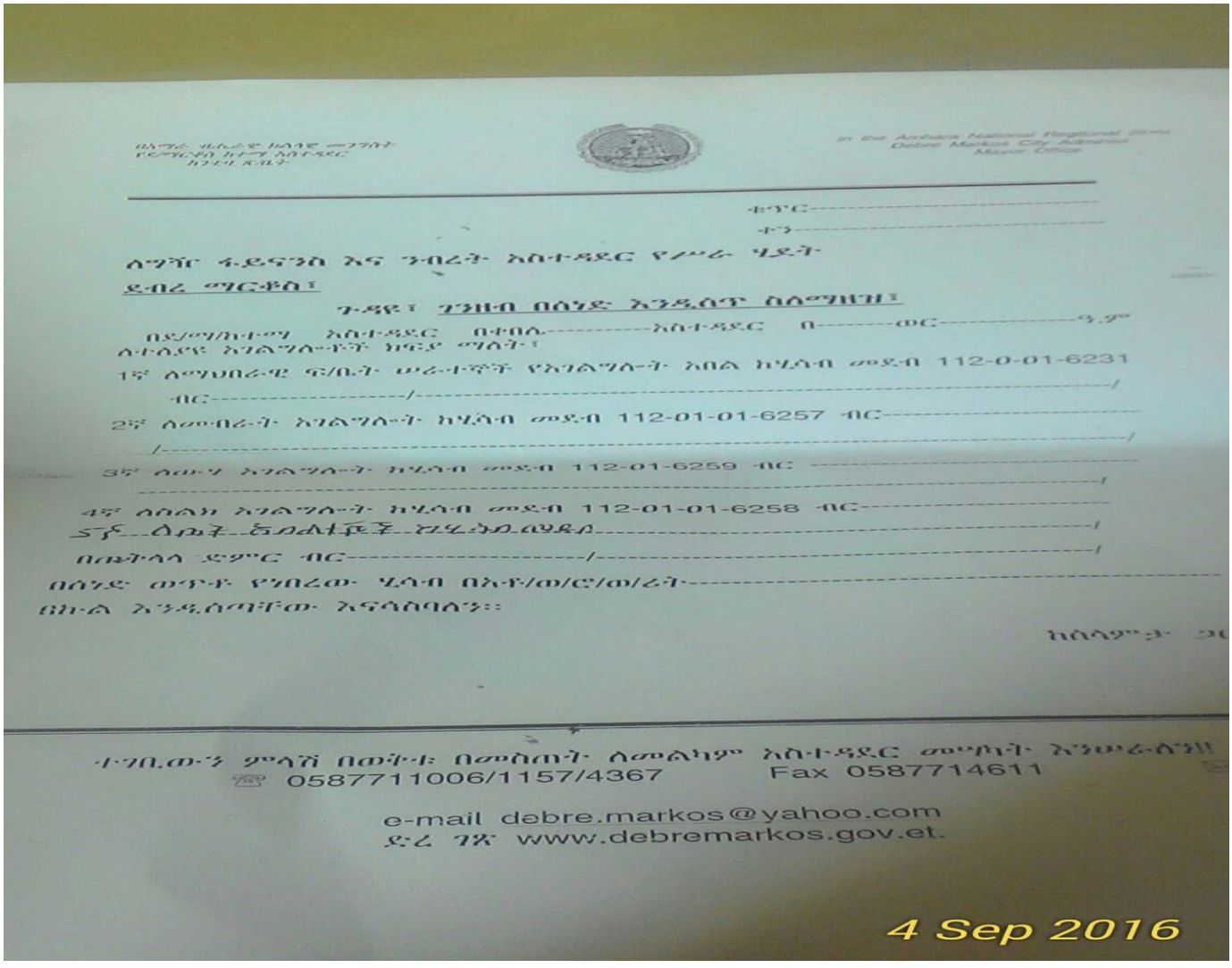

Figure 1 Debre Markos city administration mayor office documented money ordering form

Hiring employees in vacant places form for Debre Markos city administration

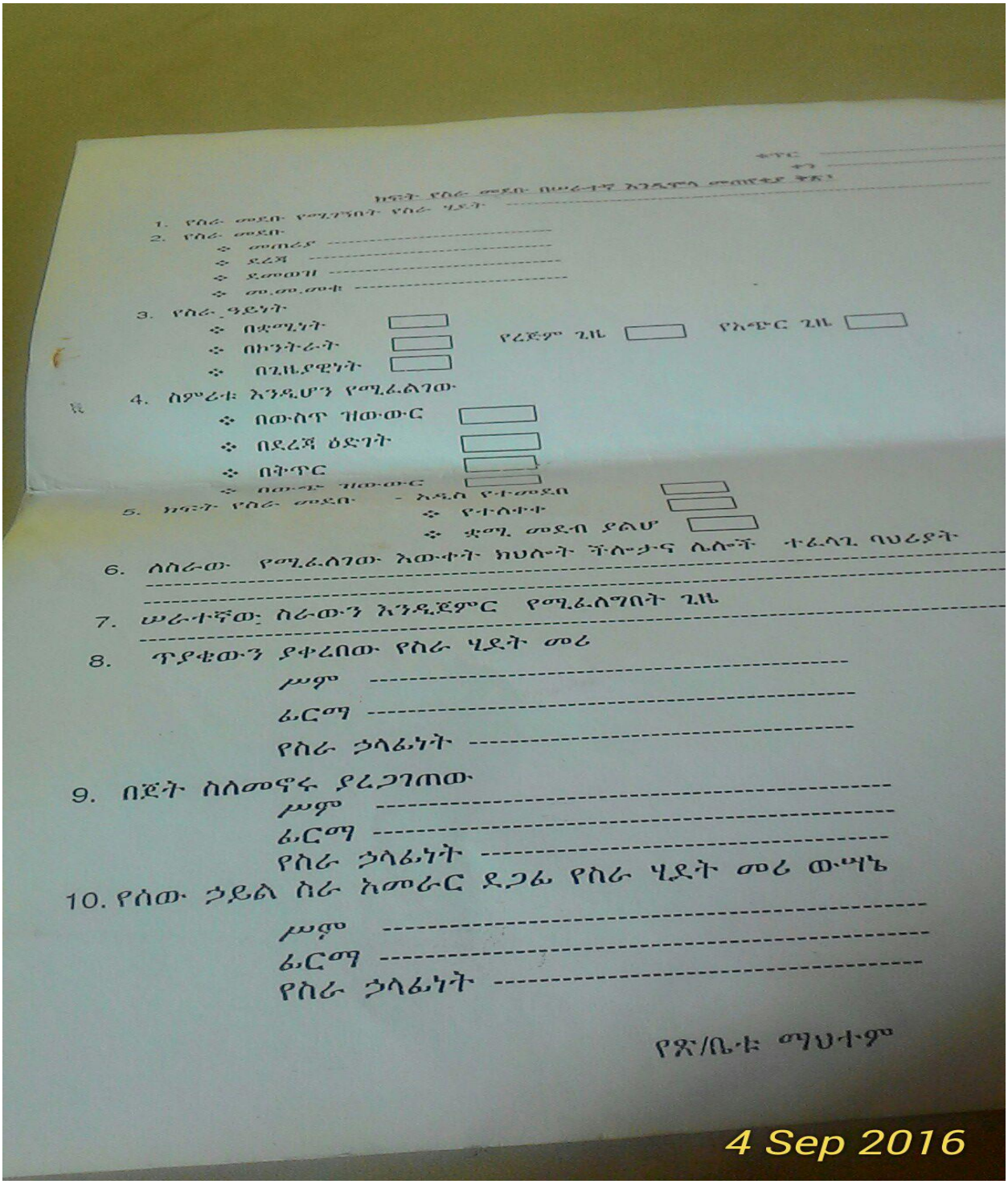

Figure 2 Hiring employees in vacant places form for Debre Markos city administration

Compensation payment ordering form for commercial bank of Ethiopia

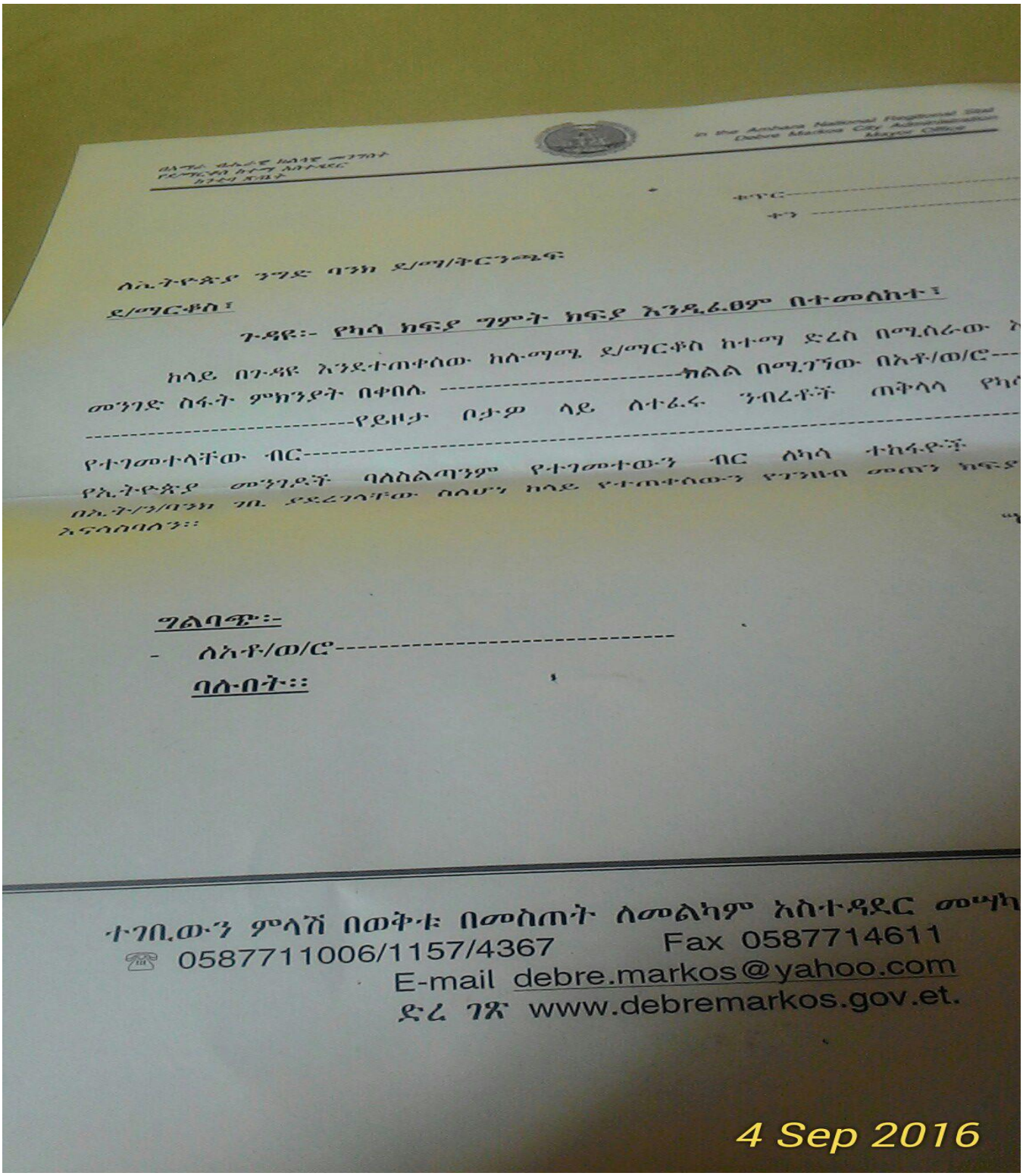

Figure 3 Compensation payment ordering form for commercial bank of Ethiopia

Employee transfer request form for Debre Markos city administration

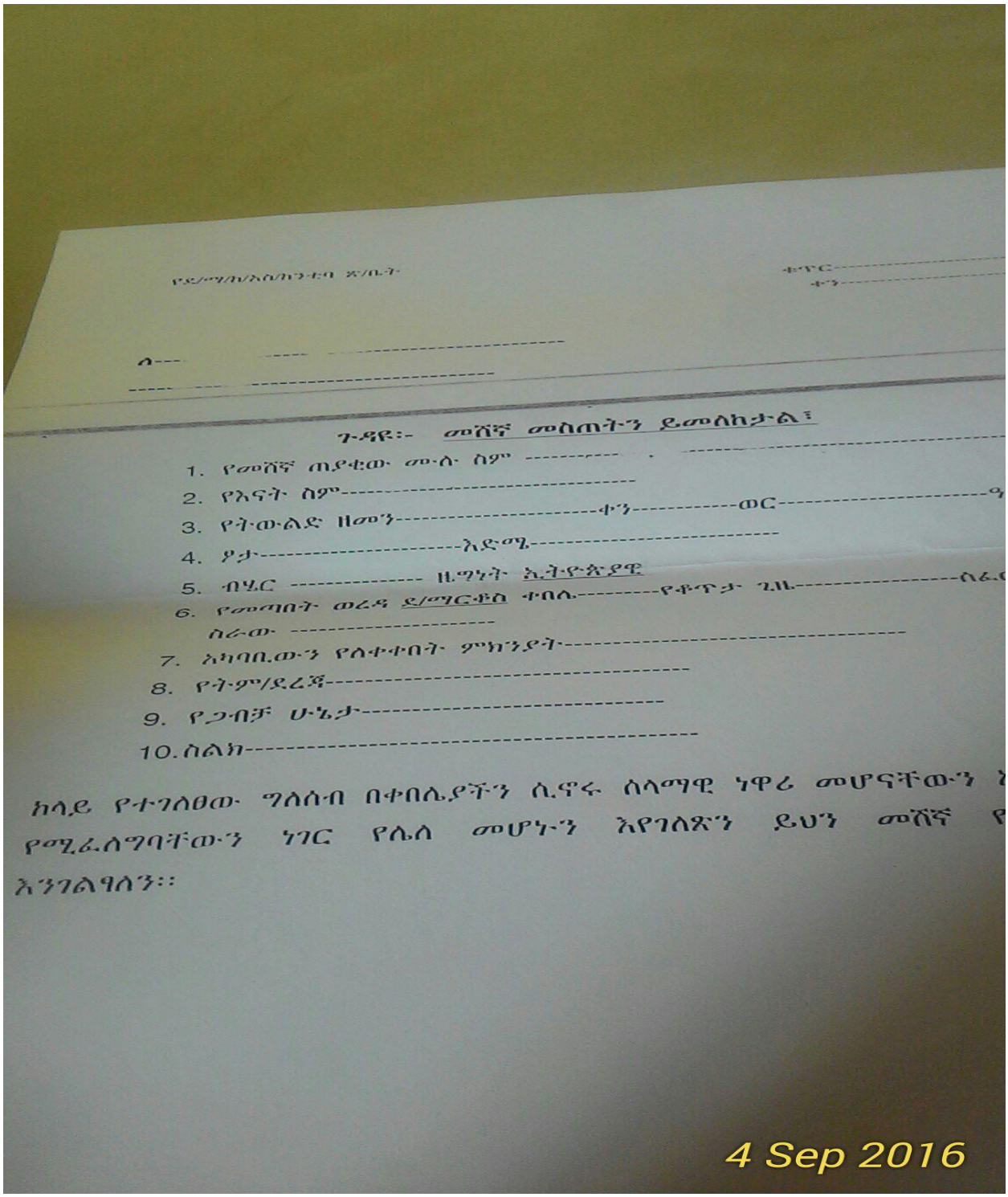

Figure 4 Employee transfer form for Debre Markos city administration

Prototype Implementation and coding

Message Driven Bean (MDB2) EJBModule02 Code

```
package com.debremarkos1;
// imports here
@MessageDriven(activationConfig = {
 @ActivationConfigProperty(propertyName = "destinationType", propertyValue = "javax.jms.Topic"),
 @ActivationConfigProperty(propertyName = "connectionFactoryJndiName", propertyValue = "jms.ShareableConnectionFactory"),
 @ActivationConfigProperty(propertyName = "topicMessagesDistributionMode", propertyValue = "One-Copy-Per-Application"),
 @ActivationConfigProperty(propertyName = "subscriptionDurability", propertyValue = "Durable") }, mappedName = "jms.myTopic")
public class App02Subscriber implements MessageListener {
public void onMessage(Message message) {
 try {
  TextMessage textMessage = (TextMessage) message;
  String text = textMessage.getText();
  System.out.println("Message '" + text + "' received by Application 02");
 } catch (JMSException e) {
  throw new EJBException(e);
 }}}
```

Message producer code

```
package com.debremarkos1;
// imports
public class ProducerClient extends JFrame {
 private static final long serialVersionUID = 3654532969956126528L;
 private JLabel titleLabel;
 private JTextField textField;
 private JButton sendButton;
 public ProducerClient() {
```

```
  super("Send text message via JMS");  }
private void init() {
  SetDefaultCloseOperation (EXIT_ON_CLOSE);
  titleLabel = new JLabel("Enter the text to send:");
  textField = new JTextField(20);
  JPanel panel = new JPanel();
  sendButton = new JButton("Send");
  sendButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
      String text = ProducerClient.this.textField.getText();
      try {
        new MessageSender().send(text);

        textField.setText("");
        JOptionPane.showMessageDialog(ProducerClient.this, "Message sent...",
            "Success", JOptionPane.INFORMATION_MESSAGE);
      } catch (Exception jmsEx) {
        JOptionPane.showMessageDialog(ProducerClient.this,
            "Error sending message. Please check logs.", "Error",
            JOptionPane.ERROR_MESSAGE);
        jmsEx.printStackTrace();
      } }  });
  panel.add(textField);
  panel.add(sendButton);
  getContentPane().add(titleLabel, BorderLayout.NORTH);
  getContentPane().add(panel, BorderLayout.CENTER);
  pack();
  setVisible(true);  }
public static void main(String args[]) {
  SwingUtilities.invokeLater(new Runnable() {
```

```
    Public void run() {
      new ProducerClient().init();
    } }); }}
```

Message sender client code

```
package com.debremarkos1;
import java.util.Arrays;
import java.util.Date;
import java.util.Hashtable;
import javax.jms.BytesMessage;
import javax.jms.Connection;
import javax.jms.ConnectionFactory;
import javax.jms.JMSException;
import javax.jms.Message;
import javax.jms.MessageConsumer;
import javax.jms.MessageListener;
import javax.jms.MessageProducer;
import javax.jms.Queue;
import javax.jms.Session;
import javax.jms.Topic;
import javax.jms.TopicSubscriber;
import javax.naming.Context;
import javax.naming.InitialContext;
@SuppressWarnings("all")
public class MessageSender {
  public void send(String text) throws Exception {
    Context context = new InitialContext();
    ConnectionFactory cf;
    Connection connection = null;
    Session session = null;
    MessageProducer producer = null;
    try {
```

```
    cf = (ConnectionFactory) context.lookup("weblogic.jms.ConnectionFactory");
    Topic topic = (Topic) context.lookup("jms.myTopic");
    connection = cf.createConnection();
    session = connection.createSession(false, Session.AUTO_ACKNOWLEDGE);
    producer = session.createProducer(topic);
    producer.send(session.createTextMessage(text));
   } finally {
    closeAll(producer, session, connection, context);
   }  }
  private void closeAll(MessageProducer producer, Session session,
    Connection connection, Context context) {
   try {
    producer.close();
   } catch (Exception e) {
   }
   try {
    session.close();
   } catch (Exception e) {
   }
   try {
    connection.close();
   } catch (Exception e) {
   }
   try {
    context.close();
   } catch (Exception e) {
   }  }}
```

Message Consumer Client Application/Subscriberapp01 Code

```
package com.debremarkos1;
import java.awt.BorderLayout;
import java.awt.event.WindowAdapter;
```

```
import java.awt.event.WindowEvent;
import java.text.SimpleDateFormat;
import java.util.Date;
import javax.swing.JFrame;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.SwingUtilities;
import javax.swing.table.DefaultTableModel;
public class ConsumerClient extends JFrame {
  private static final long serialVersionUID = 3654532969956126528L;
  private JTable messagesTable;
  private DistributedDestinationController distributedDestinationController = new
DistributedDestinationController(
      this);
  public ConsumerClient() {
    super("Messages Received"); }
  private void init() {
    Object columns[] = { "Message", "Receive time" };
    messagesTable = new JTable(new DefaultTableModel(columns, 0));
    getContentPane().add(new JScrollPane(messagesTable), BorderLayout.CENTER);
    distributedDestinationController.registerListeners();
    addWindowListener(new WindowAdapter() {
      @Override
      public void windowClosing(WindowEvent e) {
        ConsumerClient.this.distributedDestinationController
            .unregisterListeners();
        System.exit(0);
      }; });
    pack();
    setVisible(true);  }
  public void onTextMessage(final String text) {
```

```
   SwingUtilities.invokeLater(new Runnable() {
     @Override
     public void run() {
       DefaultTableModel model = (DefaultTableModel) messagesTable.getModel();
       model.insertRow(model.getRowCount(), new Object[] { text,
           new SimpleDateFormat("dd/MM/yyyy HH:mm:ss").format(new Date()) });
     }  });}
  public static void main(String args[]) throws Exception {
   SwingUtilities.invokeLater(new Runnable() {
     public void run() {
       ConsumerClient consumerClient = new ConsumerClient();
       consumerClient.init();
     }   });
  }}
```

Message Consumer Client Application/Subscriberapp02 Code

```
package com.debremarkos1;
import java.awt.BorderLayout;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;
import java.text.SimpleDateFormat;
import java.util.Date;
import javax.swing.JFrame;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.SwingUtilities;
import javax.swing.table.DefaultTableModel;
public class ConsumerClient extends JFrame {
  private static final long serialVersionUID = 3654532969956126528L;
  private JTable messagesTable;
  private DistributedDestinationController distributedDestinationController = new
DistributedDestinationController(
```

```
    this);
public ConsumerClient() {
  super("Messages Received");  }
private void init() {
  Object columns[] = { "Message", "Receive time" };
  messagesTable = new JTable(new DefaultTableModel(columns, 0));
  getContentPane().add(new JScrollPane(messagesTable), BorderLayout.CENTER);
  distributedDestinationController.registerListeners();
  addWindowListener(new WindowAdapter() {
    @Override
    public void windowClosing(WindowEvent e) {
      ConsumerClient.this.distributedDestinationController
          .unregisterListeners();
      System.exit(0);
    };   });
  pack();
  setVisible(true);  }
public void onTextMessage(final String text) {
  SwingUtilities.invokeLater(new Runnable() {
    @Override
    public void run() {
      DefaultTableModel model = (DefaultTableModel) messagesTable.getModel();
      model.insertRow(model.getRowCount(), new Object[] { text,
          new SimpleDateFormat("dd/MM/yyyy HH:mm:ss").format(new Date()) });
    }    });}
public static void main(String args[]) throws Exception {
  SwingUtilities.invokeLater(new Runnable() {
    public void run() {
      ConsumerClient consumerClient = new ConsumerClient();
      consumerClient.init();
    } });  }}
```