

**FRAMEWORK FOR INTEROPERABLE AND DISTRIBUTED
EXTRACTION-TRANSFORMATION-LOADING (ETL) BASED ON
SERVICE ORIENTED ARCHITECTURE**

MOHAMMED M. I. AWAD

**DOCTOR OF PHILOSOPHY
UNIVERSITI UTARA MALAYSIA
2012**



Awang Had Salleh Graduate School of Arts and Sciences

UUM CAS

Engaging Minds for a Better Tomorrow

PERAKUAN KERJA TESIS / DISERTASI

(Certification of thesis / dissertation)

Kami, yang bertandatangan, memperakukan bahawa
(We, the undersigned, certify that)

MOHAMMED M. I. AWAD

calon untuk ijazah
(candidate for the degree of)

PhD

telah mengemukakan tesis / disertasi yang bertajuk:
(has presented his/her thesis / dissertation of the following title):

"FRAMEWORK FOR INTEROPERABLE AND DISTRIBUTED EXTRACTION-TRANSFORMATION-LOADING (ETL) BASED ON SERVICE ORIENTED ARCHITECTURE"

seperti yang tercatat di muka surat tajuk dan kulit tesis / disertasi.
(as it appears on the title page and front cover of the thesis / dissertation).

Bahawa tesis/disertasi tersebut boleh diterima dari segi bentuk serta kandungan dan meliputi bidang ilmu dengan memuaskan, sebagaimana yang ditunjukkan oleh calon dalam ujian lisan yang diadakan pada : **19 Oktober 2011.**

That the said thesis/dissertation is acceptable in form and content and displays a satisfactory knowledge of the field of study as demonstrated by the candidate through an oral examination held on:

October 19, 2011.

Pengerusi Viva:
(Chairman for VIVA)

Prof. Dr. Norshuhada Shiratuddin

Tandatangan
(Signature)

Pemeriksa Luar:
(External Examiner)

Assoc. Prof. Dr. Wan Mohd Nasir Wan Kadir

Tandatangan
(Signature)

Pemeriksa Dalam:
(Internal Examiner)

Dr. Nor Laily Hashim

Tandatangan
(Signature)

Nama Penyelia/Penyelia-penyelia:
(Name of Supervisor/Supervisors)

Dr. Mohd Syazwan Abdullah

Tandatangan
(Signature)

Tarikh:

(Date) **October 19, 2011**

Permission to Use

In presenting this thesis in fulfilment of the requirements for a postgraduate degree from Universiti Utara Malaysia, I agree that the Universiti Library may make it freely available for inspection. I further agree that permission for the copying of this thesis in any manner, in whole or in part, for scholarly purpose may be granted by my supervisor(s) or, in their absence, by the Dean of Awang Had Salleh Graduate School of Arts and Sciences. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to Universiti Utara Malaysia for any scholarly use which may be made of any material from my thesis.

Requests for permission to copy or to make other use of materials in this thesis, in whole or in part, should be addressed to:

Dean of Awang Had Salleh Graduate School of Arts and Sciences
UUM College of Arts and Sciences
Universiti Utara Malaysia
06010 UUM Sintok

Abstrak

Pengekstrakan, Transformasi dan Proses Pemuatan (ETL) merupakan fungsi-fungsi utama dalam penyelesaian gudang data. Kekurangan komponen pengagihan dan saling kendali menyebabkan wujudnya lompang yang menimbulkan banyak masalah dalam domain ETL. Ini terjadi kerana komponen-komponen dalam kerangka kerja ETL sedia ada adalah saling berkait. Kajian ini membincangkan bagaimana untuk mengagihkan komponen-komponen ETL supaya komponen pengagihan dan saling kendali dapat dilaksanakan. Tambahan pula, kajian ini menunjukkan bagaimana kerangka kerja ETL dapat diperluaskan. Untuk mencapai tujuan tersebut, Perkhidmatan Berorientasikan Seni Bina digunakan untuk memperjelaskan ciri-ciri pengagihan dan saling kendali yang tidak wujud sebelum ini, dengan cara menyusun semula kerangka kerja ETL. Kajian ini menyumbang kepada bidang ETL dengan penambahan konsep pengagihan dan saling kendali kepada kerangka kerja ETL. Seterusnya, kajian ini juga menyumbang kepada bidang penggudangan data dan kepintaran perniagaan kerana ETL merupakan konsep utama dalam bidang ini. Metodologi Design Science Approach (DSA) dan Scrum digunakan untuk mencapai matlamat kajian ini. Integrasi di antara kedua-dua metodologi tersebut dapat mencapai objektif kajian ini. Kerangka kerja ETL yang baru ini direalisasikan menerusi pengujian dan penghasilan satu prototaip yang berdasarkan kepada kerangka kerja tersebut. Kejayaan prototaip ini dinilai berdasarkan tiga kajian kes yang melibatkan data dan alatan daripada tiga organisasi. Organisasi tersebut menggunakan penyelesaian gudang data untuk menghasilkan laporan statistik yang membolehkan pengurusan atasan membuat keputusan. Dapatan ketiga-tiga kajian kes ini menunjukkan komponen pengagihan dan saling kendali dapat dicapai dengan menggunakan kerangka kerja yang baru dalam ETL.

Katakunci: Pengekstrakan, Transformasi dan Proses Pemuatan, Gudang data, Sistem, Perkhidmatan berorientasikan seni bina.

Abstract

Extraction, Transformation and Loading (ETL) are the major functionalities in data warehouse (DW) solutions. Lack of component distribution and interoperability is a gap that leads to many problems in the ETL domain, which is due to tightly-coupled components in the current ETL framework. This research discusses how to distribute the Extraction, Transformation and Loading components so as to achieve distribution and interoperability of these ETL components. In addition, it shows how the ETL framework can be extended. To achieve that, Service Oriented Architecture (SOA) is adopted to address the mentioned missing features of distribution and interoperability by restructuring the current ETL framework. This research contributes towards the field of ETL by adding the distribution and interoperability concepts to the ETL framework. This leads to contributions towards the area of data warehousing and business intelligence, because ETL is a core concept in this area. The Design Science Approach (DSA) and Scrum methodologies were adopted for achieving the research goals. The integration of DSA and Scrum provides the suitable methods for achieving the research objectives. The new ETL framework is realized by developing and testing a prototype that is based on the new ETL framework. This prototype is successfully evaluated using three case studies that are conducted using the data and tools of three different organizations. These organizations use data warehouse solutions for the purpose of generating statistical reports that help their top management to take decisions. Results of the case studies show that distribution and interoperability can be achieved by using the new ETL framework.

Keywords: Extraction, Transformation and Loading, Data warehousing, Service Oriented Architecture.

List of Publications, Invited Speaker and Awards

Awad, M. M. I., & Abdullah, M. S. (2010). A Framework for Interoperable Distributed ETL Components Based on SOA. *Proceeding of the 2nd International Conference on Software Technology and Engineering (ICSTE 2010)*.

Awad, M. M. I., & Abdullah, M. S. (2010). Extending ETL Framework using Service Oriented Architecture. *Procedia Computer Science Journal*, (3), 110-114.

Mohammed M I Awad - Keynote Speaker¹. A Framework for Open-Source Interoperable Distributed ETL Components Based on SOA². *Malaysia Open Source Conference 2010 (MOSC 2010)*.

Best Award. *Malaysia Technology Expo 2011 (MTE 2011)*.

Gold Medal. *Malaysia Technology Expo 2011 (MTE 2011)*.

Silver Medal. *Seoul International Invention Fair 2011 (SIIF 2011)*, South Korea.

Gold Medal. *Malaysia Technology Expo 2012 (MTE 2012)*.

¹ <http://conf.oss.my/speakers.html>

² <http://conf.oss.my/schedule.html>

Acknowledgement

By the Name of Allah, the Most Gracious and the Most Merciful

My most profound thankfulness goes to my father and mother who have motivated me since my childhood to help me reach this success point. Thank you my parents and I hope to be the good son who can make you proud of him.

My deep thankfulness goes to my supervisor Dr. Mohd Syazwan Abdullah for supervising me during the journey of this research. I like his way of conducting the research, especially, his instructions for reviewing the literature, and the whole research as well.

Last but not least, I would like to thank my brothers and sisters as well as my friends.

Table of Contents

Permission to Use	i
Abstrak	ii
Abstract	iii
List of Publications, Invited Speaker and Awards	iv
Acknowledgement	v
Table of Contents	vi
List of Tables	xi
List of Figures	xii
List of Appendices	xv
List of Abbreviations	xvi
CHAPTER ONE INTRODUCTION	1
1.1 Research Background	1
1.2 Research Motivation	4
1.3 Problem Statement	7
1.4 Research Questions	11
1.5 Research Objectives	11
1.6 Research Strategy	13
1.7 Scope of the Research	15
1.8 Contributions	16
1.9 Thesis Organization	18
1.10 Conclusion	19
CHAPTER TWO EXTRACTION, TRANSFORMATION, AND LOADING IN DATA WAREHOUSING	20
2.1 Data Warehouse	20
2.2 Brief Discussed Overview of traditional ETL Framework Components	22
2.3 Industrial ETL Tools	26
2.4 Component Coupling in Current ETL Structure	30
2.5 Components Distribution	32
2.6 Compatibility of ETL Components with Heterogeneous Environments	33
2.7 Extensibility and Scalability of ETL Framework	36

2.8 ETL Tools Administration	38
2.9 ETL Tools Licensing	40
2.10 Conclusion	41
CHAPTER THREE DISTRIBUTED TECHNOLOGIES	42
3.1 Distributed Systems	42
3.1.1 Remote Procedure Call (RPC)	45
3.1.2 Common Object Request Broker Architecture (CORBA)	46
3.1.3 Distributed Component Object Model (DCOM)	48
3.1.4 Remote Method Invocation (RMI)	49
3.1.5 Service Oriented Architecture (SOA)	51
3.1.6 Comparison of Distributed Technologies	53
3.2 Application Architectures and SOA	57
3.2.1 Types of Application Architectures	58
3.3 Extensible Markup Language (XML)	62
3.4 Web Services in SOA	63
3.4.1 Web Services versus SOA Concepts	63
3.4.2 Simple Object Access Protocol (SOAP) in Web Services	64
3.4.3 Web Services Description Language (WSDL)	64
3.4.4 Messaging	65
3.5 Integration Benefits of Adopting SOA in the ETL Framework	66
3.6 Conclusion	67
CHAPTER FOUR RESEARCH METHODOLOGY	68
4.1 Design Science Approach (DSA)	69
4.1.1 DSA phases	70
4.2 Scrum Methodology	75
4.2.1 Scrum Artifacts	76
4.2.2 SCRUM Phases	77
4.3 New ETL Framework Development Using DSA	78
4.4 Applying Scrum Methodology for the Development Phase of DSA	87
4.5 Conclusion	93
CHAPTER FIVE THE NEW ETL FRAMEWORK	94

5.1 Overview of the New ETL Framework	94
5.2 Distributed Architecture Specifications	100
5.3 Web Services Involvement Specifications.....	101
5.3.1 A Web Service for every Distributed ETL Component.....	103
5.3.2 Web Service Operations.....	103
5.3.3 WSDL Documents	104
5.3.4 XML Schema	107
5.4 Orchestration Point Specifications	109
5.5 Specifications of the Composition of Partners and Configurations Based on SOA	112
5.6 Specifications of Extending the New ETL Framework	115
5.7 Meta-Model for the New ETL Framework	117
5.8 Feedback from the Experts Regarding the Theoretical Framework	128
5.9 Conclusion	131
CHAPTER SIX SOA-BASED ETL PROTOTYPE.....	133
6.1 Analysis.....	133
6.1.1 Requirements (Prototype Backlog) Determination.....	134
6.1.2 Backlog Division	135
6.2 Database Sprint	138
6.3 Coding ETL Components Sprint.....	143
6.3.1 Extraction Task	145
6.3.2 Transformation Task	147
6.3.3 Classified-Fragmentation Task	149
6.3.4 Loading Task	151
6.4 Distributed Components and Web Services Sprint.....	153
6.5 Business Process Execution Language (BPEL) Creation Sprint	154
6.6 Sprint of Assembling Prototype Components in One Composite Application..	157
6.7 Sprints of Testing	158
6.7.1 Unit Testing Sprint.....	160
6.7.2 Classified-Fragmentation Speed and Scalability Testing Sprint	161
6.7.3 Compatibility Testing Sprint.....	163

6.7.4 End To End Testing Sprint	163
6.8 Conclusion	166
CHAPTER SEVEN EVALUATION.....	167
7.1 Case Study 1: Applying ETL Functionalities on Palestine Electric Company (PEC) using Traditional and New ETL Tools.....	170
7.1.1 ETL Business Needs	170
7.1.2 Extracting Required Fields for Data Warehouse Star-Schema.....	171
7.1.3 Applying ETL Functionalities Using the Traditional ETL Tool	172
7.1.4 SOA-based ETL Prototype	174
7.1.5 Goals Achieved.....	175
7.2 Case Study 2: Applying ETL Functionalities on Limkokwing University of Creative Technology (LUCT) using Traditional and New ETL Tools.....	179
7.2.1 ETL Business Needs	179
7.2.2 Extracting Required Fields for Data Warehouse Star-Schema.....	180
7.2.3 Applying ETL Functionalities Using the Traditional ETL Tool	181
7.2.4 SOA-based ETL Prototype	183
7.2.5 Goals Achieved.....	184
7.3 Case Study 3: Applying ETL Functionalities on Professionals Information Technology (PIT) Company using Traditional and New ETL Tools	188
7.3.1 ETL Business Needs	189
7.3.2 Extracting Required Fields for Data Warehouse Star-Schema.....	189
7.3.3 Applying ETL Functionalities Using the Traditional ETL Tool	190
7.3.4 SOA-based ETL Prototype	192
7.3.5 Goals Achieved.....	193
7.4 Conclusion	196
CHAPTER EIGHT CONCLUSION AND FUTURE WORK.....	197
8.1 Conclusion	197
8.1.1 Problems of Current ETL Framework	198
8.1.2 Proposing the New ETL Framework	199
8.1.3 Defining the New ETL Framework	200
8.1.4 Validating the New ETL Framework.....	200

List of Tables

Table 2.1: Component Coupling features of Some ETL Tools	31
Table 2.2: Compatibility Issues of Some Popular Commercial ETL Tools.....	35
Table 2.3: Administration Capabilities of Industrial ETL Tools	39
Table 3.1: Distributed Technologies based on Components Distribution and Interoperability	54
Table 3.2: Distributed Technologies based on Components Portability, Extensibility, and Legacy Compatibility.....	55
Table 4.1: DSA Methodology Phases	70
Table 4.2: Types of Case Study Evidence (Tellis, 1997).....	73
Table 4.3: Research Strategy Based on DSA.....	79
Table 4.4(Part A): Details of the Case Studies based on Case Study Design	91
Table 4.4 (Cont., Part B): Details of the Case Studies based on Case Study Design	92
Table 5.1: EBNF Symbols Summary (ISO, 1996; Yong Xia, 2002; Gargantini, 2007).....	120
Table 5.2: Problems Satisfaction	130
Table 5.3: Solutions Satisfaction	130
Table 5.4: Advantages Satisfaction.....	131
Table 6.1: Time Deference between Fragmented and Un-Fragmented Data for Report Generation.....	162
Table 7.1: Summary Report for 2010 Electricity Blackouts.....	173
Table 7.2: Observation Results Checklist	178
Table 7.3: Summary Report for Students' Performance in Java II for Feb-June, 2011 Semester.....	182
Table 7.4: Observation Results Checklist	188
Table 7.5: Summary Report for Trainers' Performance in Training IT Courses.....	191
Table 7.6: Observation Results Checklist	196

List of Figures

Figure 1.1: Research Strategy	13
Figure 2.1: General Data Warehouse Components (Kimball & Caserta, 2004)	21
Figure 2.2: Traditional ETL Framework.....	23
Figure 3.1: SOA Parts (Barai et al., 2008)	53
Figure 3.2: N-Tier Architecture (Armstrong et al., 2004).....	58
Figure 3.3: Data Portability Feature in XML (Barai et al., 2008).....	62
Figure 4.1: Flow Chart of the Methodology Applied in this Research	88
Figure 4.2: Case Study Design.....	90
Figure 5.1: A Conceptual Framework for Interoperable Distributed ETL Components	95
Figure 5.2: Flow Diagram for Steps to Consume an ETL Service by a Client	97
Figure 5.3: ETL Composition Architecture Adopted From (Salter & Jennings, 2008).....	114
Figure 5.4: A Framework for Interoperable Distributed ETL Components with Classified - Fragmentation	116
Figure 5.5: Meta-Model for Interoperable and Distributed ETL Framework Components Based on SOA.....	119
Figure 5.6: GlassFish Server and Containers (Armstrong et al., 2004)	128
Figure 6.1: CLINIC Database	139
Figure 6.2: EXTRACT TEMP STORAGE Database	140
Figure 6.3: TRANSFORM TEMP STORAGE Database	140
Figure 6.4: CLASSIFICATION Database	141
Figure 6.5: LOAD Database	142
Figure 6.6: Class Diagram of ETL Components.....	144
Figure 6.7: config.txt (JDBC Connection Variables).....	145
Figure 6.8: Extraction Sequence Diagram	146
Figure 6.9: Transformation Sequence Diagram	147
Figure 6.10: Classified-Fragmentation Sequence Diagram	149
Figure 6.11: Transformation Sequence Diagram	152
Figure 6.12: Design of the BPEL Orchestration Point.....	156
Figure 6.13: Design of the Composite Application	158
Figure 6.14: The Main Web Interface of the SOA-based ETL Prototype	159
Figure 6.15: GlassFish Tester Result for the Extract Web Service.....	160
Figure 6.16: A Statistical Report Generated from a Clinical DW Repository	162

Figure 6.17: End to End Test Case Input File (input.xml).....	164
Figure 6.18: Auto generated End to End Test Case Output File (output.xml).....	164
Figure 6.19: Sample Data before Executing the Transformation Component	165
Figure 6.20: Sample Data after Executing the Transformation Component	166
Figure 7.1: GlassFish ESB Based on NetBeans IDE for Managing the SOA-based ETL Prototype	168
Figure 7.2: First Step in Executing Traditional ETL Tools	169
Figure 7.3: Second Step in Executing Traditional ETL Tools.....	169
Figure 7.4: Third Step in Executing Traditional ETL Tools.....	169
Figure 7.5: Sample Partial Source PEC DB Schema	171
Figure 7.6: Screenshot of Auto-generated Data in Turbine_transaction_history Table.....	171
Figure 7.7: Screenshot of Auto-generated Data in Transaction_types Table	171
Figure 7.8: Star Schema of the Fact and Dimension Tables Extracted from PEC Database	172
Figure 7.9: Graph Report for 2010 Electricity Blackouts	174
Figure 7.10: A Sample of the Auto Generated Data of the Fact Table (elec_brkout) of the Star Schema Explored in Figure 7.8	176
Figure 7.11: The Data of the Dimension Table (months) of the Star Schema Explored in Figure 7.8	176
Figure 7.12: The Data of the Dimension Table (periods) of the Star Schema Explored in Figure 7.8	177
Figure 7.13: The Data of the Dimension Table (turbines) of the Star Schema Explored in Figure 7.8	177
Figure 7.14: The Data of the Dimension Table (periods) of the Star Schema Explored in Figure 7.8 (after executing the translation component)	177
Figure 7.15: Sample Partial Source DB Schema of LUCT.....	180
Figure 7.16: Star Schema of the Fact and Dimension Tables Extracted from LUCT Database	181
Figure 7.17: Summary Report for Students' Performance in Java II for Feb-June, 2011 Semester	183
Figure 7.18: A Sample of the Auto Generated Data of the Fact Table (student_performance) of the Star Schema Explored in Figure 7.16	185
Figure 7.19: The Data of the Dimension Table (marks) of the Star Schema Explored in Figure 7.16	186
Figure 7.20: The Data of the Dimension Table (attendance) of the Star Schema Explored in Figure 7.16	186

Figure 7.21: The Data of the Dimension Table (student_status) of the Star Schema Explored in Figure 7.16	186
Figure 7.22: The Data of the Dimension Table (gender) of the Star Schema Explored in Figure 7.16	186
Figure 7.23: The Data of the Dimension Table (finalexam) of the Star Schema Explored in Figure 7.16	187
Figure 7.24: Sample Partial Source DB schema of PIT	189
Figure 7.25: Star Schema of the Fact and Dimension Tables Extracted from PIT Database	190
Figure 7.26: Graph Report for Trainers' Performance in Training IT Courses.	191
Figure 7.27: A Sample of the Auto Generated Data of the Fact Table (trainer_evaluation) of the Star Schema Explored in Figure 7.25	194
Figure 7.28: The Data of the Dimension Table (trainer_details) of the Star Schema Explored in Figure 7.25	194
Figure 7.29: The Data of the Dimension Table (performance_level) of the Star Schema Explored in Figure 7.25	194

List of Appendices

Appendix A Details of the Prototype Design..... 220

Appendix B Source Code of the Prototype 275

Appendix C Questionnaires as Deliverables of Structured Interviews Done with Industry
Experts 301

Appendix D Case Studies User Manual and Parts of the Source Code 332

List of Abbreviations

Acronym	Description
BI	Business Intelligence
DW	Data Warehouse
ETL	Extraction-Transformation-Loading
SOA	Service Oriented Architecture
J2EE	Java 2 Enterprise Edition
XML	eXtensible Markup Language
SOAP	Simple Object Access Protocol
DSA	Design Science Approach
WSDL	Web Services Description Language
LAN	Local Area Network
DSA	Data Staging Area
OMG	Object Management Group
CORBA	Common Object Request Broker Architecture
RMI	Remote Method Invocation
RPC	Remote Procedure Call
DCOM	Distributed Component Object Model
HTML	Hyper Text Markup Language
JSP	Java Server Pages
EJB	Enterprise Java Beans
DBMS	Database Management System
EIS	Enterprise Information System
HTTP	Hyper Text Transfer Protocol
JMS	Java Messaging Service
OLAP	online analytical processing
IT	Information Technology
OWB	Oracle Warehouse Builder
API	Application Programming Interface

BODI	Business Objects Data Integrator
CAL	client access license
SSIS	SQL Server Integration Services
GUI	Graphical User Interface
BIDS	Business Intelligence Development Studio
DCOM	Distributed Component Object Model
SC	Service Container
JDBC	Java Database Connectivity
BPEL	Business Process Execution Language

CHAPTER ONE

INTRODUCTION

This chapter presents the background and outlines the motivation of the research. This is followed by the research problems, the research question and the research objectives. Moreover, the research strategy is discussed and the scope of the research is argued. Furthermore, the research contribution is highlighted, the organization of the thesis is explored and chapter conclusions are presented.

1.1 Research Background

Data warehouses (DW) have become a main component of the corporate information system architecture, in which it plays a major role in building decision support systems (Vassiliadis *et al.*, 2002; Darmont *et al.*, 2005; Wrembel & Koncilia, 2007). By collecting data from a variety of internal and external sources, data warehouses use the transformation functionality which is a function in the ETL framework (explained in Chapter Two) to provide homogeneous information for analysis and reporting tasks (Wrembel & Koncilia, 2007; Bala *et al.*, 2009).

The uses of data warehousing products and services have been increasing over the years by industry as well as the development of the related technologies (Sen & Sinha, 2007; Wrembel & Koncilia, 2007). Furthermore, within the last decade, data warehouse field has made a very important step by moving from simple centralized repositories to a platform for data integration and analysis (Vitt *et al.*, 2002; Mundy *et al.*, 2006; Watson & Wixom, 2007; Xi & Hongfeng, 2009). This move is pushing the success of the whole Business Intelligence (BI) field.

BI refers to techniques used in identifying, extracting and analyzing business data. These techniques provide historical, current and predictive views of business operations. Common functions of BI technologies are reporting, online analytical processing, analytics, data mining, business performance management, benchmarking, text mining and predictive analytics. These functions aim to support better business decision-making (Almeida *et al.*, 1999; Vitt *et al.*, 2002; Watson & Wixom, 2007; Tam, 2010). Furthermore, BI which is highly dependent on data warehousing; is successfully used together with warehouses in many industries including: healthcare, manufacturing, financial services, education, telecommunication, population, and other fields (Almeida *et al.*, 1999; Vitt *et al.*, 2002; Darmont & Boussaid, 2006; Mundy *et al.*, 2006; Tam, 2010).

Research problems related to creating, maintaining, and using data warehouse technology are somewhat similar to those specific for database systems. In other words, a data warehouse can be considered as a large database system with additional functionalities (Almeida *et al.*, 1999; Massachusetts, 2008; Silvers, 2008).

General database problems of index selection, materialized view maintenance, data integration, and query optimization have been reactivated in warehousing research (Vassiliadis *et al.*, 2002; Vassiliadis *et al.*, 2005; Tziovara *et al.*, 2007; Bâra *et al.*, 2008; Dessloch *et al.*, 2008; Siqueira *et al.*, 2009). On the other hand, some research problems are specific to data warehousing such as data acquisition, data cleaning, data warehouse refreshment, evolution of data warehouse schema, multidimensional and parallel query optimization, conceptual modeling for the data warehouses, data quality management, and data extraction, transformation and loading (ETL)

enhancements (McCabe & Grossman, 1996; Bruckner *et al.*, 2002; Vassiliadis *et al.*, 2002; Du & Wong, 2004; Wehrle *et al.*, 2005; Zhang *et al.*, 2006; Sahama & Croll, 2007; Wehrle *et al.*, 2007; Santos & Bernardino, 2008; Mahboubi & Darmont, 2009).

ETL processes are meant to extract, transform and load the data into data warehouse for decision making (Wrembel & Koncilia, 2007). Effective ETL processes represent a major success factor for data warehouse projects and can absorb up to 80 percent of the time spent on any warehousing project (Vassiliadis *et al.*, 2002). These ETL processes are important because of the valuable functionalities that are performed using them. For example, they remove mistakes and correct missing data, provide documented measures of confidence in data, capture the flow of transactional data for safekeeping, adjust data from multiple sources to be used together, and structure data to be usable by end-user tools (Trujillo & Lujnмора, 2003; Kimball & Caserta, 2004; Tziovara *et al.*, 2007; Liao *et al.*, 2008).

Developing or using ETL is both a simple and complicated task. While the basic role of ETL is simply to get data out of the source and load it into the data warehouse (Sellis, 2006; Sen & Sinha, 2007; Morris *et al.*, 2008; Mrunalini *et al.*, 2009; Muñoz *et al.*, 2009; Simitsis *et al.*, 2009; Simitsis *et al.*, 2010), the development of ETL functionalities to incorporate additional requirements may break the ETL tasks into many little sub-cases, depending on data sources, business rules, existing software and destination reporting applications (Vitt *et al.*, 2002; Kimball & Caserta, 2004; Simitsis *et al.*, 2005). The development of these types of special requirements in current ETL tools is an uphill task to combine and reuse the broken little sub-cases

and to keep perspective on the simple overall mission of the ETL system (Kimball & Caserta, 2004; Kshemkalyani & Singhal, 2008).

ETL processes perform at least three specific functionalities, and these are focused around the movement of data from one place or system to another (Sellis, 2006; Morris *et al.*, 2008; Mrunalini *et al.*, 2009; Shaikh *et al.*, 2010). These functionalities are: (1) the first function generally is to read data from an input source (file, relational table, or message queue); (2) to pass the stream of information through a process to modify, enhance, or eliminate data elements based on the instructions of the job; and (3) to take the resultant data and store it back to a file or relational table. These three steps are known as extraction, transformation and loading, respectively.

1.2 Research Motivation

The age of information technology (IT) is erasing the boundaries of cities, states, and countries. The success of IT applications depends on how remote distributed subsystems interoperate among each other's (Stonebraker & Hellerstein, 2001; Tanenbaum & Van Steen, 2002; Blair *et al.*, 2009; Li *et al.*, 2010). The biggest challenge is in aligning these independent subsystems into components that can interoperate across the enterprise branches through many locations (Coulouris *et al.*, 2001; Kshemkalyani & Singhal, 2008; Li *et al.*, 2010).

Data warehouses have occupied a big portion of those IT solutions to provide information for analytical processing, decision making, mining tools and other related technologies (Almeida *et al.*, 1999; Bruckner *et al.*, 2002; Du & Wong, 2004;

Inmon, 2005; Sahama & Croll, 2007; Santos & Bernardino, 2008; Mahboubi & Darmont, 2009).

Extract, Transform, and Load processes play a central role in the data warehouse solutions, and ETL is considered as the core component of a successful data warehouse system (Trujillo & Lujnmora, 2003; Muñoz *et al.*, 2009). ETL physically integrates data from multiple heterogeneous sources in a central repository referred to as data warehouse (Trujillo & Lujnmora, 2003; Kimball & Caserta, 2004; Muñoz *et al.*, 2009).

According to (Vassiliadis *et al.*, 2002), ETL consumes more than 60% of the data warehouse development effort. Furthermore, ETL and Data Cleaning tools are estimated to cost at least one third of the effort and expenses in the budget of the data warehouse projects, and this number can rise up to 80% of the development time in a data warehouse project. In addition, the ETL processes cost up to 55% of the total costs of data warehouse runtime.

Due to its importance and high cost, many research projects are carried out to enhance the ETL framework. Some of these projects in the last few years have concentrated on Real-Time data warehouse to solve the periodic Extraction, Transformation, and Loading problems (Bruckner *et al.*, 2002; Nelson & Wright, 2005; Abrahim, 2007; Dou *et al.*, 2008; Santos & Bernardino, 2008). On the other hand, there is no sufficient research works that have been carried out to bridge the gap in this field regarding ETL components distribution and interoperability (Trujillo & Lujnmora, 2003; Kimball & Caserta, 2004; Simitsis *et al.*, 2005; Tziovara *et al.*, 2007; Wu *et al.*, 2007; Zhang & Wang, 2008; Skoutas *et al.*, 2009). That is because

all of the previous research works view ETL as a tightly coupled software architecture rather than isolated components (Trujillo & Lujnmore, 2003; Kimball & Caserta, 2004; Simitsis *et al.*, 2005; Tziovara *et al.*, 2007; Zhang & Wang, 2008; Skoutas *et al.*, 2009), this is because the focus of these research works were on achieving other requirements of the ETL framework rather than involving distribution and interoperability features.

Data warehouse gets its data from many sources available in different separated locations (Kimball & Caserta, 2004). Each data source needs a complete ETL tool to sometimes do the extraction only (Henry *et al.*, 2005; Dung & Kameyama, 2007; Agrawal *et al.*, 2008; Mrunalini *et al.*, 2009; Suzumura *et al.*, 2010). Since an ETL tool is only available as one tightly coupled software (Kimball & Caserta, 2004; Apache, 2010; IBM, 2010; Microsoft, 2010; Oracle, 2010; SAS, 2010), there is no option other than installing all the ETL features together in each location, while the transformation and loading features are sometimes redundant for these sources. These two features would be necessary only in the location of the data warehouse destination and not the sources. Therefore, distributing ETL into loosely coupled and interoperable components can play a central role in solving this complication, and result in eliminating the redundant usage of many ETL licenses for the same project (Wu *et al.*, 2007). In addition, it can enable reusability, centralized ETL administration, ETL components portability, and ETL ease of use (Wu *et al.*, 2007).

Data warehouses normally include huge amounts of data that leads to slow report generation due to this massive amount of data (Almeida *et al.*, 1999; Vitt *et al.*, 2002; Tam, 2010). There are some hardware based solutions to this issue, but there is

lack of research regarding extending the current ETL framework by an extra component to solve this complication; due to the tightly-coupled disadvantage of the current ETL framework (Vassiliadis *et al.*, 2002; Trujillo & Lujnmora, 2003; Kimball & Caserta, 2004; Vassiliadis *et al.*, 2005; Tziovara *et al.*, 2007; Zhang & Wang, 2008; Skoutas *et al.*, 2009).

1.3 Problem Statement

Data warehouses are complex systems employed to integrate the organization's data from several distributed and heterogeneous sources (Almeida *et al.*, 1999; Ault, 2003; Kimball & Caserta, 2004; Mundy *et al.*, 2006; Silvers, 2008). The heterogeneous sources are located in different locations far from each others. Each location has its own specific infrastructure for the systems running in that location. For example, it has its own operating system and deployment infrastructure such as .NET, J2EE, or IBM mainframe (Apache, 2010; IBM, 2010; Microsoft, 2010; Oracle, 2010; SAS, 2010). Furthermore, each of these infrastructures needs a special ETL tool to be compatible with. In addition, each data source needs a complete ETL tool to be installed in the same location of the source (Kimball & Caserta, 2004), while sometimes only the Extract function is needed to extract data from this source (Kimball & Caserta, 2004). This results in a problem of an increase in the ETL licenses needed for a DW project and an increase in the complexity for the ETL user due to the redundant processes (Wu *et al.*, 2007).

Sometimes, due to the complexity, long learning curve of the available ETL tools, and difficulty to achieve some extensibility in terms of additional functionalities; some organizations prefer to turn to in-house development to perform ETL tasks

(Kimball & Caserta, 2004; Inmon, 2005; Temenos, 2005), which increases the cost and effort of the data warehouse project (Wehrle *et al.*, 2007). Furthermore, tightly coupled components of a software require teams of architects and designers to untangle the complex implications of change in the support of new business requirements or system enhancements (Sneed, 2006; Wu *et al.*, 2007; Tam, 2010). As a result, tightly coupled components cause problems in terms of cost, maintenance, enhancement, and reusability of the ETL components (Wu *et al.*, 2007). This leads to the necessity that the architecture of the software framework to consider the component coupling factor and how much loosely the components should be coupled.

According to (Kimball & Caserta, 2004; Wrembel & Koncilia, 2007; Wu *et al.*, 2007), the current ETL framework lacks of components flexibility and loose coupling. This results in complications to add new components to the ETL tools; to support special business needs (Kimball & Caserta, 2004; Newcomer & Lomow, 2004; Kshemkalyani & Singhal, 2008). For instance, although data warehouses provide an appropriate infrastructure for efficient querying, reporting, mining, and other advanced analysis techniques (Inmon, 2005; Silvers, 2008), the complexity of data warehouse environments especially the ETL framework is rising every day, and data volumes are growing at a significant pace, which makes report generation relatively slow due to the massive amount of data (Inmon, 2005). According to (Wehrle *et al.*, 2007), data warehouse repositories based on one central server often suffer from either storage or computing bottlenecks, especially when complex aggregates need to be stored permanently or computed on demand. (Wehrle *et al.*,

2007) refers to a high cost solution to the massive data problem, which is cluster-type systems with large numbers of worker nodes connected through high-speed LAN.

In addition to the high expenses, integrating existing resources from several distant sites using this solution; needs a system that efficiently organizes these resources in a transparent manner, that at times is a challenge to implement (Pentaho, 2006; Pentaho, 2009). Some implementations of the ETL frameworks like “Pentaho Open Source Business Intelligence” include a fragmentation feature like Pentaho “Partitioning” (Pentaho, 2006; Pentaho, 2009). However, this feature does not classify data based on certain conditions to fulfill specific business needs. Furthermore, this fragmentation aims mainly to enable the fact and the dimension tables in the data warehouse to be separated among a cluster of servers. This belongs to a physical (hardware) solution of the performance problem, and such solution is outside the scope of this research. Therefore, an extensible ETL framework with loosely coupled components can resolve this complication, because, a specific component can easily be added as an extension to the framework to resolve the performance problem (Newcomer & Lomow, 2004; Wu *et al.*, 2007).

Administration of ETL tools in many data source locations for the same project to extract data from many different sources, requires additional administration, communication and maintenance effort (Wu *et al.*, 2007). This is a problem resulted from the reality that the administrators are often different persons from one location to another and they could use different ETL tools and do different configurations to these tools (Kimball & Caserta, 2004; Albrecht & Naumann, 2008). Furthermore, in

the current ETL framework, there are impediments to include ETL as a part of a complete portal that manages the whole DW project because of complexity in the current ETL framework to communicate with other components of the portal (Wu *et al.*, 2007).

Therefore, there are gaps in the current ETL framework and these are related to: distribution and interoperability of the ETL components. These gaps lead to problems of the current ETL framework. These problems include: the complexity of extending the ETL tools to suit special business needs, the ETL administration complexity, and an increase of effort needed to implement a DW project. In addition, the distribution and interoperability gaps lead to: impediments regarding ETL compatibility with different administrator environments, an increase of the cost to implement a DW project due to the increase of the number of ETL licenses needed, and an extra effort needed to develop and use the ETL processes. Furthermore, the same gaps lead to a redundancy problem of including all the ETL features in every ETL administrator location due to the tightly coupled architecture of the available ETL framework.

As such, defining a conceptual framework for ETL that includes the features of components distribution and interoperability addresses the problems highlighted in this section.

1.4 Research Questions

As identified in section 1.3, there are problems due to the absence of components distribution and interoperability of the current ETL frameworks. Therefore, the main research question of this study is:

“Can a conceptual framework for ETL that includes the features of component distribution and interoperability be defined to enhance the current tightly coupled ETL framework?”

The main research question can be divided into three sub questions, which are:

- i. What are the problems of the current ETL framework that result from the absence of components distribution and interoperability?
- ii. How can the problems of the current ETL framework be tackled for defining distributed and interoperable components for ETL framework?
- iii. How to define, test and evaluate the new ETL framework?

1.5 Research Objectives

The overall objective of this research is to define a conceptual framework for interoperable distributed ETL components. This framework is an enhancement to the current ETL framework in terms of components distribution and interoperability.

In particular, the research objectives are:

- i. to identify problems of the current ETL framework due to the absence of component distribution and interoperability.

- ii. to define components distributable and interoperable ETL conceptual framework.
- iii. to demonstrate the applicability of the proposed ETL framework by the development of ETL prototype.
- iv. to test and evaluate the distribution and interoperability of the prototype that validates the new ETL framework.

Each of the research objectives are achieved according to the research strategy (following the phases of the research methodology) in section 1.6.

1.6 Research Strategy

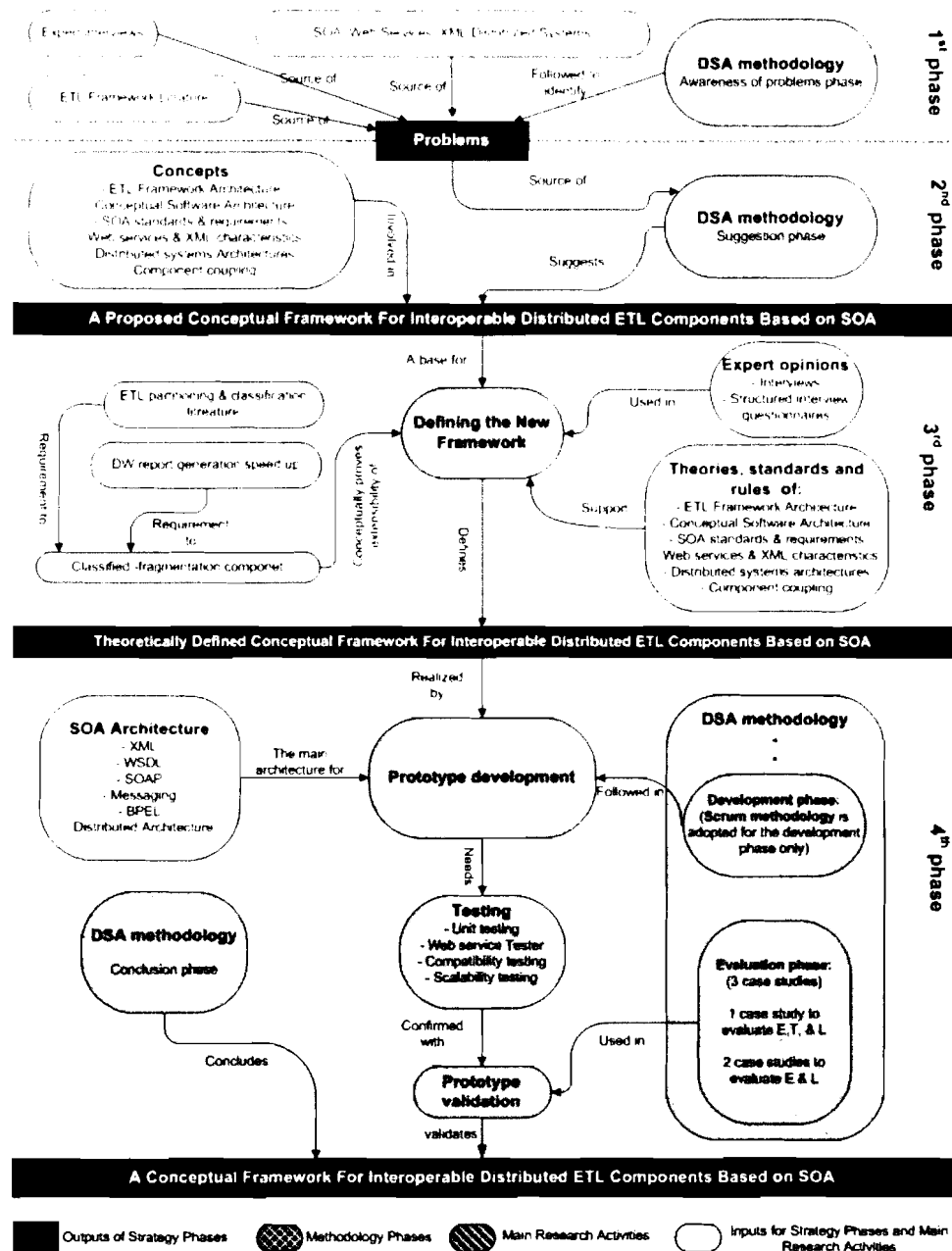


Figure 1.1: Research Strategy

The research strategy consists of four phases to achieve the research objectives and answer the research question. Figure 1.1 illustrates the research strategy. The 1st

phase answer the question of: *What are the problems of the current ETL framework, which result from the absence of components distribution and interoperability?* The discussion of ETL literature, together with exploring the advantages of involving SOA, XML, distributed architecture and other related technologies are important to identify the available problems, which are resultant from the absence of components distribution and interoperability in the current ETL framework. Upon completion of this phase, a complete awareness of the problems is achieved.

The 2nd phase starts with analyzing the requirements of proposing the research work according to the suggestion phase of the DSA methodology. SOA framework plays a central role in resolving components distribution and interoperability issues of the newly defined framework. The research problems motivate for more research to bridge the gaps that have been left in the ETL field regarding distribution and interoperability. A literature about distributed systems, web services, SOA and software architecture is essential in determining the methods and standards that help in bridging the gaps. After determining the appropriate methods and following them in bridging the gaps; the resultant partial solutions are combined together to establish an initial conceptual framework for interoperable distributed ETL components. This framework is the output of the 2nd phase, and is also referred in the thesis as the “new ETL framework” or “restructured ETL framework”.

The 3rd phase concentrates on defining the complete theoretical framework before realizing it by a prototype. Defining the framework depended on theories, standards and rules of ETL and SOA and other concepts related to SOA. In addition to that, expert opinions through structured interviews are utilized in defining the framework.

The 4th phase focuses on testing and validating the new framework. Specific methods that comply with the ETL and SOA frameworks are used to define, test and verify the framework. The framework is realized by developing a prototype for testing and validation purposes. Analysis, design, development, and deployment tools are used for the implementation of that prototype, and the prototype is evaluated using three case studies. In addition to evaluating the distribution and interoperability of the framework components, one case study is used to evaluate each of Extraction, Transformation and Loading components, while the other two case studies are used in evaluating Extraction and Loading components. The three case studies are explored in details in chapter 4 and chapter 7. Upon completion of this phase, the goal of the research is achieved and the conceptual framework has been defined, verified and validated.

1.7 Scope of the Research

This research concentrates on bridging the gaps of the current ETL Framework regarding components distribution and interoperability. Therefore, the scope of the research includes: ETL framework and processes; distribution concept which is limited to software techniques and does not use hardware techniques, as the hardware based techniques are already explored by previous researches (Wehrle *et al.*, 2007). Core architectures include 1-tier, 2-tier, 3-tier and N-tier; and applying distribution and interoperability techniques to the new framework is limited to SOA, web services, XML, and software distribution standards. The classification and fragmentation regarding types of data in data warehouse for the extended classified-fragmentation component is limited to text, image or/and video. All of these scopes

are fully or partially used in the research. However, the focus of this research is generally to define a conceptual framework for interoperable distributed ETL components.

1.8 Contributions

The high-level goal of this research is to define a new ETL framework for Interoperable Distributed ETL Components based on Service Oriented Architecture. Therefore, this research contributes towards the field of ETL by adding the distribution and interoperability concepts to the ETL framework. Furthermore, this research adds contributions towards the area of data warehousing and business intelligence because ETL is a core concept in this area, particularly, in the area of data warehousing and business intelligence that covers the design, implementation and usage of ETL. This research:

- Contributes to the ETL field by involving the distribution concept among the ETL framework components, which enables the loose coupling among these ETL components. In addition, this research has provided interoperability in the interaction between clients (ETL administrators) and the loosely-coupled distributed ETL components by using SOA as the interaction architecture among the framework components.
- Contributes to the data warehouse and business intelligence area by providing distributable interoperable ETL framework to be involved in the design and implementation of data warehouse and business intelligence projects.

- Contributes to the ETL vendors, since it enables them to develop new releases of ETL tools including the distribution and interoperability features based on the new ETL framework. Furthermore, the new framework simplifies the process of extending and adding new components to the vendor's ETL tools, and reduces the cost and effort of developing ETL tools that are compatible with legacy systems. In addition, the new framework simplifies the reusability of ETL components since it enables ETL developers in DW industry to adopt the code of an existing ETL component developed by them or by any other ETL vendor who follows the new framework specifications, and then reuse it to meet new ETL business requirements. This reuse results in a huge savings in ETL tool development cost and time.
- Contributes to the ETL administrators (users) by simplifying the administration process of a DW project. The ETL administration can be centralized on one server because the ETL components can be deployed on one portal server. Therefore, the administrator focuses on a central unique ETL tool instead of administering many ETL tools at many geographical locations. Furthermore, the new framework contributes to the ETL administrators by eliminating the requirements and settings of the user machine (ETL administrator PC) such as operating system requirements and compatibility requirements, because the concepts of the new ETL framework allow ETL vendors to develop ETL components that can be deployed as parts of a business intelligence portal and the client can execute the ETL functionalities through the browser. However, ETL tools developed using the

traditional ETL framework needs to be installed on the user machines (normally ETL administrator PCs), which requires special operating systems and special configurations and settings on the client machines.

- Contributes to the ETL customers (Companies that buy non-free ETL tools to implement DW projects) by reducing the number of licenses needed for implementing a DW project. Therefore, instead of having a license for every data source administrator, it is sufficient to have only one tool that can be deployed as components of a Business Intelligence portal and then every administrator can have an access to these components through his account on the portal.

1.9 Thesis Organization

The thesis consists of eight chapters; it starts with Chapter 1 that discusses the research background, motivation, problems, questions, objectives, strategy, scope and contribution. Then, Chapter 2 presents the literature of data warehouse, especially ETL framework, followed by discussion on the SOA framework which is very essential to restructure the ETL framework in Chapter 3. Chapter 4 discusses the research methodology that utilizes suitable methods for executing this research. This is followed by Chapter 5 that conceptually defines the theoretical framework. Discussion on the prototype is highlighted in Chapter 6, while Chapter 7 presents the evaluation of the prototype. Finally, Chapter 8 concludes the research and presents the future works.

1.10 Conclusion

This chapter has introduced the research background, presented the motivation of the research and highlighted the research problems, objectives, scope and contribution. Chapter 2 critically discusses the literature related to the traditional (current) ETL framework.

CHAPTER TWO

EXTRACTION, TRANSFORMATION, AND LOADING IN DATA WAREHOUSING

This chapter discusses about data warehousing, the ETL framework components and commercial ETL tools. It also highlights the ETL components coupling, distribution and compatibility. In addition, it argues the ETL extensibility, scalability, administration and licensing.

2.1 Data Warehouse

A data warehouse is a central repository for all or significant parts of the data that an enterprise's various business systems collect (Inmon, 2005) and it has become a main component of the corporate information system architecture (Almeida *et al.*, 1999; Holzer *et al.*, 1999; Massachusetts, 2008; Silvers, 2008). A data warehouse model can be classified into two parts, back room and front room. As shown in Figure 2.1, these are physically, logically, and administratively separated. In other words, the back and front rooms are on different machines. They follow different data structures, and are managed by different IT specialists (Vassiliadis *et al.*, 2002; Kimball & Caserta, 2004; Tziovara *et al.*, 2007). Data management for data warehouses involves acquiring data, transforming and delivering that data to the query-friendly front room. No query services are provided in the back room. Data access is prohibited in the back room, and therefore, the front room is dedicated only for this purpose (Kimball & Caserta, 2004; Santos & Bernardino, 2008; Zhu *et al.*, 2008a).

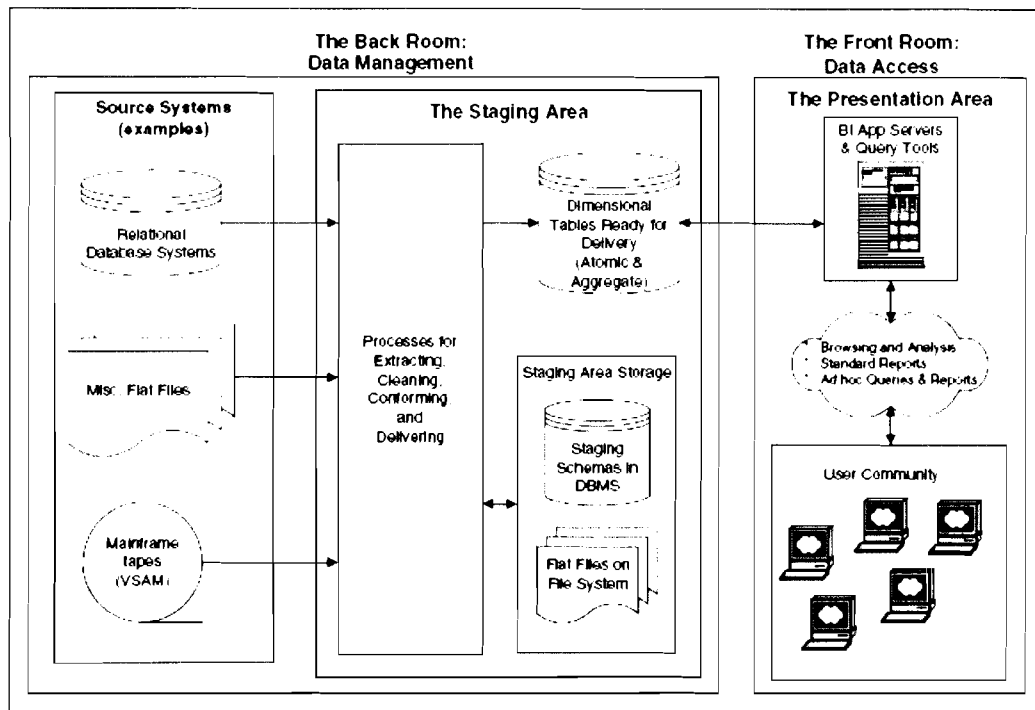


Figure 2.1: General Data Warehouse Components (Kimball & Caserta, 2004)

The staging area is a back-room facility, where the data is placed after it is extracted from the source systems, cleaned, manipulated, and prepared to be loaded to the presentation layer of the data warehouse. Any meta-data generated by the ETL processes that is useful to end users must be from the back room and is offered in the presentation area of the data warehouse (Kimball & Caserta, 2004; Microsoft, 2009; Oracle, 2009; Pentaho, 2009; SAS, 2010).

In this chapter, the literature regarding ETL research works to highlight the gaps of these works by showing where recent stages in this area have reached, and are critically discussed.

2.2 Brief Discussed Overview of traditional ETL Framework Components

A framework can be described as some concepts related to each others in some relations. The framework can contain many models in which the model constitutes a subset of a framework (Darmont *et al.*, 2005; Jerstad *et al.*, 2005; Zhang & Wang, 2008). Many research works such as (Vara *et al.*, 2005; Zhang & Gracanin, 2008) define the output of an SOA based research as a framework. In addition, other research works such as (Roy *et al.*, 2008; Zhang & Wang, 2008; Thomsen & Pedersen, 2009; Xi & Hongfeng, 2009) describe the ETL processes together as an ETL framework because the ETL processes together fulfil the requirements to be defined as a framework due to the availability of the concepts represented by the ETL components and the relationships among these concepts.

The data extraction, transformation, and loading are the three major components of the traditional ETL framework (Zhang & Wang, 2008). Data extraction is an ongoing function carried out at very frequent intervals and extracts data from many heterogenous data sources (Kimball & Caserta, 2004). There are many types of data transformation in a data warehouse with many different tasks. It is not just a field-to-field conversion but there are many types of data transformation. Transformation component targets to resolve the heterogeneity of the extracted data which is caused due to the changes in data structure, format and semantics (Kimball & Caserta, 2004; Thomsen & Pedersen, 2009; Simitsis *et al.*, 2010).

The data load in a data warehouse in comparison with the load for a new operational system has some differences. For the implementation of a new operational system, the data is sometimes converted and loaded once to get the new system started.

Loading of data in a data warehouse does not cease with the initial implementation. Just like extraction and transformation, data loading is not just an initial activity to get the data warehouse started. Apart from the initial data load, the ongoing incremental data loads and the periodic full refreshes are done (Kimball & Caserta, 2004; Tziovara *et al.*, 2007; Zhang & Wang, 2008).

Based on the ETL literatures from 2002 to 2009, the traditional ETL framework has common tightly coupled functionalities (Vassiliadis *et al.*, 2002; Trujillo & Lujmora, 2003; Kimball & Caserta, 2004; Simitsis *et al.*, 2005; Iqbal & Daudpota, 2006; Tziovara *et al.*, 2007; Liao *et al.*, 2008; Skoutas *et al.*, 2009). These functionalities, concepts behind them, and relationships between them are concluded in one framework diagram as shown in Figure 2.2.

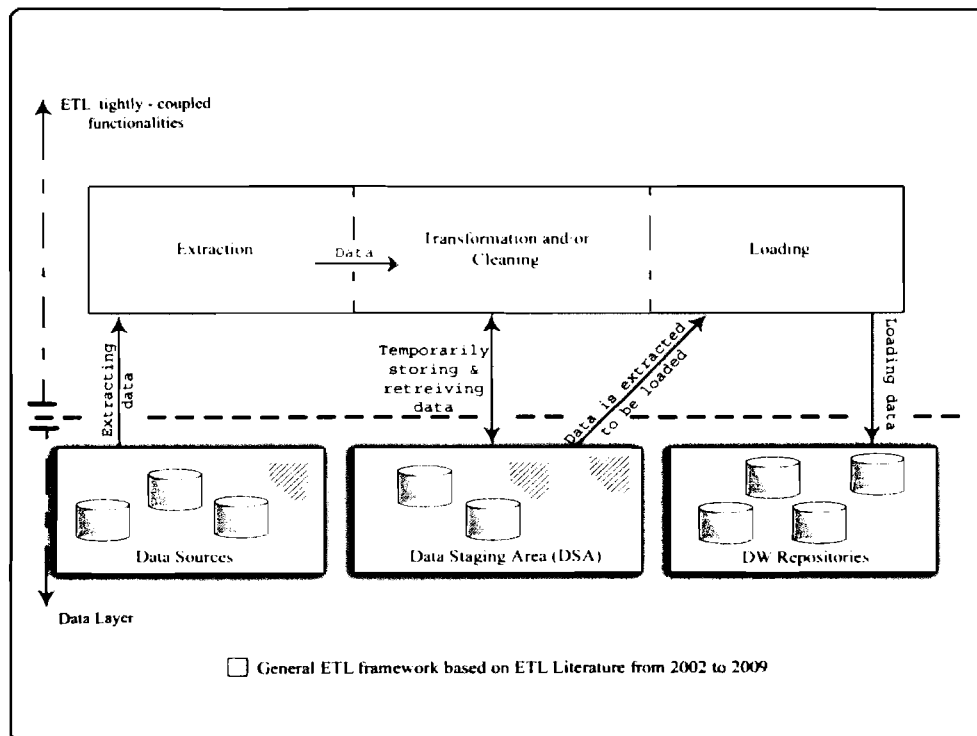


Figure 2.2: Traditional ETL Framework

In the data layer of Figure 2.2, the data stores that are involved in the overall process are depicted. On the left side, the original data providers (typically, relational databases and files) are shown. The data from these sources is extracted (as shown in the upper left part of Figure 2.2) by extraction routines. Then, this data is propagated to the Data Staging Area (DSA) where it is transformed and cleaned before being loaded to the data warehouse. The data warehouse repositories are depicted in the right part of Figure 2.2 and comprise the target data stores. Eventually, the loading of the central warehouse is performed through the loading routines depicted on the upper right part of Figure 2.2.

To have the whole abstract picture of the ETL framework, it is necessary to know the considerations taken when a vendor needs to implement an ETL tool (Kimball & Caserta, 2004). Even if the scope of this research is limited, it is necessary to understand the whole picture of the required specifications of implementing an ETL tool. Building an ETL system is constrained by some criteria and specifications to meet some business requirements such as: the data formats, the source data, the existing legacy systems, and the skills of available staffs and the needs of end users (Almeida *et al.*, 1999; Kimball & Caserta, 2004; Inmon, 2005). Two issues must be taken into consideration when building an ETL system based on the current ETL framework: the first is planning and design and the second is data flow.

- a. *Planning and design* (Almeida *et al.*, 1999; Kimball & Caserta, 2004; Inmon, 2005)

The first step in the planning and design is the understanding of the requirements and some realities. These include: business needs, security

requirements, data integration, data latency, end user delivery interfaces, available development skills, available management skills and legacy licenses. The second step is related to architecture where top decisions are made about how to build an ETL tool. These decisions may include and not limited to: hand-coded versus ETL vendor tool; batch versus streaming data flow; horizontal versus vertical task dependency; scheduler automation; exception handling; quality handling; recovery and restart. The third step in the planning and design is system implementation that may include and not limited to: hardware; software; coding practices; documentation practices and specific quality checks.

- b. *Data flow* (Almeida *et al.*, 1999; Bonifati *et al.*, 2001; Kimball & Caserta, 2004; Inmon, 2005)

The data flow is a simple generalization of the extract, transform and load scenario. The extract step includes and not limited to: reading source data models; connecting to and accessing data; scheduling the source system; capturing changed data and storing the extracted data to disk. The transform step involves and not limited to: setting column properties; enforcing structure; enforcing data and value rules; enforcing complex business rules; building a meta-data foundation to describe data quality; storing the cleaned data to disk. Eventually, the load step includes and not limited to: generating dimensions; loading dimensions tables; conforming dimensions and facts; loading text facts in dimensions and storing the delivered data to disk. There are some extra operations. These operations may include and not limited to:

scheduling; job execution; exception handling; recovery and restart; quality checking; release and support.

These specifications are considered as recommendations to ETL vendors. Some vendors apply all of the specifications, while others just pick some specifications to apply or could add additional specifications (Kimball & Caserta, 2004; Apache, 2010; IBM, 2010; Microsoft, 2010; Oracle, 2010; SAS, 2010).

2.3 Industrial ETL Tools

Industry programmers can set up ETL processes using almost any programming language, but building such processes from scratch is a complex process. Increasingly, companies are buying ETL tools to be used in data warehouse projects (Oracle, 2010). At this time, a good ETL tool should be able to communicate with many different relational databases and read the various file formats used throughout an organization (Kimball & Caserta, 2004; Apache, 2010; IBM, 2010; Microsoft, 2010; Oracle, 2010; SAS, 2010). Many ETL tools now have data profiling, data quality, meta-data and other important capabilities (Pentaho, 2009). On the other hand, none of these tools has considered the distribution and interoperability as a base for designing and developing the tools. This section briefs four popular commercial ETL tools, these are:

- a. *IBM InfoSphere DataStage* (IBM, 2010): is an ETL tool and part of the IBM Information Platforms Solutions suite and IBM InfoSphere. It exists in various editions such as the Server Edition and the Enterprise Edition. In March 2005, IBM made DataStage part of the WebSphere family as

WebSphere DataStage. In 2006, it was released with the IBM Information Server under the Information Management family. In 2008, it was renamed as InfoSphere Information Server, which has these editions: (1) Enterprise Edition that has a parallel processing architecture and parallel ETL jobs; (2) Server Edition, early DataStage versions only contained Server Jobs. DataStage 5 added Sequence Jobs and DataStage 6 added Parallel Jobs via Enterprise Edition; (3) MVS Edition that is developed on a Windows or Unix/Linux platform and transferred to the mainframe as compiled mainframe jobs; and (4) DataStage TX Edition that is designed for processing complex transactions and messages, formerly known as Mercator.

- b. *SQL Server Integration Services (SSIS)* (Microsoft, 2010): is a component of the Microsoft SQL Server database software, which is used to perform a wide range of data migration tasks. It features a fast and flexible data warehousing tool used for data extraction, transformation, and loading. It is also used to automate maintenance of SQL Server databases. The first released came together with Microsoft SQL Server 2005, it replaced data transformation services, which had been a feature of SQL Server since Version 7.0, and it is only available in the "Standard" and "Enterprise" editions and has these features: (1) the SSIS Import/Export Wizard that allows the user to create packages that move data from a data source to a destination without transformations. It can quickly move data from a variety of source types to a variety of destination types, including text files and other SQL Server instances; (2) developers can use a visual development tool based on Microsoft Visual Studio called the SQL Server Business

Intelligence Development Studio (BIDS). It allows users to edit SSIS packages using a drag-and-drop user GUI (Graphical User Interface). A scripting environment in which to write programming code is also available; (3) a connection includes the information necessary to connect to a data source. Tasks refer to the connection by its name, allowing the details of the connection to be changed or configured at runtime; (4) there are lots of tasks ranging from the file system task, which can copy or move files to the data transformation task. The data transformation task copies data and implements the ETL features of the product; (5) the precedence constraint preceding a particular task is met before that task executes. The runtime enables executing tasks in parallel if their precedence constraints allow. In addition, constraints may allow different paths of execution depending on the success or failure of other tasks; (6) a workflow is designed for a number of events in the different scopes where they occur. In this way, tasks are executed in response to happenings within the package, such as cleaning up after errors; (7) tasks can reference variables to store results, make decisions, or affect configurations; (8) a package can be saved to a file or to a store with a hierarchical namespace within an SQL Server instance, and the package content is persisted in XML; and (9) after completion, the designer allows the user to start the packages execution. On starting, the package can be debugged or monitored.

- c. *Oracle Warehouse Builder (OWB)* (Oracle, 2010): is an ETL tool produced by Oracle Corporation. It offers a graphical environment to build, manage and maintain data integration processes in business intelligence systems. The

primary use of OWB is transferring heterogeneous data sources in data warehousing and data migration from legacy systems. Furthermore, it offers capabilities for relational and dimensional modeling, data profiling and data cleaning. Oracle Warehouse Builder was first released in January 2000 (release 2.0.4). The 3i release enhanced the ETL mapping designer, then, 9i release in 2003 introduced the mapping debugger, process flow editing, integrated match/merging and name/address cleaning. The 10gR1 release was released with the 10g database, and the 10gR2 release was a big release that is able to perform a wide range of functionality from dimensional modeling to data profiling and quality. The OWB 11gR1 release was a move into the database release stack, and included the server components installed with the database.

- d. *SAS Enterprise Data Integration Server* (SAS, 2010): enables organizations to extract, transform and load data from across the enterprise to create consistent and accurate information. It has these features: (1) creation of ETL processes that are easily modified and have embedded data quality processing is enabled; (2) an easy-to-use transformation user interface that supports collaboration, reuse of processes and common meta-data; (3) single or multiple-source data acquisition, transformation, cleaning and loading to create data warehouses, data marts, or BI and analytic data stores; (4) meta-data is captured and documented throughout the data integration and transformation processes and is available for immediate reuse; (5) transformations can run on any platform with any data source; (6) transformations can be executed interactively and scheduled to run in batch at

set times or based on events; (7) easily refresh, append and update during loading; (8) optimize loading techniques with user-selectable options; and (9) database-aware loading techniques, including bulk load facilities, index and key creation, and dropping and truncating of tables.

This section has briefly listed some features available in four commercial ETL tools, while sections 2.4 to 2.9 discuss specific features in these tools related to the research scope.

2.4 Component Coupling in Current ETL Structure

Many vendors such as IBM, Microsoft, Oracle and SAS (Statistical Analysis System) have developed powerful ETL tools, which leads to suitable vendor solutions that cover a wide range of functional options to perform ETL functions. The tools can extract data from multiple sources, perform transformation functions, and do loads (Kimball & Caserta, 2004).

Extract, Transform and Load components in the current ETL tools are tightly coupled with each others, and cannot be used separately in a data warehouse environment (without the need to involve other parts of the ETL in the same environment) because of tight coupling feature of the current ETL framework (Vassiliadis *et al.*, 2002; Trujillo & Lujnmora, 2003; Kimball & Caserta, 2004; Vassiliadis *et al.*, 2005; Sellis, 2006; Liao *et al.*, 2008; Skoutas *et al.*, 2009). Despite that, some commercial ETL tools decoupled some functionalities for reusability purposes.

Table 2.1: Component Coupling features of Some ETL Tools

IBM WEBSHERE DATA STAGE	MICROSOFT INTEGRATION SERVICES	ORACLE WAREHOUSE BUILDER	SAS ETL STUDIO
Enables the reusability of some existing code through some APIs that allows the reusability of specific functions rather than complete ETL functionalities (IBM, 2010)	Has transformation packages that can only be customized while it does not allow extending the code of its functionalities (Microsoft, 2009)	Enables the reusability of legacy code as well as newly developed code, and a transformation library that has PL/SQL code, which can be reused as well, but this is done only within the scope of Oracle Warehouse Builder framework (Oracle, 2010)	Provides a comprehensive applications development environment (reusable components) that enables developers to quickly create customized applications to fit user needs, and this solution is restricted to be within the scope of SAS tools (SAS, 2010)

Table 2.1 shows some features of certain ETL vendors related to the coupling factor of the tools. As shown in Table 2.1, the four tools do not include the loose coupling features among the main ETL components. As a result of that, the three E-T-L components must be used together and cannot be used or reused separately, for

example, Transformation and Loading features are redundant at the Extraction location because sometimes only Extraction is needed at data source locations.

2.5 Components Distribution

The current ETL framework is lacking of the mentioned distribution advantages as it does not follow distribution requirements at the component level (Vassiliadis *et al.*, 2002; Trujillo & Lujnmora, 2003; Kimball & Caserta, 2004; Zhu *et al.*, 2008b; Muñoz *et al.*, 2009). Some implementations of the current ETL framework done by key industry vendors such as IBM, Oracle, Microsoft and SAS have applied some distribution features to their ETL tools, but this distribution is applied to the whole tool level rather than the component level. The following are some distribution features of some commercial ETL tools:

- IBM WebSphere Data Stage tool adopts client server architecture. There are 3 components at the Client. The DataStage Administrator, Manager and Director; each equipped with a specific set of functionalities (IBM, 2010; Tam, 2010).
- Microsoft Integration Services ETL tool (Microsoft, 2009; Microsoft, 2010) is built on client server architecture and includes four key parts:
 - a. Integration Services Service: Monitors running Integration Services packages and manages the storage of packages.
 - b. Integration Services object model: Includes native and managed application programming interface (API) for accessing Integration Services tools, command line utilities and custom applications.

- c. Integration Services runtime and Run-time executables: Saves the layout of packages, runs packages, and provides support for logging, breakpoints, configuration, connections, and transactions.
 - d. Data Flow Task: Encapsulates the data flow engine. The data flow engine provides the in-memory buffers that move data from source to destination and calls the sources that extract data from files and relational databases, the transformations that modify data, and the destinations that load data or make data available to other processes.
- Oracle Warehouse Builder is a repository-based tool for ETL and data warehousing. The basic architecture comprises two components, the design environment and the runtime environment. The design component is client based like windows platform. For running the mappings, it uses Oracle Database 10g database server (Oracle, 2009; Oracle, 2010).
 - SAS architecture is a client-server architecture that uses XML as its transport language, and provides a central repository for meta-data for the entire enterprise (SAS, 2010).

2.6 Compatibility of ETL Components with Heterogeneous Environments

“Whether a data warehouse project is successful or not mostly depends on the processes of data integration” (Oracle, 2010). The processes usually mean ETL processes. Data warehouse designers suffer from the integrity problem of heterogeneous database and heterogeneous environments because ETL is a significant challenge and a critical task performed at the early stages of a data

warehouse project (Kimball & Caserta, 2004; Agrawal *et al.*, 2008; Mahboubi & Darmont, 2009).

The problem of heterogeneity in data sources includes two parts. First is the structural heterogeneity, such as different database supplier (SQL Server and Oracle), different data type and different schema. Second is the semantic heterogeneity, which includes different naming conventions (e.g., synonyms), different representation formats (e.g., units of measurement, currencies, encodings), and different ranges of values. Both of the mentioned heterogeneity types are resolved in previous researches (Shani *et al.*, 2006; Kumari *et al.*, 2008; Mahboubi & Darmont, 2009).

On the other hand, in heterogeneous environments for ETL components, there is lack of research discussing it, especially if the components are distributed among many platforms (Wu *et al.*, 2007). This issue is importance because if the parts of ETL could be compatible with all platforms or programming environments (i.e. the Extraction component as well as any other component can be compatible with .NET, J2EE, or any other framework, without any change in the component coding), then, the ETL components will have the portability and reusability advantages (Wu *et al.*, 2007).

As a conclusion, not all ETL tools are compatible with all server environments, and each vendor's tool runs on a specific environment including specific operating system and other requirements. Table 2.2 includes some compatibility issues of some popular commercial ETL tools.

Table 2.2: Compatibility Issues of Some Popular Commercial ETL Tools

	IBM WebSphere Data Stage Tool (IBM, 2010; Tam, 2010)	MS Integration Services (Microsoft, 2009; Microsoft, 2010)	Oracle Warehouse Builder (Oracle, 2009; Oracle, 2010)	SAS (SAS, 2010)
Operating System compatibility	Compatible with all popular operating systems	Compatible with Microsoft Operating Systems (Windows family)	Compatible with all popular operating systems	Compatible with all popular operating systems
Database compatibility	Although it is compatible with all database systems, It has a high level of compatibility with DB2 database	Although it is compatible with all database systems, It has a high level of compatibility with MS SQL Server database	Although it is compatible with all database systems, It has a high level of compatibility with Oracle database	It is compatible with all database systems
Availability of Compatibility standards	No	No	No	No
Distributable & Compatible components	No	No	No	No
Compatible with legacy software systems	No	No	No	No

As shown in Table 2.2, some tools are more compatible with the vendor environments. For example, MS Integration Services is compatible with Windows

operating systems. In addition, the four tools do not have a standard way for compatibility with other software components or with legacy systems. Furthermore, the components of these tools cannot be deployed separately as loosely coupled distributed components.

2.7 Extensibility and Scalability of ETL Framework

Many companies need to add special components to ETL to solve additional problems in their business. Sometimes, this task is complex due to the ETL components that are tightly-coupled (Trujillo & Lujnmora, 2003; Kimball & Caserta, 2004; Wu *et al.*, 2007). Furthermore, due to the complexity, long learning curve of the available ETL tools, and difficulty to achieve some extensibility in terms of additional functionalities; many organizations prefer to turn to in-house development to perform the ETL tasks (Kimball & Caserta, 2004; Agrawal *et al.*, 2008; Liao *et al.*, 2008).

For example, if some organizations need to add a fragmentation component to the framework to enhance report generation speed, certainly, there are some solutions to improve query performance of data warehousing (Microsoft, 2010). Some industrial tools such as Pentaho Business Intelligence have the fragmentation/partitioning feature (Pentaho, 2006; Pentaho, 2009) to do this, however, this feature needs a technical expert to deal with it, i.e. the user needs to write sophisticated queries for the classification/fragmentation conditions. To avoid this, it is beneficial that the ETL framework includes classified-fragmentation functionality for special business needs, especially for some domains such as health-care which deals with massive amounts of historical data. Due to the limitations of the fragmentation techniques

available in the current ETL tools, it is worth that this feature exists in the ETL framework; to be available for implementation in future tools for ETL users to fragment the fact tables into many fact tables based on the data type or time. This leads to a faster report generation in the data warehouse environments as the speed of the report generation in the presentation layer of the business intelligence can be faster.

The scalability of a software can be reached when a distributed system running on a collection of a small number of machines can be easily extended to a large number of machines or use a more sophisticated software techniques to increase the processing power of the software (Du & Raghavendra, 2005). ETL designers should establish the scalability of an ETL system across the lifetime of its usage (Kimball & Caserta, 2004; Silvers, 2008). This includes understanding the volumes of data that will have to be processed within service level agreements (SAS, 2010). The time available to extract from source systems may change, which means the same amount of data may have to be processed in lesser time. Some ETL systems have to scale to process terabytes of data to update data warehouses with tens of terabytes of data. Increasing volumes of data may require designs that can scale from daily batch to multiple-day micro-batch to integration with message queues or real-time change-data capture for continuous transformation and update (Kimball & Caserta, 2004; Temenos, 2005).

Some commercial ETL tools have considered the scalability factor in the design of their tools, such as:

- IBM WebSphere Data Stage (IBM, 2010) scales well with unlimited transforms, flexible data flows, high data throughput but it lacks the components extensibility feature.
- MS Integration Services (Microsoft, 2010) runs mainly on the Microsoft Windows platform. Currently, SQL Integration Services Service is available on 32 bit but it is expected to support 64 bit in future. This provides an opportunity for providing scalability. However, it does not support components extensibility.
- Oracle Warehouse Builder (Oracle, 2010) utilize the powerful potential of Oracle 10g that provides a highly scalable solution. The ETL processes modeled in warehouse builder make use of Oracle 10g new performance enhancements specifically added for performing ETL tasks. Despite this, it does not have the extensibility feature to suit special business needs.
- SAS (SAS, 2010) is highly scalable, but lacks extensibility features.

2.8 ETL Tools Administration

This section starts with highlighting some administration features of some popular ETL tools available in the market such as IBM InfoSphere DataStage, Microsoft Integration Services, Oracle Warehouse Builder and SAS ETL Studio. Table 2.3 lists some of these administration features.

Table 2.3: Administration Capabilities of Industrial ETL Tools

IBM WEBSPHERE DATA STAGE	MICROSOFT INTEGRATION SERVICES	ORACLE WAREHOUSE BUILDER	SAS ETL STUDIO
<ul style="list-style-type: none"> ▪ Has log files that are available in a client component called director ▪ Rejected rows can be captured with the help of a rejected link available ▪ Monitor window is available to provide run-time feedback on user-selected intervals (IBM, 2010) 	<ul style="list-style-type: none"> ▪ Error rows can be logged to tables or files for offline investigation and resubmission ▪ It supports row-level error flows. An error flow can direct the data rows that contain errors to a separate component output, the error output (Microsoft, 2010) 	<ul style="list-style-type: none"> ▪ Generates log files and using runtime audit viewer and detailed information of error occurrences that can be viewed, which could help in trouble shooting. This can be done by setting breakpoint and watch ▪ Syntax error in the data is handled by the warehouse builder (Oracle, 2010) 	<ul style="list-style-type: none"> ▪ SAS built-in administration features need to be handled within the code ▪ Administration features are associated with SAS warehouse administrator ▪ It needs to enable Perl regular expressions in a data step to send debug output to the SAS log ▪ It supports exception handling with no extra effort required ▪ It can be restarted from the point of failure (SAS, 2010)

In the current ETL framework, managing/administering many ETL installations in many data source locations for the same project to extract data needs extra administration, communication and maintenance effort due to the absence of centralized administration capabilities and the administrators are often different

persons working on the same DW project and there is a different administrator from every location in the same project. This requires them to use different ETL tools and do different configurations to these tools. Furthermore, in the current ETL framework, there are impediments to include ETL as a part of a complete portal that manages the DW project because of the complexity in the current ETL framework to communicate with other components of a portal (Wu *et al.*, 2007).

2.9 ETL Tools Licensing

ETL tools provide the database administrators the power to move and manipulate large quantities of different information across disparate database platforms easily. However, these solutions are expensive, and could be beyond the budget of many organizations (Kimball & Caserta, 2004). The licensing issues are reviewed for four popular industrial ETL tools, and these are:

- IBM WebSphere Data Stage (IBM, 2010; Tam, 2010) adopts the CPU based licensing.
- MS Integration Services SQL Server (Microsoft, 2009) is available under three licensing options: (1) Processor Licensing Model, under this model, a license is required for each physical or virtual processor accessed by an operating system environment running SQL Server. This license does not require any device or user client access licenses (CALs). (2) A server license is required for each operating system environment running an instance of SQL Server, as well as a CAL for each client device that accesses a system running SQL Server. (3) A server license is required for each operating

system environment running an instance of SQL Server, as well as a CAL for each user that accesses a system running SQL Server.

- Oracle Warehouse Builder (Oracle, 2010) needs both Oracle Developer Suite license and an Oracle database enterprise edition license to run Warehouse Builder 10g.
- SAS (SAS, 2010) provides evaluation and permanent licenses with different packages, which supports multi-user environment.

2.10 Conclusion

This chapter has presented the data warehouse environment and explored the ETL framework and design including the previous works in the ETL area. It also discusses the distribution, interoperability, portability, extensibility, component-coupling, and licensing features. The discussion emphasizes both the traditional ETL framework as a common framework and tools that are based on it. Chapter 3 will discuss the concepts of SOA framework, and highlights the reasons for building the ETL framework based on it, in order to achieve the distribution and interoperability features.

CHAPTER THREE

DISTRIBUTED TECHNOLOGIES

Chapter two has discussed the ETL literature and what the current ETL framework is missing regarding: component distribution, interoperability, extensibility and loose-coupling features. This chapter discusses the distributed technologies and highlights the best technology that is capable of overcoming with these missing features. This capability will be demonstrated throughout this chapter.

3.1 Distributed Systems

A distributed system is a collection of independent entities that cooperate to solve a problem that cannot be individually solved (Coulouris *et al.*, 2001; Tanenbaum & Van Steen, 2002; Du & Raghavendra, 2005; Sutherland *et al.*, 2007; Kshemkalyani & Singhal, 2008). For computing systems, a distributed system has been characterized as a collection of computers that do not share common memory or a common physical clock, that communicate by a messages passing over a communication network, where each computer has its own memory and runs its own operating system. Typically the computers are loosely coupled and they cooperate to address a problem collectively (Wehrle *et al.*, 2005).

The distributed software can also be termed as middleware. In other words, the middleware is the distributed software that drives the distributed system, while providing transparency of heterogeneity at the platform level (Kshemkalyani & Singhal, 2008). Various primitives and calls to functions defined in various libraries of the middleware layer are embedded in the user program code. In addition, there

exist several libraries to choose from to invoke primitives for the more common functions of the middleware layer (Wehrle *et al.*, 2005). Furthermore, there are several standards such as Object Management Group's (OMG), Common Object Request Broker Architecture (CORBA), Remote Method Invocation (RMI) and the Remote Procedure Call (RPC) mechanisms for distributed computing (Coulouris *et al.*, 2001; Kshemkalyani & Singhal, 2008). Currently, deployed commercial versions of middleware often use CORBA, DCOM (distributed component object model), EJB and RMI technologies (Tanenbaum & Van Steen, 2002; Issarny *et al.*, 2008; Blair *et al.*, 2009).

The use of distributed systems is highly recommended at times due to the following motivations (Coulouris *et al.*, 2001; Tanenbaum & Van Steen, 2002; Kshemkalyani & Singhal, 2008; Oracle, 2010; SAS, 2010):

- a. *Access to geographically remote data and resources:* In many scenarios, the data cannot be replicated at every sites participating in the distributed execution because it may be too large or too sensitive to be replicated. For example, data within a multinational corporation is both too large and too sensitive to be replicated at every branch. It is therefore stored at a central server that can be queried by branch offices. In addition to that, limited resources in mobile devices as well as in the wireless technology with which these devices communicate have increased the need for distributed protocols and middleware.
- b. *Enhanced reliability:* The distribution concept has increased the system reliability because of the possibility of replicating resources and executions,

and since geographically distributed resources are not likely to crash at the same time under normal circumstances. Reliability is dependent on several aspects such as:

- *Availability:* The resource should be accessible at all times.
- *Integrity:* The value/state of the resource should be correct, in the face of concurrent access from multiple processors, as per the semantics expected by the application.
- *Fault-tolerance:* The ability to recover from system failures, where such failures may be defined to occur in one of the many failure models.

- c. *Increased performance/cost ratio:* By resource sharing and accessing geographically remote data and resources, the performance/cost ratio is increased. Although higher throughput has not necessarily been the main objective behind using a distributed system, nevertheless, any task can be partitioned across the various computers in the distributed system. Such a configuration provides a better performance/cost ratio than using special parallel machines.
- d. *Scalability:* As the processors are usually connected by a wide-area network, adding more processors does not pose a direct bottleneck for the communication network.
- e. *Modularity and incremental expandability:* Heterogeneous processors may be easily added into the system without affecting the performance, as long as those processors are running the same middleware algorithms. Similarly, existing processors may be easily replaced by other processors.

Sections 3.1.1 to 3.1.5 highlight some technologies that support distributed architecture. Furthermore, Tables 3.1 and 3.2 in section 3.6 summarize a brief comparison among these distributed technologies regarding components' distribution and interoperability.

3.1.1 Remote Procedure Call (RPC)

The first distributed computing technology to gain widespread use was the Remote Procedure Call commonly known as RPC. RPC is designed to be as similar to making local procedure calls as possible. The idea behind RPC is to make a function call to a procedure in another process and address space either on the same processor or across the network on another processor without having to deal with the concrete details of how this should be done besides making a procedure call (Coulouris *et al.*, 2001; Tanenbaum & Van Steen, 2002; Du & Raghavendra, 2005; Kshemkalyani & Singhal, 2008).

Before an RPC call can be made, both the client and the server both have to have stubs for the remote function that are usually generated by an interface definition language (IDL). When an RPC call is made by a client the arguments to the remote function are marshalled and sent across the network and the client waits until a response is sent by the server. There are some difficulties with marshalling certain arguments such as pointers (Coulouris *et al.*, 2001; Tanenbaum & Van Steen, 2002; Du & Raghavendra, 2005; Kshemkalyani & Singhal, 2008), since a memory address on a client is completely useless to the server so various strategies for passing pointers are usually implemented two rules: (1) disallowing pointer arguments and

(2) copying what the pointer points at and sending that to the remote function. RPC programs face a number of problems such as

- Network packets containing client requests being lost.
- Network packets containing server responses being lost.
- Client being unable to locate its server.

3.1.2 Common Object Request Broker Architecture (CORBA)

A CORBA usually consists of an Object Request Broker (ORB), a client and a server. An ORB is responsible for matching a requesting client to the server that performs the request, using an object reference to locate the target object. When the ORB examines the object reference and discovers that the target object is remote, it marshals the arguments and routes the invocation out over the network to the remote object's ORB. The remote ORB then invokes the method locally and sends the results back to the client via the network (Coulouris *et al.*, 2001; Wehrle *et al.*, 2007; Issarny *et al.*, 2008; Sutherland *et al.*, 2008; Sutherland *et al.*, 2009). There are many optional features that ORBs can implement besides merely sending and receiving remote method invocations including looking up objects by name, maintaining persistent objects, and supporting transaction processing. A primary feature of CORBA is its interoperability between various platforms and programming languages (Issarny *et al.*, 2008).

The first step in creating a CORBA application is to define the interface for the remote object using the OMG's interface definition language (IDL) (Issarny *et al.*, 2008). Compiling the IDL file will yield two forms of stub files; one that implements

the client side of the application and another that implements the server. Stubs and skeletons serve as proxies for clients and servers, respectively. Because IDL defines interfaces so strictly, the stub on the client side has no interacting with the skeleton on the server side, even if the two are compiled into different programming languages, use different ORBs and run on different operating systems. Then, in order to invoke the remote object instance, the client first obtains its object reference via the Orb. To make the remote invocation, the client uses the same code that it would use in a local invocation but use an object reference to the remote object instead of an instance of a local object.

When the ORB examines the object reference and discovers that the target object is remote, it marshals the arguments and routes the invocation out over the network to the remote object's ORB instead of to another process within the on the same computer. CORBA also supports dynamically discovering information about remote objects at runtime. The IDL compiler generates type information for each method in an interface and stores it in the Interface Repository (IR). A client can thus query the IR to get run-time information about a particular interface and then use that information to create and invoke a method on the remote CORBA server object dynamically through the Dynamic Invocation Interface (DII). Similarly, on the server side, the Dynamic Skeleton Interface (DSI) allows a client to invoke an operation of a remote CORBA Server object that has no compile time knowledge of the type of object it is implementing (Issarny *et al.*, 2008).

CORBA is often considered a superficial specification because it concerns itself more with syntax than with semantics. CORBA specifies a large number of services

that can be provided but only to the extent of describing what interfaces should be used by application developers. Unfortunately, the minimum that CORBA requires from service providers lacks mention of security, high availability, failure recovery, or guaranteed behaviour of objects outside the basic functionality provided and instead CORBA deems these features as optional. The end result of the lowest common denominator approach is that ORBs vary so wildly from vendor to vendor that it is extremely difficult to write portable CORBA code due to the fact that important features such as transactional support and error recovery are inconsistent across ORBs (Coulouris *et al.*, 2001; Bertrand *et al.*, 2005; Du & Raghavendra, 2005). Fortunately a lot of this has changed with the development of the CORBA Component Model, which is a superset of Enterprise Java Beans (Sullins & Whipple, 2005).

3.1.3 Distributed Component Object Model (DCOM)

Distributed Component Object Model (DCOM) is the distributed version of Microsoft's COM technology which allows the creation and use of binary objects/components from languages other than the one they were originally written in, it currently supports Java (J++), C++, Visual Basic, JScript, and VBScript. DCOM works over the network by using proxy's and stubs. When the client instantiates a component whose registry entry suggests that it resides outside the process space, DCOM creates a wrapper for the component and hands the client a pointer to the wrapper. This wrapper, called a proxy, simply marshals methods calls and routes them across the network (Voth *et al.*, 1998; Issarny *et al.*, 2008). On the other hand, DCOM creates another wrapper, called a stub, which unmarshals

methods calls and routes them to an instance of the component (Tanenbaum & Van Steen, 2002; Bertrand *et al.*, 2005; Du & Raghavendra, 2005; Kshemkalyani & Singhal, 2008).

3.1.4 Remote Method Invocation (RMI)

RMI is a technology that allows the sharing of Java objects between Java Virtual Machines (JVM) across a network. An RMI application consists of a server that creates remote objects that conform to a specified interface, which are available for method invocation to client applications that obtain a remote reference to the object (Bertrand *et al.*, 2005; Maassen *et al.*, 2008). RMI treats a remote object differently from a local object when the object is passed from one virtual machine to another. Rather than making a copy of the implementation object in the receiving virtual machine, RMI passes a remote stub for a remote object. The stub acts as the local representative, or proxy, for the remote object and basically is, to the caller, the remote reference. The caller invokes a method on the local stub, which is responsible for carrying out the method call on the remote object. A stub for a remote object implements the same set of remote interfaces that the remote object implements. This allows a stub to be cast to any of the interfaces that the remote object implements (Barai *et al.*, 2008). However, this also means that only those methods defined in a remote interface are available to be called in the receiving virtual machine (Tanenbaum & Van Steen, 2002; Bertrand *et al.*, 2005; Du & Raghavendra, 2005; Kshemkalyani & Singhal, 2008).

RMI provides the unique ability to dynamically load classes via their byte codes from one JVM to the other even if the class is not defined on the receiver's JVM.

This means that new object types can be added to an application simply by upgrading the classes on the server with no other work being done on the part of the receiver. This transparent loading of new classes via their byte codes is a unique feature of RMI that greatly simplifies modifying and updating a program.

The first step in creating an RMI application is creating a remote interface. A remote interface is a subclass of `java.rmi.Remote`, which indicates that it is a remote object whose methods can be invoked across virtual machines. Any object that implements this interface becomes a remote object.

To show dynamic class loading at work, an interface describing an object that can be serialized and passed from JVM to JVM shall also be created. The interface is a subclass of the `java.io.Serializable` interface. RMI uses the object serialization mechanism to transport objects by value between Java virtual machines. Implementing `Serializable` marks the class as being capable of conversion into a self-describing byte stream that can be used to reconstruct an exact copy of the serialized object when the object is read back from the stream. Any entity of any type can be passed to or from a remote method as long as the entity is an instance of a type that is a primitive data type, a remote object, or an object that implements the interface `java.io.Serializable`. Remote objects are essentially passed by reference. A remote object reference is a stub, which is a client-side proxy that implements the complete set of remote interfaces that the remote object implements. Local objects are passed by copy, using object serialization. By default all fields are copied, except those that are marked `static` or `transient`. Default serialization behaviour can be overridden on a class-by-class basis.

Thus clients of the distributed application can dynamically load objects that implement the remote interface even if they are not defined in the local virtual machine. The next step is to implement the remote interface, the implementation must define a constructor for the remote object as well as define all the methods declared in the interface once the class is created, the server must be able to create and install remote objects. The process for initializing the server includes; creating and installing a security manager, creating one or more instances of a remote object, and registering at least one of the remote objects with the RMI remote object registry (or another naming service such as one that uses JNDI), for bootstrapping purposes. An RMI client behaves similarly to a server; after installing a security manager, the client constructs a name used to look up a remote object. The client uses the Naming.lookup method to look up the remote object by name in the remote host's registry. When doing the name lookup, the code creates a URL that specifies the host where the server is running.

3.1.5 Service Oriented Architecture (SOA)

SOA is not a solution for a certain problem, but it is a framework architecture which adds some features to traditional software frameworks (Newcomer & Lomow, 2004; Barai *et al.*, 2008; Kumari *et al.*, 2008). Typically, “*SOA is a software architecture that starts with an interface definition and builds the entire application topology as a topology of interfaces, interface implementations, and interface calls*” (Barai *et al.*, 2008).

The core architecture of SOA is based on: services, messages, dynamic discoveries and web services (Chester, 2001; Agrawal *et al.*, 2002; Brown *et al.*, 2003;

Papazoglou, 2003; Heinzl *et al.*, 2006; Kumari *et al.*, 2008; Lam & Minsky, 2010).

In addition, the approach of designing web services offers the business logic to be decomposed in separated services (Natis, 2003; Hau *et al.*, 2008; Zhang *et al.*, 2008).

Each of these services is a distinct logical unit, and these units are interoperable to each other's.

The interoperability means that the units can communicate with each other's smoothly. Therefore, the interoperability requires that the business logic gets encapsulated in a service. A service can be an independent logical unit or it can contain other set of services. In case the service is used to call other sets of dependant services, to refer to those services, these must contain the service descriptions (Hau *et al.*, 2008; Phan *et al.*, 2008; Wang & Liu, 2008). The service description in its basic form contains the information of service name and location of the service being called (Derong *et al.*, 2005; Louridas, 2006; Matsumura *et al.*, 2006).

These logical units have to follow certain sets of communication standards to enable information flow across the enterprise offerings in an understandable form. The information is exchanged as messages from the interface designed within the system (Barai *et al.*, 2008).

At a higher level, SOA consists of three core components which are service provider (service), service consumer (consumer) and directory services (Broker), which also can be called "orchestration point" or "registry service" (Newcomer & Lomow, 2004).

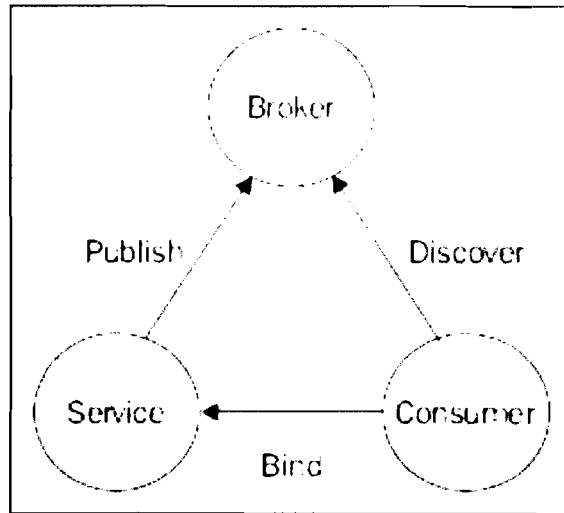


Figure 3.1: SOA Parts (Barai et al., 2008)

As shown in Figure 3.1, the service provider includes business processes as services. The services provided by the provider are called by the consumer to achieve some business goals. The process of services being provided and consumed is achieved by using directory services (broker) which lie between the service provider and the consumer.

To be available to the consumer, the service is published to the directory services in the broker. The consumer discovers the service from the broker. If the service is found, it refers and binds to the service and executes the processing logic.

3.1.6 Comparison of Distributed Technologies

Sections 3.1.1 – 3.1.5 explored and discussed five popular distributed technologies. For the purpose of highlighting the best technology that is capable of overcoming the distribution and interoperability gaps of the current ETL framework, Tables 3.1 and 3.2 show a comparison of the distributed technologies listed in sections 3.1.1 - 3.1.5

regarding components distribution and interoperability and the features that are dependent on the distribution and interoperability features.

Table 3.1: Distributed Technologies based on Components Distribution and Interoperability

	Components Distribution	Components Interoperability	Central Components Orchestration
RPC	Supported	Supported	Not supported
CORBA	Supported	Supported	Not supported
DCOM	Supported	Supported	Not supported
RMI	Supported	Supported	Not supported
SOA	Supported	Supported	Supported

Table 3.1 compares the distribution and interoperability features among ETL components. In addition, it compares the central orchestration among the ETL components. It is shown that the distribution and interoperability are supported by the five distribution technologies listed in the table. On the other hand, the central orchestration among the distributed components is completely supported only in SOA, while the other four technologies listed in Table 3.1 do not support central orchestration, and the interoperability is done directly among the components. This causes difficulties to manage and orchestrate many components sufficiently.

Table 3.2: Distributed Technologies based on Components Portability, Extensibility, and Legacy Compatibility

	Components Reusability	Components Portability	Components Extensibility	Compatibility with Legacy Systems
RPC	Limited support	Not supported	Supported	Not supported
CORBA	Limited support	Not supported	Supported	Not supported
DCOM	Limited support	Not supported	Supported	Not supported
RMI	Limited support	Not supported	Supported	Not supported
SOA	Unlimited support	Supported	Supported	Supported

Table 3.2 shows a comparison among distributed technologies based on components portability, extensibility, and legacy compatibility. Components reusability is supported without limitations by SOA, while it is supported by RPC if the used programming languages are compatible with each others. CORBA supports components reusability for all programming languages but with lots of configuration steps if the languages of the component are different, while DCOM supports reusability only within the same infrastructure. This infrastructure includes: programming languages, distribution technology, and configuration files. RMI supports reusability only for components that are built based on java programming language.

Components portability enables the distributed components to be executable in any programming environment and then to be stand alone components that can be plugged as portlets to web portals (Priebe & Pernul, 2003). The portability is

supported only is SOA due to the web services availability in which every component can be encapsulated in a web service that is standalone and pluggable to any SOA based portal. All distributed technologies support extensibility (Coulouris *et al.*, 2001; Tanenbaum & Van Steen, 2002; Kshemkalyani & Singhal, 2008), while only SOA supports components compatibility with legacy systems due to the web services involvement in SOA (Newcomer, 2002; Newcomer & Lomow, 2004; Weerawarana *et al.*, 2005).

Based on the brief literature about distributed technologies discussed in this section, several benefits can be obtained when SOA is used in redefining the ETL framework such as: distribution, interoperability, reusability, portability, and compatibility with legacy systems (Barai *et al.*, 2008). All of these advantages and other SOA specifications are discussed later in this chapter.

Since one of the major roles of SOA is to enhance the features of computerized frameworks and models (Jerstad *et al.*, 2005; Sneed, 2006; Barai *et al.*, 2008; Maurizio *et al.*, 2008; Laskey & Laskey, 2009), and ETL framework is a computerized framework (Kimball & Caserta, 2004), therefore, SOA can play a central role in enhancing the ETL framework features regarding component distribution and interoperability. In addition, it is worth to have a look at the available types of computerized application architectures. This helps in identifying the most suitable architecture for the new ETL framework.

3.2 Application Architectures and SOA

More than one set of applications will be running together to achieve some business goals when a system or an application is designed to do many duties (Coulouris *et al.*, 2001; Armstrong *et al.*, 2004). These applications are developed using different kinds of blueprints that are referred to as architecture. This architecture is an abstract view of the entire application, or a high-level overview of the system (Tanenbaum & Van Steen, 2002). Thus, application architecture can be considered as a representation of the structure of components and the interaction between them in any framework or model (Armstrong *et al.*, 2004). J2EE and .NET are good examples for software development and deployment frameworks (Armstrong *et al.*, 2004).

If J2EE is taken as an example of a software framework, the business requirements are converted into a high-level design where the first layer of HTML or JSP acts as the presentation layer. The business logic is encapsulated in the middle layer that can be built on Servlets or Enterprise Java Beans (EJB). In addition, the data is handled in the third layer which can be for example MySQL DBMS (Armstrong *et al.*, 2004). Each organization may have multiple application architectures, which would suit the need of different business goals. These applications could be Web based, or even the custom client server applications (Armstrong *et al.*, 2004).

3.2.1 Types of Application Architectures

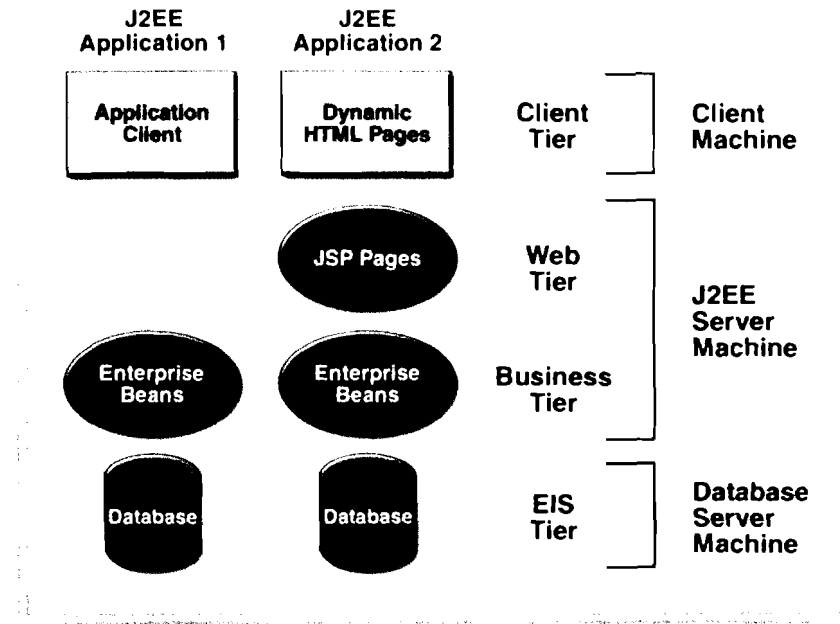


Figure 3.2: N-Tier Architecture (Armstrong *et al.*, 2004)

There are many types of application architectures, each has its own characteristics that have to be considered when a decision has to be taken to design a system based on it (Armstrong *et al.*, 2004; Barai *et al.*, 2008). The types of application architectures are:

a. *Client-Server Architecture*

The client-server architecture which is known as two-tier architecture separates the client from the server (Voth *et al.*, 1998; Coulouris *et al.*, 2001; Mykknen *et al.*, 2003; Shah *et al.*, 2003). The client initiates the request,

which the server processes and responds to. The client can send a request to one or more than a server at the same time.

Using this architecture, the responsibilities can be divided between the server and the client. Earlier, client and server parts were tightly coupled into an application. Due to this, it was difficult to process multiple clients. But, with the client-server architecture in place, business process is done within the server. This enables multiple clients to be plugged in at the same time (Voth *et al.*, 1998; Coulouris *et al.*, 2001; Mykknen *et al.*, 2003; Shah *et al.*, 2003).

Client-server architecture can be divided also to a three or N tier architecture. Each of these is briefly explained in this section. These two types of architectures are both based on three basic layers, the presentation layer, the business layer, and the database layer. Presentation layer is the one with which the client interact, it could either be a thin or a fat client (Armstrong *et al.*, 2004; Barai *et al.*, 2008). Business layer processes the information supplied by the presentation layer to accomplish a business goal according to a set of business rules (Voth *et al.*, 1998; Mykknen *et al.*, 2003; Urgaonkar *et al.*, 2005). Data layer that is sometimes called Enterprise Information System (EIS) stores the data and logic that would be used to achieve business goals. Before going to the three and N tier architectures, point (b) explains the one tier architecture.

b. *One-Tier Architecture*

The single tier application has the three layers: the presentation, the business, and the data layer tightly coupled and run on a single processing unit (Armstrong *et al.*, 2004). The application is designed in a way that the interaction between the layers is interwoven.

The limitations of 1-tier application architecture are:

- Changes to the database, in case it is being edited by multiple users can come with conflict problems (Tanenbaum & Van Steen, 2002; Armstrong *et al.*, 2004; Sullins & Whipple, 2005).
- Difficulty in scalability, as the application is running on a single machine (Armstrong *et al.*, 2004).

c. *Three-Tier Architecture*

In this type of architecture, the business layer resides between the presentation and the data layer. This enables the presentation logic to be independent of the data layer, and all its communication will happen through the business layer only (Sullins & Whipple, 2005).

The business layer can be multi-threaded so that multiple clients can access the business process. Typically, these business processes take client calls, convert them to queries, and then interact with the data layer. Subsequently, it translates the response from the data layer and passes it to the presentation layer. As a conclusion of the 3-tier architecture discussion, it has these

advantages: (1) the business layer can be multithreaded, which enables multiple clients to access the business functions; (2) enables the presentation layer to be light weight, as it does not have to take care of the database operations; (3) the components in each layer are reusable; and (4) each of the layers is easily scalable because it enables load balancing and clustering (Sullins & Whipple, 2005).

d. *N-Tier architecture*

An n-tier application as shown in Figure 3.2; has more than three layers. The business logic from the middle layer is structured in two layers. A part of the business logic resides in the application server that connects to the EIS (Enterprise Information Service) layer and the other part of the business logic remains in a web server, which connects to the presentation layer (Sullins & Whipple, 2005).

In a typical web-based solution, the client has access to the business through a browser. The browser calls the business logic in the web-server (Armstrong *et al.*, 2004). The web server then transfers the calls to the application server, which effectively sends the request to the data layer.

Adapting the N-Tier architecture in any distributable framework such as the ETL distributable framework has many advantages such as: (1) N-tier application offers the advantages of distributed computing that have been discussed in section 2.5; (2) the ability to distribute the components helps in applying the portability and reusability of the components (Voth *et al.*, 1998; Coulouris *et al.*, 2001; Mykknen *et*

al., 2003; Shah *et al.*, 2003; Armstrong *et al.*, 2004); (3) each of the tiers can reside in a different system even if the systems are heterogeneous (Armstrong *et al.*, 2004); (4) the division of the tasks helps in reducing load on each tier; (5) higher code maintainability can be achieved, which reduces the number of errors (Sullins & Whipple, 2005); and (6) all of the system components can appear as a unique system due to the distribution transparency that N-tier architecture offers (Wehrle *et al.*, 2005; Tsenov, 2007; Sutherland *et al.*, 2009; Li *et al.*, 2010).

Therefore, the N-tier architecture is adopted in the conceptual restructuring of the new ETL framework in this study as a result of its advantages over other tiers.

3.3 Extensible Markup Language (XML)

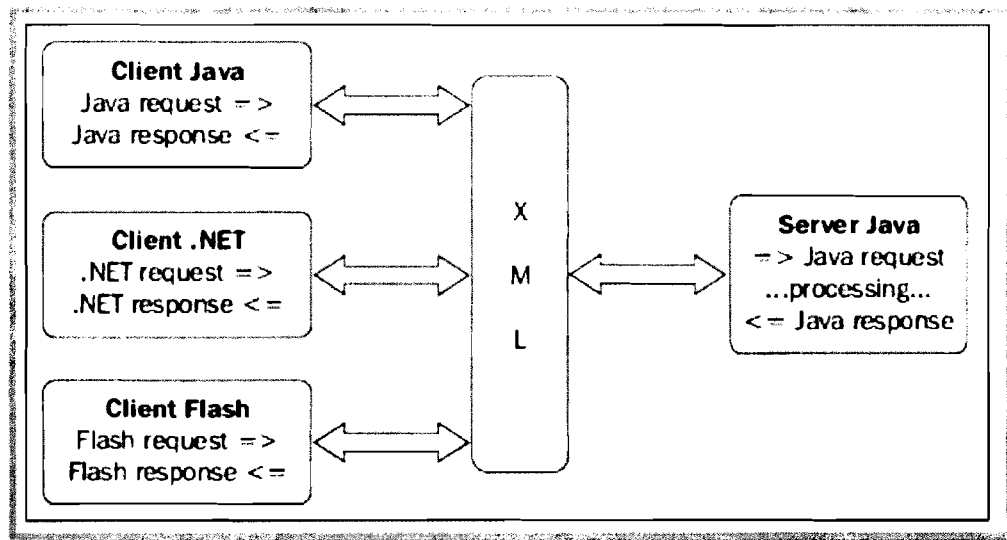


Figure 3.3: Data Portability Feature in XML (Barai *et al.*, 2008)

The eXtensible Markup Language was designed for data exchange (Cleveland, 2002) and has demonstrated its strength over time. Some of its features are that it is

structured, portable and extensible (Newcomer, 2002; Qiu *et al.*, 2002; Erl, 2004). Portability of XML is of great importance for data exchange. It typically means that the data can be exchanged among many heterogeneous environments (W3C, 1999). For example as shown in Figure 3.3, data can be exchanged between Java and .NET and Macromedia Flash applications without any changes (W3C, 1999; Wolter, 2001; Barai *et al.*, 2008). This means the data is application independent. Therefore, because of its portability feature (W3C, 1999), XML plays a major role in web services and SOA frameworks. This is explained further in section 3.4.

3.4 Web Services in SOA

Section 3.3 has briefly explained what XML is, and the reason of its adoption in web services and SOA implementation, however, there are no discussions on web services. The terms SOA and web services are sometimes mixed which results in some misunderstanding. Sections 3.4.1 to 3.4.4 explain web services in more details.

3.4.1 Web Services versus SOA Concepts

SOA is like a methodology, it is an architectural design choice (Sprott & Wilkes, 2004; Stojanovic *et al.*, 2004; Stal, 2006; Salter & Jennings, 2008). The first word is not short for "Web Service"; it is just "Service" in its wider meaning (Barai *et al.*, 2008). SOA is used as a basis for designing the high-level interfaces of the business rules of a given domain model.

On the other hand, web services concept is an architecture that constitutes a distributed computing environment (Shil & Ahmed, 2006), in which applications call functionalities from other applications either locally or remotely over an internal

network or an IP-network in a loosely-coupled way (Werner *et al.*, 2004; Barai *et al.*, 2008). Web services are encapsulated and can be invoked via standardized XML based interfaces (Cheng *et al.*, 2006).

3.4.2 Simple Object Access Protocol (SOAP) in Web Services

The Simple Object Access Protocol (SOAP) is a web service standard communication protocol (Chester, 2001; Tsai *et al.*, 2002; Yingying *et al.*, 2003; Weerawarana *et al.*, 2005; Louridas, 2006). It basically defines the structure of the exchanged messages, which is composed of an "envelope" with a "header" and a "body". This protocol adds powerful features to web services, which are: (1) auto generation of classes involved in the communication process (Barai *et al.*, 2008); (2) auto generation of the web service descriptor (WSDL) (Shil & Ahmed, 2006); (3) auto generation of client classes starting from the service descriptor (Newcomer, 2002); and (4) ability to be used with network protocols other than HTTP such as SMTP or JMS (Barai *et al.*, 2008).

3.4.3 Web Services Description Language (WSDL)

WSDL carries information about the service such as the input or output parameter, location of the service and binding information (Curbera *et al.*, 2002; Newcomer, 2002; Weerawarana *et al.*, 2005; Barai *et al.*, 2008). This helps in locating the service when a consumer requests it.

The WSDL document is supposed to have all information needed by the consumer to locate and execute the business logic within the web service (Tsai *et al.*, 2002; Vara *et al.*, 2005; Shil & Ahmed, 2006). It can be classified in two different entities:

abstract and concrete (Curbera *et al.*, 2002; Newcomer, 2002; Weerawarana *et al.*, 2005; Barai *et al.*, 2008). The abstract definition constitutes port and messages, whereas the concrete definition constitutes the binding, port, and service information.

3.4.4 Messaging

SOAP is the messaging protocol of sharing information over networks (Chester, 2001; Louridas, 2006). The information is stored in the form of XML data within SOAP which has a message specification that consists of envelope, header and body (Curbera *et al.*, 2002; Werner *et al.*, 2004; Heinzl *et al.*, 2006).

The header contains information about the SOAP message and the meta-data required by the message. This is, however, optional in the SOAP envelope. The body contains the actual message required for the execution of the Web services (Barai *et al.*, 2008). The body also includes information about faults which can be considered as a way of error handling (Barai *et al.*, 2008).

Messaging in SOA does the sharing of information (Werner *et al.*, 2004; Yang *et al.*, 2008). SOA-based solutions have a frequent usage of messages. The messages are designed in common format (Curbera *et al.*, 2002; Werner *et al.*, 2004; Heinzl *et al.*, 2006; Barai *et al.*, 2008) to be used by services available in an SOA-based solution.

3.5 Integration Benefits of Adopting SOA in the ETL Framework

Based on the discussion in sections 3.1 to 3.4, the advantages of involving SOA in the conceptual restructuring of the ETL framework are:

a. *Portability and Interoperability* (Barai *et al.*, 2008; Laskey & Laskey, 2009):

The portability feature, which SOA adds to its services and which is a direct result to XML adoption for data exchange between the web services leads to that any of the ETL components can be used as a portlet in a web portal (Priebe & Pernul, 2003). These can also be remotely managed through the web portal or even a normal web application to centralize the management of all the components even if they are available in different locations.

b. *Compatibility* (Jerstad *et al.*, 2005; Sneed, 2006; Barai *et al.*, 2008): As a result of applying SOA, any component of the ETL can be compatible with any platform or development environment. For example, the Extraction component as well as any other component can be compatible with .NET or J2EE framework without any change in the component coding. This is a direct result to the compatibility feature which SOA framework offers.

c. *Platform-neutrality* (Jerstad *et al.*, 2005; Iqbal & Daudpota, 2006; Sneed, 2006; Barai *et al.*, 2008; Klmek *et al.*, 2010): XML-based message information flow enhances the capability to achieve platform neutrality. These XML messages are based on XML standards, which eliminate the need to set up other messaging standards that can differ across platforms between ETL components. For example, if the Extract component of ETL is

encapsulated in a Java service inside a J2EE framework, and the Transform service is encapsulated in a .NET service inside a .NET framework, the two services can exchange messages while each is available in a different framework.

- d. *Extensibility and Reusability* (Jerstad *et al.*, 2005; Sneed, 2006; Barai *et al.*, 2008; Roach *et al.*, 2008; Laskey & Laskey, 2009): One of the most important goals of adapting SOA in the ETL framework is the component extensibility as well as reusability. Traditional ETL framework components were tightly coupled, which have led to difficulties to extend and reuse the ETL components in heterogeneous environments.

3.6 Conclusion

This chapter has briefly explained the SOA framework and its concepts, and has listed the advantages of conceptual restructuring of the ETL framework based on it. In addition, it has discussed the technologies which SOA is built on such as: web services, XML, SOAP, WSDL and messaging. Furthermore, the types of application architectures have been discussed and commented to brief the advantages of preferring N-tier architecture for an SOA-based ETL framework. Chapter 4 will discuss the methodology that was used in this research.

CHAPTER FOUR

RESEARCH METHODOLOGY

Chapters one, two and three have discussed thoroughly the gaps and problems in the current ETL framework. Based on the problems highlighted in the current ETL framework, the goal of this research is to define a conceptual framework for distributable and interoperable ETL components. Thus, this chapter elaborates the research methodology that was used in order to achieve the research aim in general, and the specific objectives.

The research strategy discussed in section 1.6 consists of four phases (first, second, third and fourth). Each phase achieves certain objectives of the research, and is able to answer the research question related to these objectives.

In the first phase, the focus is on understanding the problems of current ETL framework, while the second phase concentrates on exploring the SOA framework capabilities as well as proposing a conceptual ETL framework with distribution and interoperability features. The third phase concentrates on defining the new framework. The last phase focuses on evaluating and validating the new ETL framework. In order to achieve this goal, suitable methodologies have been adopted as a guide in implementing these research phases.

The research methodology used in this research adopts both the DSA (Design Science Approach) and Scrum methodologies. DSA is adopted as the main and comprehensive methodology of the research, while Scrum is adopted only for the prototype that was developed as a proof of the framework concepts. Scrum is

adopted for the prototype development part because it structures the prototype development process. That is because Scrum is one of the best methodologies for the development work (Cristal *et al.*, 2008; Judy & Krumins-Beens, 2008; Rayhan & Haque, 2008; Schwaber, 2009; Sutherland *et al.*, 2009). DSA is explained in section 4.1, while Scrum methodology is discussed in section 4.2.

4.1 Design Science Approach (DSA)

Design Science Approach (DSA) also known as Design Science Research (DSR) can be defined as “*a research paradigm in which a designer answers questions relevant to human problems via the creation of innovative artifacts, thereby contributing new knowledge to the body of scientific evidence. The designed artifacts are both useful and fundamental in understanding that problem*” (Hevner & Chatterjee, 2010).

The term *artifact* is used to describe something that is artificial, or constructed by humans, as opposed to something that occurs naturally (Hevner & Chatterjee, 2010). Such artifacts must improve upon existing solutions to a problem or perhaps provide a first solution to an important problem. IT artifacts, which are the end-goal of any design science research project, are: (1) constructs (vocabulary and symbols); (2) models (abstractions and representations); (3) methods (algorithms and practices); (4) instantiations (implemented and prototype systems); and (5) better theories.

Design science research has some guidelines that should be considered in conducting a research. These guidelines are: (1) design science research must produce a viable artifact in the form of a construct, a model, a method, or an instantiation; (2) the objective of design science research is to develop technology-based solutions to

important and relevant business problems; (3) evaluation of the utility, quality, and efficiency of a design artifact must be demonstrated via well-executed evaluation methods; (4) research must provide clear contributions in the areas of the design artifact, design foundations, and/or design methodologies; (5) the search for an effective artifact requires utilizing available means to reach desired ends while satisfying laws in the problem environment; and (6) design science research must be presented effectively to both technology-oriented and management-oriented audiences.

4.1.1 DSA phases

DSA is a widely-adopted research methodology in the field of information technology (Hevner & March, 2003; Vaishnavi & Kuechler, 2004; Niehaves & Becker, 2006). DSA consists of five phases as illustrated in Table 4.1.

Table 4.1: DSA Methodology Phases

PROCESS STEPS	OUTPUTS	DESCRIPTIONS
AWARENESS OF THE PROBLEM	Proposal	An awareness of relevant problems and proposing new research effort.
SUGGESTION	Tentative design	Suggesting the solution for the problems by proposing the tentative design.
DEVELOPMENT	Artifacts	Developing the artifacts based on tentative design and its implementation.
EVALUATION	Performance measures	Evaluation of the artifacts based on the criteria defined for certain expectations.
CONCLUSION	Results	Finalizing and concluding a specific research effort.

In this section, the five phases are explained based on the works of (Hevner & March, 2003; Hevner & Chatterjee, 2010), while section 4.3 discusses how these phases are used in this research :

- a. *Awareness of Problem:* An awareness of an interesting problem may come from multiple sources: intensive research, new developments in industry or in a reference discipline. Reading in a related discipline may also provide the opportunity for application of new findings to the researcher's field. The output of this phase is a proposal, formal or informal, for a new research effort.
- b. *Suggestion:* The suggestion phase follows immediately behind the proposal. Indeed, in any formal proposal for design research, a tentative design and the performance of a prototype based on that design would be an integral part of the proposal. Suggestion is an essentially creative step wherein new functionality is considered based on a novel configuration of either existing or new elements.
- c. *Development:* The tentative design or the theoretical framework or model is realized in this phase. The techniques for implementation will vary depending on the artifact to be constructed.
- d. *Evaluation:* Once constructed, the artifact is evaluated according to criteria that are frequently made explicit in the proposal (awareness of problem and suggestion phases). After that, analysis either confirms or contradicts a hypothesis.

- e. *Conclusion*: This phase is the final phase of a specific research effort. It is the result of satisfaction. Not only are the results of the effort consolidated and “written up” at this phase, but the knowledge gained in the effort is frequently categorized as either *firm* (facts that have been learned and can be repeatedly applied or behavior that can be repeatedly invoked) or as *loose ends* (further research could be done).

For the evaluation phase, case study is one of the widely used evaluation methods in software engineering and computer science. It is an ideal method when an in-depth investigation is needed. Case studies are designed to bring out the details from the viewpoint of the participants. Case studies have been also used in varied investigations, particularly in sociological studies (Yin, 1994).

(Yin, 1994) has identified six alternatives of evidence for case study results. The use of each of these depends on the case study needs to be conducted. The six alternatives identified by (Yin, 1994) are:

- documentation,
- archival records,
- interviews,
- direct observation,
- participant observation, and
- physical artifacts.

The case study should use the type that is relevant to the study. Table 4.2 details the strengths and weaknesses of each type.

Table 4.2: Types of Case Study Evidence (Tellis, 1997)

TYPE OF EVIDENCE	STRENGTHS	WEAKNESSES
<ul style="list-style-type: none"> Documentation 	<ul style="list-style-type: none"> Stable - repeated review Unobtrusive - exist prior to case study Exact - names etc. Broad coverage - extended time span 	<ul style="list-style-type: none"> Retrievability - difficult Biased selectivity Reporting bias - reflects author bias Access - may be blocked
<ul style="list-style-type: none"> Archival Records 	<ul style="list-style-type: none"> Stable - repeated review Unobtrusive - exist prior to case study Exact - names etc. Broad coverage - extended time span Precise and quantitative 	<ul style="list-style-type: none"> Retrievability - difficult Biased selectivity Reporting bias - reflects author bias Access - may be blocked Privacy might inhibit access
<ul style="list-style-type: none"> Interviews 	<ul style="list-style-type: none"> Targeted - focuses on case study topic Insightful - provides perceived causal inferences 	<ul style="list-style-type: none"> Bias due to poor questions Response bias Incomplete recollection Reflexivity - interviewee expresses what interviewer wants to hear
<ul style="list-style-type: none"> Direct Observation 	<ul style="list-style-type: none"> Reality - covers events in real time Contextual - covers event context 	<ul style="list-style-type: none"> Time-consuming Selectivity - might miss facts Reflexivity - observer's presence might cause change Cost - observers need time
<ul style="list-style-type: none"> Participant Observation 	<ul style="list-style-type: none"> Reality - covers events in real time Contextual - covers event context Insightful into interpersonal behavior 	<ul style="list-style-type: none"> Time-consuming Selectivity - might miss facts Reflexivity - observer's presence might cause change Cost - observers need time Bias due to investigator's actions
<ul style="list-style-type: none"> Physical Artifacts 	<ul style="list-style-type: none"> Insightful into cultural features Insightful into technical operations 	<ul style="list-style-type: none"> Selectivity Availability

Documents could be letters, memoranda, agendas, study reports, or any items that could be added to the database. The validity of the documents should be carefully reviewed so as to avoid incorrect data being included in the database. One of the most important uses of documents is to provide evidence gathered from other

sources. The potential for over-reliance on document as evidence in case studies has been criticized. There could be a danger of this occurrence if the investigator is inexperienced and mistakes some types of documents for unmitigated truth.

Archival records could be useful in some studies since they include service records, maps, charts, lists of names, survey data, and even personal records such as diaries. The investigator must be meticulous in determining the origin of the records and their accuracy.

Interviews are one of the most important sources of case study information. The interview could take one of several forms: open-ended, focused, or structured. In an open-ended interview, the researcher could ask for the informant's opinion on events or facts. This could serve to provide previously gathered data. In a focused interview, the respondent is interviewed for only a short time, and the questions asked could have come from the case study protocol. The structured interview is particularly useful in studies of neighborhoods where a formal survey is required.

Direct observation in a case study occurs when the observer(s) is/are selected from the organization where the case study is conducted. The observations could be formal or casual activities, but the reliability of the observation is the main concern. Using multiple observers or conducting the case study in more than one organization are ways to guard against this problem.

Participant observation is a unique mode of observation in which the researcher may actually participate in the events being studied. This technique could be used in studies of organizations. To avoid the potential bias of the researcher as an active

participant, one or more participants from the organization should be involved to get more precise observation results.

Physical artifacts could be any physical evidence that might be gathered during a site visit. That might include tools, art works, notebooks, computer output, and other such physical evidence.

4.2 Scrum Methodology

Scrum is considered as an iterative and incremental agile methodology (Sutherland *et al.*, 2007; Sutherland *et al.*, 2008; Schwaber, 2009), which is suitable for developing systems and managing projects. For this reason it is adopted for the prototype development in this research work. It is different from traditional software methodologies in which a system is treated as a whole task rather than subsets (Uy & Ioannou, 2008). Scrum begins with the establishment of the system backlog, which is a list of system requirements (Marchenko & Abrahamsson, 2008). Each element of the backlog is prioritized according to its importance. The element of the highest priority (that is more important to be done first) comes first (Sulaiman *et al.*, 2006; Barton & Campbell, 2007). Each part of the backlog is a sprint which takes around a month time (Schwaber, 2009). The sprint begins with a planning meeting, where the project team plan the work they will complete in the coming sprint, then each day begins includes a 15-minute meeting to show progress and rearrange the team members, work plans, and identify impediments to productivity (Judy & Krumins-Beens, 2008; Schwaber, 2009).

4.2.1 Scrum Artifacts

Scrum methodology includes three artifacts which are system (product) backlog, sprint backlog, and burn-down chart (Judy & Krumins-Beens, 2008; Sutherland *et al.*, 2009).

- a. *System Backlog*: At the beginning of the project, the system owner prepares a list of requirements prioritized by their importance regarding business values (Marchenko & Abrahamsson, 2008; Uy & Ioannou, 2008). This requirement list is the “System Backlog”. The backlog includes all features visible to the customer, and the technical requirements as well. The highest priority items in the backlog are broken down into small chunks to estimate and test them easily (Sulaiman *et al.*, 2006; Barton & Campbell, 2007). About 10 developer-days effort can be an acceptable size for a system backlog item that can be implemented in the next iteration (Cristal *et al.*, 2008; Sutherland *et al.*, 2009).
- b. *Sprint Backlog*: The sprint backlog is an artifact of the sprint planning meeting (Judy & Krumins-Beens, 2008; Rayhan & Haque, 2008). As mentioned earlier, system backlog's features are broken down into subsets; each of them is a sprint backlog (Sutherland *et al.*, 2007; Schwaber, 2009) which is a list of the specific development tasks required to implement some features. Each task is broken down into pieces that requires less than two working days for each (Schwaber, 2009). After the completion of the sprint backlog, the total estimated work is compared with original estimates of the system Backlog. If there is a big difference, the team negotiates with the

system owner to get the right work amount (Sutherland *et al.*, 2007; Schwaber, 2009).

- c. *Burndown Chart*: The burndown chart targets to show the cumulative remaining work in a sprint every day (Schwaber, 2009). During the sprint planning meeting, the Scrum team identifies specific tasks that have to be completed for the sprint success (Schwaber, 2009). All the sprint backlog remaining work which needs to be completed is called the cumulative backlog. After completing the sprint tasks, the Scrum master who is leading the team identifies and calculates the remaining work (Schwaber, 2009). The sprint is successful only when the cumulative sprint backlog reaches zero at the end of the sprint (Schwaber, 2009).

Scrum as other methodologies is divided into phases. These phases are explored in section 4.2.2.

4.2.2 SCRUM Phases

SCRUM is divided into Pregame, Game and Postgame phases (Sutherland *et al.*, 2007; Schwaber, 2009).

- a. *Pregame*: The pregame phase is composed of planning and architecture (Sutherland *et al.*, 2007; Schwaber, 2009).
 - The Planning is a definition of a new release based on currently known backlog, including an estimate of its schedule and cost (Barton & Campbell, 2007; Judy & Krumins-Beens, 2008; Marchenko & Abrahamsson, 2008). If the system is new, this phase includes both conceptualization and analysis, but if it is related only to

enhancements, this phase includes limited analysis (Niehaves & Becker, 2006).

- The architecture is considered as a design on how the backlog items will be implemented (Hevner & March, 2003; Niehaves & Becker, 2006; Sutherland *et al.*, 2008).

b. Game: The game phase concentrates on developing the sprints (Marchenko & Abrahamsson, 2008; Uy & Ioannou, 2008), typically the development of new functionalities with respect to the time, requirements, quality and cost.

c. Postgame: The postgame is considered the final phase of the Scrum methodology (Barton & Campbell, 2007; Schwaber, 2009), this phase concentrates on the closure which is a preparation for the release, including final documentation, pre-release testing, and release.

Therefore, it can be concluded that Scrum only concentrates on the system development aspects rather than covering all of the research phases. Thus, to cover all of the research phases, the DSA methodology that is explored in section 4.2; is adopted, and Scrum is used in DSA to be followed only in the development phase of DSA.

4.3 New ETL Framework Development Using DSA

The strategy of this research is designed based on DSA methodology. As a result of that and in combination with what has been explained in chapter one regarding the research strategy; the alignment of the research strategy with the DSA is presented in Table 4.3, which shows how the DSA process steps correspond to the research strategy phases.

Table 4.3: Research Strategy Based on DSA

METHODOLOGY	STRATEGY	OBJECTIVES	DELIVERABLES
AWARENESS OF THE PROBLEM	PHASE I	<ul style="list-style-type: none"> ▪ To identify the problems of ETL frameworks. 	<ul style="list-style-type: none"> ▪ Literature on methods and problems for defining a conceptual ETL framework with distribution and interoperability features.
SUGGESTION	PHASE II, III	<ul style="list-style-type: none"> ▪ To propose conceptual ETL framework with distribution and interoperability features. ▪ To define the theoretical framework. 	<ul style="list-style-type: none"> ▪ Distributed technologies and related concepts. ▪ Proposing a conceptual ETL framework with distribution and interoperability features. ▪ Defining the theoretical framework. ▪ Designing the ETL components based on the proposed framework. ▪ Suggesting a prototype for validating the framework. ▪ Suggesting 3 case studies to evaluate the prototype (one case study to evaluate E,T &L, and two case studies to evaluate E&L).
DEVELOPMENT USING SCRUM METHODOLOGY	PHASE IV	<ul style="list-style-type: none"> ▪ To develop a prototype based on the theoretical framework to help in validating the framework. 	<ul style="list-style-type: none"> ▪ Develop a prototype for a sample DW application following scrum methodology and fulfilling the defined theoretical framework. ▪ Conducting case studies for the purpose of evaluating the prototype (one case study to evaluate E,T&L, and two case studies to evaluate E&L)..
EVALUATION	PHASE IV	<ul style="list-style-type: none"> ▪ To have an evaluated conceptual ETL framework with distribution and interoperability features. 	<ul style="list-style-type: none"> ▪ Validating the theoretical framework through literature and expert opinions. ▪ Validating the framework through showing that the prototype follows the new theoretical framework. ▪ Testing and validating the prototype itself using. ▪ Evaluating the prototype using 3 case studies (one case study to evaluate E,T&L, and two case studies to evaluate E&L).. ▪ Refine the framework based on feedback from the testing and validation processes.
CONCLUSION	PHASE IV	<ul style="list-style-type: none"> ▪ To confirm and conclude the conceptual ETL framework. 	<ul style="list-style-type: none"> ▪ Confirm the framework as a theoretical framework that is accurate for any ETL implementation that targets to have distribution and interoperability features of the ETL components.

The achievement of phase I objectives contributes to the achievement of the objectives of phase II. In addition, the success of objectives in phase II contributes to the achievement of phase III objectives. Eventually, the success of phase III contributes to the work of phase IV that ends with the achievement of a conceptual framework for a component distributable and interoperable ETL.

As shown in Table 4.3, the Table matches the methodology phases with the research objectives and with the phases of the strategy followed in this research, and shows the deliverables of each phase of the methodology. The objectives of the research match the five phases of DSA to relate each objective to the suitable methodology phase.

The awareness of problems phase of the methodology matches with the 1st phase of the strategy and answers the question of: *What are the problems of the current ETL framework, which result from the absence of components distribution and interoperability?* The discussion of ETL literature, together with exploring the advantages of involving SOA, XML, distributed architecture and other related technologies are of great importance to identify the available problems, which are resultant from the absence of components distribution and interoperability in the current ETL framework. After completing this phase, a complete awareness of the problems is reached. The deliverable of this phase includes a comprehensive literature on methods and problems for defining a conceptual ETL framework with distribution and interoperability features.

The suggestion phase of the methodology matches with the 2nd and the 3rd phases of the research strategy. The suggestion phase starts with a comprehensive and

intensive literature review of some theoretical concepts. The literature review shows that SOA framework plays a central role in resolving components distribution and interoperability issues of the new contributed framework.

The research problems motivate for more research to bridge the gaps that have been left in the ETL field regarding distribution and interoperability. Literature about distributed systems, web services, SOA and software architecture is essential in determining the methods and concepts that help in bridging these gaps. After determining the appropriate methods and concepts and following them in bridging the gaps; the resultant partial solutions are combined together to come out with a proposed conceptual framework for interoperable distributed ETL components. Furthermore, in addition to what has been suggested, three organizations were selected to conduct the three case studies required for evaluating the research prototype.

The selected organizations are: Palestine Electric Company (PEC), Limkokwing University of Creative Technology (LUCT), and Professionals Information Technology (PIT). More information about these organizations' profiles is explored in Chapter 7. Generally, the organizations are selected because they use ETL to accomplish specific business needs, and specifically:

- PEC is chosen as it has four turbines. A built-in Siemens Power Management System is used to auto-read the necessary data for each of the four turbines. Then, the data of each turbine is transferred to a separate Windows 2003 database server with an Ms SQL Server DBMS installation for each server. There are two types of statistical reports at PEC. The first

type analyses the data of each turbine separately, while the other type is related to the whole power plant in which the reports analyze the data of the four turbines together. Therefore, for the second type, an ETL tool is needed to extract data from the four data sources and store it in an Ms. SQL Server DBMS data warehouse repository to generate statistical reports for the data related to the whole power plant.

In addition, majority of PEC top management are Arabic natives, while the data units generated by Siemens Power Management System are described in English. Therefore, a component for transforming the units' language during the ETL execution is needed. Based on that, each of Extraction, Transformation and Loading functionalities are required at PEC. In the case study of this research, the data is extracted from four data sources each of them is dedicated for a certain turbine. Moreover, PEC has different operating systems running over their servers and PEC always prefers platform independent applications and this feature is available at the SOA-based ETL prototype.

Based on the requirements of generating statistical reports of this case study, and based on the data available in PEC database, the data used to conduct this case study is generally textual data describing the records related to blackout transactions in PEC. The specific data types for the data source schema and Tables (changed after extraction) are shown in section A.4.1 of appendix A.

The volumes of data used in PEC for business intelligence and data warehouse projects differ from one project to another. Therefore, the volume of data differs from one business process to another. For the business process

of this case study that creates “Summary Report for 2010 Electricity Blackouts”, the main target of using ETL is to aggregate the data of the four turbines in one central data warehouse repository regardless of the volume of data. The case study used 87 records of data to generate the Summary Report for 2010 Electricity Blackouts.

- LUCT is chosen as it has 12 campuses in Malaysia, Botswana, Cambodia, China, Indonesia, Lesotho and United Kingdom. Every campus has its own DBMS running over a different platform. In addition, some statistical reports are related to the campus while other reports are related to the whole university. Therefore, an ETL tool is required, because necessary data needs to be extracted from all branches that participate in the same business process, and then are loaded to a central Ms SQL Server DBMS DW repository located in the main campus in Cyberjaya city, Malaysia.

In the case study of this research, the data is extracted from two data sources. The first is located in Malaysia, and the second in Botswana. The two data sources have the same Ms SQL Server DBMS database schema and format. In addition, LUCT grant different privileges to every staff member at the university, therefore, they always prefer distributable functionalities of software, which easily help them to grant specific privileges to specific staff members.

Based on the requirements of generating statistical reports of this case study, and based on the data available in LUCT database, the data used to conduct this case study is generally textual data describing the records related to students' performance in Java II for Feb-June, 2011 semester. The specific

data types for the data source schema and tables (changed after extraction) are shown in section A.4.2 of appendix A.

The volumes of data used in LUCT for business intelligence and data warehouse projects differ from one project to another. Therefore, the volume of data differs from one business process to another. For the business process of this case study that creates “Summary Report for Students’ Performance in Java II for Feb-June, 2011”, the main target of using ETL is to aggregate data of Malaysia and Botswana data sources in one central data warehouse repository regardless of the volume of data. The case study used 169 records of data to generate the Summary Report for Students’ Performance in Java II for Feb-June, 2011.

- PIT is chosen as it has 7 branches in Palestine. PIT needs to do periodical statistical reports to monitor trainers and trainees performance in the overall company. Four branches over the seven branches use Ms Access DBMS for keying the data. However, Ms Access is not compatible with most of business intelligence tools for the purpose of data analysis. Therefore, necessary data needs to be extracted from the 7 branches and loaded in the central DW repository at the company main branch.

In the case study of this research, the data is extracted from three data sources. Two of them use Ms. Access and the third uses Oracle 10g DBMS. Three of them are located in Palestine. The three data sources have the same database schema and format. In addition, PIT company consult open source and platform independent projects, and both of these features are available in the SOA-based ETL prototype.

Based on the requirements of generating statistical reports of this case study, and based on the data available in PIT database, the data used to conduct this case study is generally textual data describing the records related to trainers' performance in training IT courses. The specific data types for the data source schema and tables (changed after extraction) are shown in section A.4.3 of appendix A.

The volumes of data used in PIT for business intelligence and data warehouse projects differ from one project to another. Therefore, the volume of data differs from one business process to another. For the business process of this case study that creates "Summary Report for Trainers' Performance in Training IT Courses", the main target of using ETL is to aggregate the data of the three data sources in one central data warehouse repository regardless of the volume of data. The case study used 108 records of data to generate the Summary Report for Trainers' Performance in Training IT Courses.

The deliverables of 2nd phase of the strategy are: (1) comprehensive literature on ETL and its gaps as well as literature on SOA and related concepts; (2) a proposed conceptual ETL framework with distribution and interoperability features; and (3) Three case studies specifications for the purpose of evaluating the research prototype.

The 3rd phase of the strategy concentrates on defining the theoretical framework before realizing it by a prototype. Defining the framework depended on theories, standards and rules of ETL and SOA, and other concepts related to SOA. In addition to that, expert opinions through structured interviews are considered in defining the

framework (Chapter 5 includes the complete explanation of the new framework). Structured interview is the method used to get the experts' opinions and feedback regarding their satisfaction level with the theoretical framework. The expert opinion questionnaire shown in appendix C is used as a way to conduct the structured interviews. A structured interview is a fixed format interview in which all questions are prepared and put in the same order to each interviewee. Although this style lacks the free flow of a friendly conversation (as in an unstructured interview), however, it provides the precision and reliability. In addition, structured interviews ensure candidates have equal opportunities to provide information and are assessed accurately and consistently (Brace, 2008; USA, 2008).

Sometimes, the unstructured interview appears attractive due to its loose framework, flexible content and conversational flow. On the other hand, these same features make this type of interview very subjective, which reduces its accuracy and invites legal challenges. Research consistently indicates unstructured interviews have little value in giving the right feedback. Besides that, the lack of standardization in interview procedure and questions also makes the unstructured interview susceptible to legal challenges (Brace, 2008; USA, 2008). Therefore, structured interview is selected as the method of getting the feedback and opinions of experts regarding the theoretical framework. The deliverables of the 3rd phase are: (1) defining the Theoretical Framework; and (2) designing the ETL components based on the proposed framework.

Development, evaluation and conclusion phases of the methodology match the 4th phase of the strategy. The focus is on testing and validating the new framework.

Specific methods that comply with the ETL and SOA frameworks are used to define, test and verify the framework. The framework is realized in a prototype for testing and validation purposes. Then, the prototype system based on the defined framework was developed using Scrum methodology and tested using testing tools, discussed in chapter 6, and evaluated using 3 case studies. These case studies are explained in details in Chapter 7. By this stage, the final goal of the research was achieved and the conceptual framework is completely defined, verified and validated.

As shown in Table 4.3, Scrum was adopted for the prototype development. For more clarification regarding this adoption, section 4.4 discusses the flow of the research steps after combining DSA and Scrum methodologies.

4.4 Applying Scrum Methodology for the Development Phase of DSA

Scrum is considered as one of the strongest methodologies for developing prototypes and systems, and for managing people who are involved in developing those systems (Sutherland *et al.*, 2007; Rayhan & Haque, 2008; Uy & Ioannou, 2008). Scrum was used in this research for the development of the prototype. Figure 4.1 shows a flow chart for integrating the Scrum and DSA methodologies together.

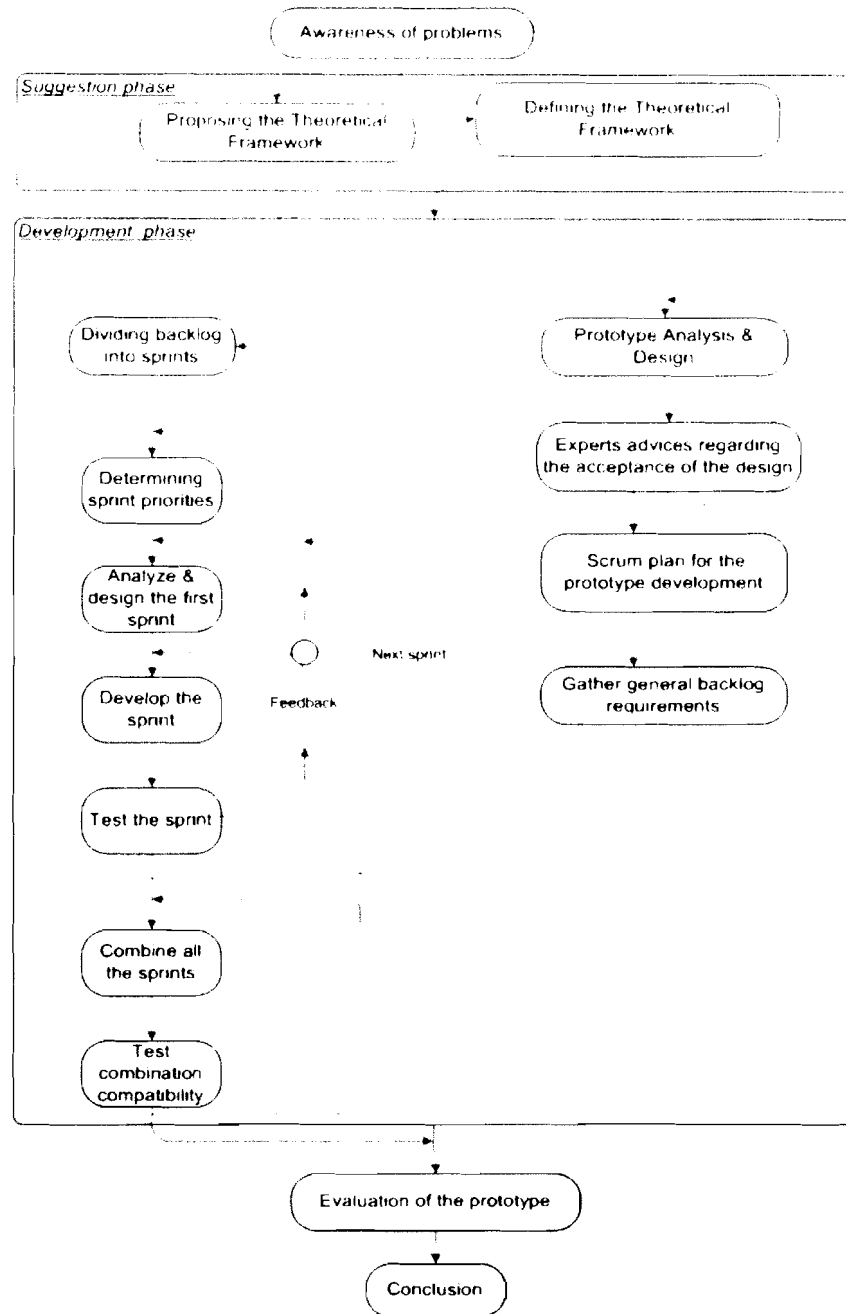


Figure 4.1: Flow Chart of the Methodology Applied in this Research

Based on the combination of both DSA and Scrum methodologies, the flow chart shows the order of steps that are followed in conducting this research. The main five

phases of DSA are the major steps in the flow chart. The development phase of DSA follows the Scrum steps as shown in the flow chart.

The flow chart shows that before going to the prototype development phase, the research is intensively conducted to accomplish: awareness of problems, proposing the theoretical framework and defining the theoretical framework. After defining the framework, Scrum is adopted for the development phase. Scrum steps are done sequentially to accomplish the prototype development: (1) prototype analysis and design; (2) getting experts advices regarding the acceptance of the design; (3) Scrum plan for the prototype development to do the whole prototype planning; (4) gathering general backlog requirements for the whole prototype; (5) dividing backlog into sprints for simplifying the development process; (6) determining sprint priorities to let the top priority tasks to take place first; (7) analyzing and designing the first sprint; (8) developing the first sprint; (9) testing the first sprint, after that the order goes for the next priority sprint and so on; (10) combining all the sprints and (11) testing combination compatibility.

The testing sprints will be done for the purpose of validating and verifying the prototype. The required types of testing are: unit testing, Web service testing, compatibility testing, classified-fragmentation speed testing, and end to end testing. Each of these tests will be done in a separate sprint.

After prototype testing, the prototype was evaluated using the data provided by 3 case studies. The method used to conduct the three case studies has adopted the recommendation of (Yin, 1994) that include four stages: (1) design the case study, (2) conduct the case study, (3) case study evidence, and (4) case study conclusion.

These four recommended steps are followed in conducting each of the three case studies. A common design for the 3 case studies is explored in Figure 4.2, while the details of conducting the 3 case studies are listed in Table 4.4.

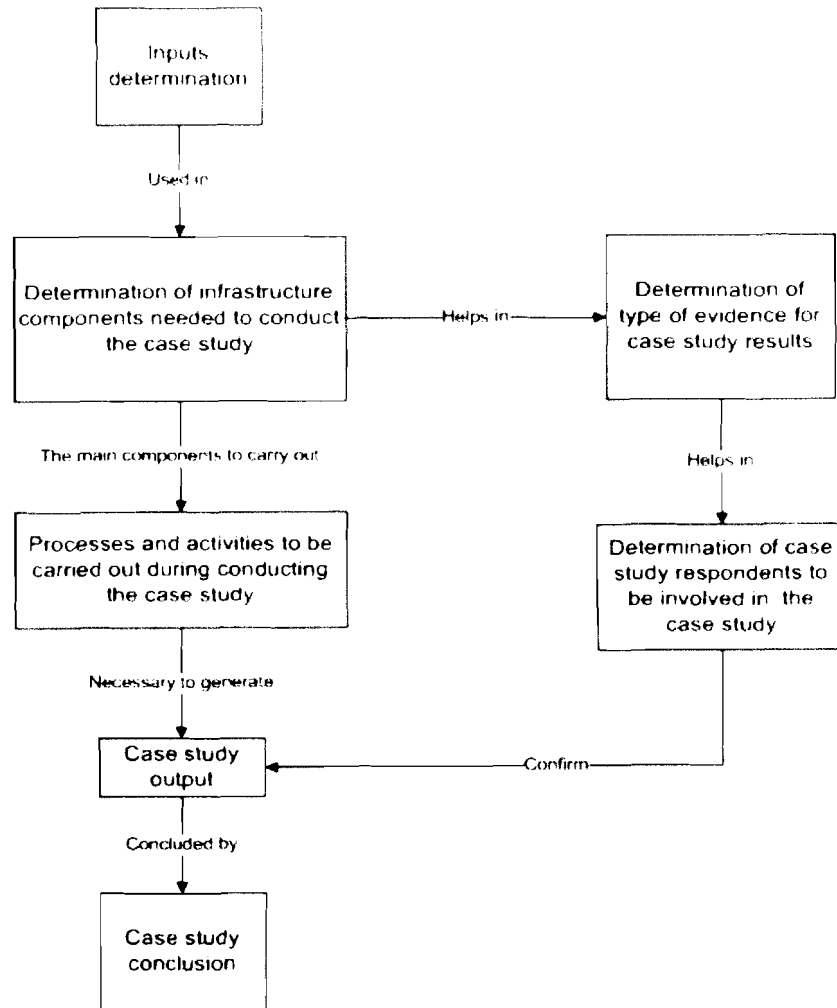


Figure 4.2: Case Study Design

Table 4.4(Part A): Details of the Case Studies based on Case Study Design

		Case study # 1	Case study # 2	Case study # 3
Inputs determined		A database schema related to the source databases. This schema included auto generated text data for the purpose of conducting the case study. The auto generated data has matched the original data in data type and format to represent the original data that is private to PEC.	A database schema related to the source databases. This schema included auto generated text data for the purpose of conducting the case study. The auto generated data has matched the original data in data type and format to represent the original data that is private to Limkokwing University.	A database schema related to the source databases. This schema included auto generated text data for the purpose of conducting the case study. The auto generated data has matched the original data in data type and format to represent the original data that is private to PIT company.
Determined infrastructure components needed to conduct the case study	Using the traditional ETL tool	<ul style="list-style-type: none"> ▪ Ms SQL Server DBMS ▪ SQL Server Integration Services (SSIS) ▪ Windows 2003 server (Details explored in section 7.1.3) 	<ul style="list-style-type: none"> ▪ Ms SQL Server DBMS ▪ SQL Server Integration Services (SSIS) ▪ Windows 2003 server (Details explored in section 7.2.3) 	<ul style="list-style-type: none"> ▪ Oracle DBMS ▪ Oracle Warehouse Builder (OWB) ▪ Windows 2003 server (Details explored in section 7.3.3)
	Using SOA-based ETL prototype	<ul style="list-style-type: none"> ▪ The SOA-based ETL prototype ▪ 2 Redhat Linux Servers ▪ 2 Windows 2003 Servers ▪ JBoss Application Server ▪ GlassFish Application Server (Details explored in section 7.1.4) 	<ul style="list-style-type: none"> ▪ The SOA-based ETL prototype ▪ 2 Redhat Linux Servers ▪ 2 Windows 2003 Servers ▪ Apache Tomcat Web Server ▪ GlassFish Application Server (Details explored in section 7.2.4) 	<ul style="list-style-type: none"> ▪ The SOA-based ETL prototype ▪ 2 Redhat Linux Servers ▪ 2 Windows 2003 Servers ▪ Apache Tomcat Web Server ▪ GlassFish Application Server (Details explored in section 7.3.4)

Table 4.4 (Cont., Part B): Details of the Case Studies based on Case Study Design

Processes and activities to be carried out during conducting the case study	<ol style="list-style-type: none"> 1. Preparing infrastructure components needed to conduct the case study such as data sources connections and software installations. 2. Extracting required fields for data warehouse Star-Schema generation using the traditional ETL tool 3. Applying Transform functionality using the traditional ETL tool 4. Applying Load functionality using the traditional ETL tool 5. Extracting required fields for data warehouse Star-Schema generation using SOA-based ETL Prototype 6. Applying Transform functionality using SOA-based ETL Prototype 7. Applying Load functionality using SOA-based ETL Prototype 	<ol style="list-style-type: none"> 1. Preparing infrastructure components needed to conduct the case study such as data sources connections and software installations. 2. Extracting required fields for data warehouse Star-Schema generation using the traditional ETL tool 3. Applying Load functionality using the traditional ETL tool 4. Extracting required fields for data warehouse Star-Schema generation using SOA-based ETL Prototype 5. Applying Load functionality using SOA-based ETL Prototype 	<ol style="list-style-type: none"> 1. Preparing infrastructure components needed to conduct the case study such as data sources connections and software installations. 2. Extracting required fields for data warehouse Star-Schema generation using the traditional ETL tool 3. Applying Load functionality using the traditional ETL tool 4. Extracting required fields for data warehouse Star-Schema generation using SOA-based ETL Prototype 5. Applying Load functionality using SOA-based ETL Prototype
Determined type of evidence for case study results	Direct observation is chosen because this type of evidence covers events in real time, covers event context.	Direct observation is chosen because this type of evidence covers events in real time, covers event context.	Direct observation is chosen because this type of evidence covers events in real time, covers event context.
Determined case study respondents to be involved in the case study	The IT Supervisor of PEC	Two IT lecturers at Cyberjaya campus, Limkokwing university.	The Company general manager & the executive manager-trainer
Case study output	Successfully loaded data to the final DW repository	Successfully loaded data to the final DW repository	Successfully loaded data to the final DW repository

4.5 Conclusion

This research focuses on defining a conceptual framework for distributable and interoperable ETL components. To achieve this goal, both the Scrum and DSA methodologies are adopted. Scrum methodology is powerful for development tasks. In contrast, it does not have a complete support for all of the research phases. DSA is a powerful methodology for research but does not handle the development phase as powerful as Scrum methodology. Thus, DSA and Scrum are combined to establish a more appropriate methodology for this research. This newly integrated methodology provides more suitable methods for executing the research.

CHAPTER FIVE

THE NEW ETL FRAMEWORK

This chapter consolidates and further defines the conceptual solution of the research problems. Defining the new ETL theoretical framework consists of some mandatory specifications. These specifications are clear and strict enough to make sure that interoperability and distribution of the ETL components exist. In particular, ETL vendors or developers who are interested to adopt the new ETL framework as a base to produce their own ETL tools have to consider a set of specifications regarding: Distribution, Web services, SOA orchestration processes, XML schemas, WSDL documents, SOA-based composition, and the framework extensibility. These specifications are discussed in sections 5.2 to 5.6. In addition, before discussing these specifications, a conceptual comparison between the traditional (current) ETL framework and the new ETL framework (which is the output of this research) is explored in section 5.1 to clearly highlight the difference between both frameworks and to explore the relations among the new ETL framework concepts.

5.1 Overview of the New ETL Framework

Figure 5.1 illustrates the new ETL framework and abstractly describes it. In the data layer, the data stores that are involved in the overall process are depicted. The left side shows the original data providers, which are typically, relational databases and files. Data Staging Area is used as temporary data storage during the transformation, cleaning and classified fragmentation processes. The data warehouse repositories are used to store the final data for report generation or other BI tasks.

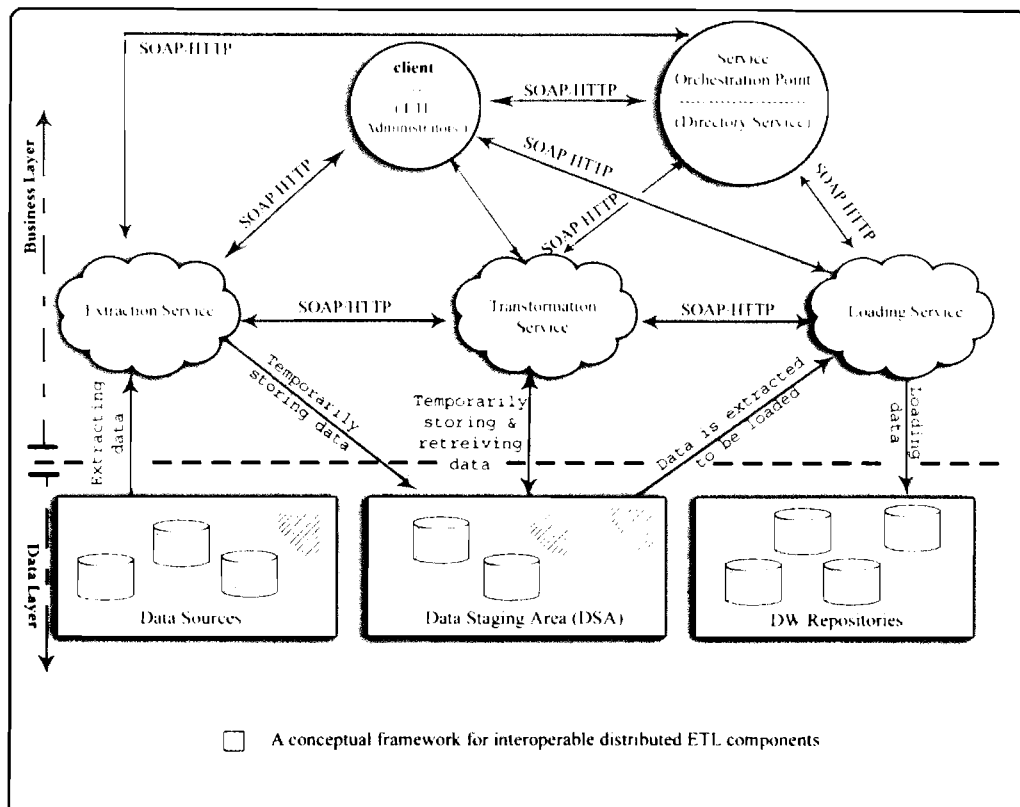


Figure 5.1: A Conceptual Framework for Interoperable Distributed ETL Components

The business layer includes four main parts which are:

- a. *Service Orchestration Point (also called directory service or service registry)*: Describes the services available in its domain which are extraction, transformation, and loading. Those three services are called service providers and register themselves in the orchestration point.
- b. *Service providers*: Each of them is a component that performs a service in response to a consumer request. The framework has three service providers which are extraction, transformation, and loading services.
- c. *Service consumers*: Each of them is a component that consumes the result of a service supplied by a provider: the main service consumer in the framework

is the client which represents ETL administrators, also the three service providers can be service consumers to other services. For example, the transformation service can request some functions from the extraction service in the case that the extraction and the transformation are executed in one patch.

- d. *Service interface*: Defines the programmatic access of the four services, establishes the identity of the service and the rules of the service invocation.

The relationship between a service provider and consumer is dynamic and established at runtime by a binding mechanism done by the orchestration point. This dynamic binding minimizes the dependencies between the service consumer and service provider, which supports the loose coupling feature of the framework.

In other words, extraction, transformation, and loading functionalities are loosely coupled to each other by distributing each functionality in an interoperable web service. The orchestration point contains information about each service such as its interface. A client can discover services by examining the orchestration point. After looking up the required ETL service, the client continues communication remotely and directly with any distributed ETL service, in this case the client is the service consumer and the ETL service is the service provider. An ETL service can also be a consumer to another ETL service, where it discovers the availability of that service using orchestration point as well.

To clearly show the flow of actions when a client demand an execution of an ETL functionality, a simple flow diagram is shown in Figure 5.2.

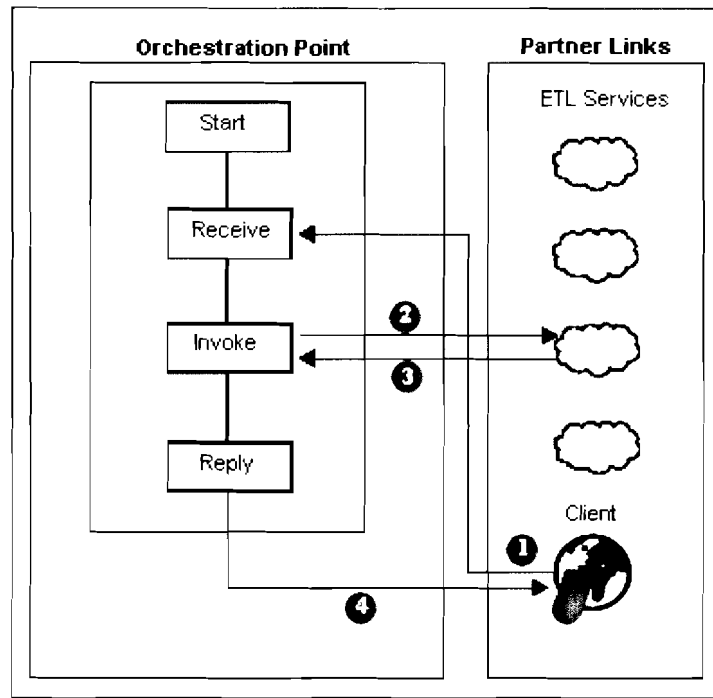


Figure 5.2: Flow Diagram for Steps to Consume an ETL Service by a Client

(1) When a client demand to consume (execute) a certain ETL service, the orchestration point starts with a receive activity in which it receives the client request; (2) then, proceeds with invoking the suitable ETL service(s); (3) the required service confirms the invocation request and (4) finishes by replying back to the client. An orchestration point process typically interacts with one or more ETL web Services (the orchestration point process is also a web service). These ETL web services are called also partner services or external service.

By comparing the traditional ETL framework briefed in section 2.2 and the new ETL framework explored in this section, the new ETL framework has involved advantages over the traditional one, these advantages are:

- a. *Components' Distribution and Interoperability:* The interaction between clients (ETL administrators) and loosely-coupled distributed ETL services has widespread interoperability. It is required that clients and services can communicate and understand each other no matter what platform they run on. This target is met because clients and ETL services have a standard way of communication among each other. In addition, interoperability provides consistency across platforms, systems, and languages for the ETL framework components.
- b. *Flexibility:* Loosely-coupled services in ETL framework are typically more flexible than tightly-coupled ETL framework. In a tightly-coupled architecture, the ETL components are tightly coupled to each other, sharing semantics, libraries, and often sharing state. This makes it difficult to evolve the application to keep up with ever changing business requirements. The loosely-coupled, document-based, asynchronous nature of loosely-coupled framework services allows applications to be flexible, and easy to evolve with changing requirements. As a direct result, the new ETL framework can have an extra component without complexities. Furthermore, the new ETL framework can be extended in the future by adding any extra component to suit enterprises' new business needs.
- c. *Reusability:* The new ETL framework allows ETL developers in data warehouse industry to adopt the code of an existing ETL component developed by any other ETL provider who follows the new framework specifications, and then reuse it to meet new ETL business requirements.

Reusing functionality that already exists outside or inside an enterprise instead of developing code that reproduces those functions; can result in a huge savings in application development cost and time.

- d. *Scalability*: Since the new ETL framework services are loosely coupled, applications that follow the framework specifications can scale easily (much easier than applications that follow more tightly-coupled frameworks). This is because there are few dependencies between the requesting application and the services it uses. That enables the service to handle more requests because it does not need to handle lots of communication overhead like tightly coupled components.
- e. *Cost Efficiency*: In the current ETL framework, if there is a demand to integrate different business resources to the ETL framework implementations (such as data source legacy systems, business partner applications, and organization-specific solutions) can be expensive because these components are tied together in a customized way. Customized solutions are costly because they require extensive analysis, development time, and effort. In addition, they are costly to maintain and extend because these are tightly-coupled, as the changes in one component of the integrated solution require changes in other components. An ETL framework based on web services and SOA should result in less costly solutions because the integration of clients and services does not require in depth analysis and unique code of customized solutions. Furthermore, because services in an SOA based ETL

framework are loosely-coupled, applications that use these services should be less costly to maintain and easier to extend than customized solutions.

This section has briefed the new ETL framework while the specifications of the framework are discussed in sections 5.2 to 5.6

5.2 Distributed Architecture Specifications

There are many distributed architectures that could be followed to meet the distribution specification of the new ETL framework. CORBA (Common Object Request Broker Architecture) (Issarny *et al.*, 2008), RMI (Remote Method Invocation) (Bertrand *et al.*, 2005; Sullins & Whipple, 2005), RPC (Remote Procedure Call) (Tanenbaum & Van Steen, 2002) and EJB (Enterprise Java Beans) (Sullins & Whipple, 2005) are samples of valid distribution architectures. Any of these distribution architectures or others could be followed to fulfill the distribution requirement of the ETL framework. By fulfilling this requirement, a distributed multi-tiered ETL application can be accomplished. The application logic should be divided into components according to function, and the various ETL components should be (optionally) installed on different machines; depending on the tier which the application component belongs to.

Java 2 Enterprise Edition (J2EE) is an open source architecture that supports the distribution concept and the multi-tiered application architecture. In addition, it freely provides all the tools might be needed to implement a distributed project including application servers, database connectors, development and deployment IDEs, and testing tools. Furthermore, J2EE is platform independent because it is

built over Java language that is completely platform independent (Armstrong *et al.*, 2004; Perin, 2009). Therefore, if the standards of EJB architecture (the distribution model of J2EE (Armstrong *et al.*, 2004; Perin, 2009)) is followed, the distribution will be similar to Figure 3.2 which shows two multi-tiered J2EE applications divided into these tiers:

- Client-tier components run on the client machine.
- Web-tier components run on web container of the J2EE server.
- Business-tier components run on the EJB container of the J2EE server.
- Enterprise information system (EIS)-tier runs on the EIS server.

Although a J2EE application can consist of the three or four tiers as shown in Figure 3.2, J2EE multi-tiered applications are generally considered to be three-tiered applications because they are distributed over three locations: client machines, the J2EE server machine, and the database or legacy machines at the back end. Therefore, it is highly recommended that the implementation of any ETL tool based on the new ETL framework to be built on a multi-tier or at least 3-tier architecture using the standards of any distributed architecture.

5.3 Web Services Involvement Specifications

The technology associated with the realization of SOA is Web services (Newcomer & Lomow, 2004; Jerstad *et al.*, 2005; Weerawarana *et al.*, 2005; Sneed, 2006; Laskey & Laskey, 2009). A typical Web service needed in the new ETL framework is comprised of:

- a. *A decoupled technical service contract:* based on the standards of WSDL and XML Schema, the service contract has to consist of a WSDL definition and an XML schema definition (W3C, 1999; Newcomer, 2002; Tsai *et al.*, 2002; Weerawarana *et al.*, 2005). This service contract declares public functions (operations). Basically, three operations for extraction, transformation and loading are needed, and an extra operation is needed for any extra extension component of the ETL.
- b. *A body of programming logic:* based on web services standards (Erl, 2004; Derong *et al.*, 2005; Louridas, 2006; Matsumura *et al.*, 2006), this logic may be custom-developed for the Web service, or it may exist as legacy logic that is wrapped by a Web service for its functionality to be made available via Web services communication standards. In the case that logic is custom-developed, it is created as components and is referred to as business logic.
- c. *Message processing logic:* based on web service communication standards (Gao *et al.*, ; Werner *et al.*, 2004; Heinzl *et al.*, 2006), message processing logic exists as a combination of parsers and processors. Much of this logic is provided by the runtime environment, but it can also be customized. The programs that carry out message-related processing are primarily event-driven and therefore can intercept a message subsequent to transmission or prior to receipt. It is common for multiple message processing programs to be invoked with every message exchange (Tsenov, 2007).

A Web service can be associated with temporary roles, depending on its utilization at runtime. For example, it acts as a service provider when it receives and responds to

request messages, but can also do the role of service consumer when it is required to issue request messages to other Web services. For instance, the transformation service can be a provider to the client and a consumer to the extraction service in case that extraction and transformation are executed in one batch.

5.3.1 A Web Service for every Distributed ETL Component

To achieve the benefits of adopting SOA, based on SOA standards (Hau *et al.*, 2008; Zhang *et al.*, 2008; Lam & Minsky, 2010), every distributed component of the ETL framework has to be wrapped with a separated Web service. Each of these services is called an external service. External services (partners) and the client service interact with the orchestration point process. This process starts and ends somewhere, and involves the interaction of at least one external partner. In the ETL framework, the partner is the web service that has the ability to process the client's request, i.e., the ETL components. The ETL administrator sends his request for processing. Then, the orchestration point receives the request, invokes a web service that processes the request, and returns the response back to the ETL administrator.

5.3.2 Web Service Operations

Remote Invocations of Web service operations of the ETL framework are asynchronous, i.e., the service provider must be capable of accepting requests from clients without notice. The designer of Web services needs to decide how to ensure that his or her implementation is compatible with the way in which a service provider supports asynchronous operations. To achieve this asynchronous feature, according to SOA standards (Newcomer & Lomow, 2004; Barai *et al.*, 2008; Hau *et*

al., 2008; Mitchell *et al.*, 2008; Mulik *et al.*, 2008; Zhang *et al.*, 2008; Laskey & Laskey, 2009; Lam & Minsky, 2010), it is highly recommended that each Web service includes only one operation. For example, there should be an extraction distributed module includes one Web service that includes one operation for the extraction functionality, and the same for transformation and loading functionalities. The asynchronous scenarios for an ETL Web service would include:

- Production and transmission of a request message by a client.
- Consumption of the request message by the service provider.
- Production and transmission of a response message by the service provider.
- Consumption of the response message by the client.

These scenarios points have to be considered in any implementation of the ETL framework, and it is left to the ETL designer to choose the technologies to achieve these.

5.3.3 WSDL Documents

There is a need for some standards to be considered when developing and integrating an ETL application (tool) based on the new ETL framework. The Web Services Definition Language (WSDL) (Curbera *et al.*, 2002; Newcomer, 2002) provides a standard for describing services, the location of services, and what operations these services provide, all in a platform and language independent way. This allows the ETL application developer to build loosely coupled services, which is one of the key factors of building successful SOA-based ETL tools.

WSDL documents could provide both an abstract and a concrete view of the ETL services they are describing. The abstract view describes the operations of the services while the concrete view describes how those operations are mapped onto a specific protocol such as SOAP. Furthermore, WSDL documents are correctly formed XML documents, which helps in developing ETL services up to Service Oriented Architecture standards (Barai *et al.*, 2008; Salter & Jennings, 2008). Correctly formed XML documents can be read and parsed automatically by applications (W3C, 1999), which allows clients to look up and consume web services automatically through the orchestration point. Briefly, the key benefits of using WSDL are:

- a. *WSDL is XML-based* (W3C, 1999; Klmek *et al.*, 2010): Any system that can read and interpret XML, can read and interpret WSDL documents of the ETL component service.
- b. *WSDL is platform and language independent* (Weerawarana *et al.*, 2005): It is not tied with one programming language or platform. WSDL-based ETL Web services could be developed and deployed in Microsoft's .NET environment and consumed via Java GUI applications. Alternately, WSDL-based web services could be developed and deployed in a Java environment and consumed via PHP-based web applications.
- c. *WSDL describes ETL services abstractly* (Weerawarana *et al.*, 2005): ETL services are described abstractly as they support loose coupling and concentrates on developing interfaces to support the abstraction which is necessary for loose coupling.

- d. *WSDL is extensible* (Curbera *et al.*, 2002; Tsai *et al.*, 2002; Vara *et al.*, 2005; Shil & Ahmed, 2006): WSDL does not tie communications to one specific protocol, for example SOAP over HTTP. As new transport mechanisms are required, additional bindings can be specified. Currently, there is only a need to access WSDL described ETL services via SOAP, but in the future, there might be a need to use JMS or SMTP bindings. WSDL format enables the extensibility of WSDL documents to add additional bindings when necessary.

For specifying the format of WSDL documents that are supposed to be followed to implement ETL Web services based on the new ETL framework, the basic structure of a WSDL document should look like the WSDL file of Table A.1.

The WSDL file is divided into 5 main elements, all lie under the main <definitions> element:

- <types/>: Describes the data types used by the ETL operations; described for the service. These data types are defined using XML Schema Definitions. XML Schema specification is explored in subsection 5.2.4.
- <message/>: Describes the messages that are used within the ETL service.
- <portType/>: Describes the operations that are available within the ETL service.
- <binding/>: Describes what binding the service is using (Up to the time of writing this thesis it is SOAP)
- <service/>: Describes connection details for the specific bindings.

5.3.4 XML Schema

Over recent years, XML has become a main technology for data interchange between different systems (W3C, 1999; Chester, 2001; Wolter, 2001; Cleveland, 2002; Newcomer, 2002; Qiu *et al.*, 2002; Green *et al.*, 2003; Erl, 2004; Iqbal & Daudpota, 2006; Mahboubi & Darmont, 2009; Klmek *et al.*, 2010). The advantages of XML over other technologies is that it is extensible (additional attributes or elements can be added to data structures) and self describing. By using these advantages, a simple XML file that could be followed as a base in the ETL framework and is shown in Table A.2.

Based on Table A.2, it can easily be seen that a single ETL element called classified-fragmentation is described. This XML file can be expanded by adding datetime element as shown in Table A.3.

Based on Table A.3, it is possible to extend XML and add new elements, attributes or entire structures into an XML file. However, describing data structures in XML is only half of the problem. The other half is: how it is known that the XML received is the expected XML? For example, if an ETL XML element is needed with fields: name and datetime, the XML sample above meets these requirements, however, also the sample below meets the requirements as well. Which one is correct?

```
<?xml version="1.0"?>
```

```
<ETL name="classified-fragmentation" datetime="2010"/>
```

Both XML files are completely valid, however there is no way that an application would be able to parse both of these XML files and obtain the same results. XML

Schema answers the problems here by allowing the description of how the XML should be defined. This can be accomplished using XML Schema in which:

- name is an attribute.
- datatype is an element called datatype or an attribute called " datatype ".

XML Schema allows the definition of the elements and attributes that are present within an XML file, and allows the XML file to be machine validated for correctness. This is important, because XML schema enables applications to validate XML files, which means that applications can check whether data is valid or not before it attempt to process the data.

The W3C description of XML Schema (W3C, 1999) is:

The XML schema language can be used to define, describe, and catalogue XML vocabularies for classes of XML documents. Any application of XML can use the Schema formalism to express syntactic, structural, and value constraints applicable to its document instances.

Given this knowledge of XML schema, a simple XML schema (etl.xsd) is defined in Table A.4, and also an XML file (etl.xml) for ETL data structure is defined in Table A.5; as an example that could be followed in the new ETL framework implementations.

Looking at the XML Schema definition file (etl.xsd), and based on general XML Schema standards (W3C, 1999; Wolter, 2001; Cleveland, 2002; Klmek *et al.*, 2010),

many important features can be seen that allow the XML structure to be defined, and these are:

- a. The .xsd file is a well formed XML document
- b. The root element of any XML files adopting this schema must be called "etl".
There is only one root element allowed.
- c. The root element "etl" is a complex type. In XML Schema Definitions, types are defined as either simple or complex (Barai *et al.*, 2008). Simple types are types such as integers, strings, dates. Complex types are data structures built up from simple types and other complex types (W3C, 1999).
- d. The complex type defines that the elements within it must appear in the order in which they are specified within the XSD file. If the elements in the XSD file are not specified in the same order, then the XML file will not validate correctly.
- e. In this example, the first element in the "etl" type is the "name" which is a string, and the second element in the "etl" type is the "datetime" code which is a date.

This is just an abstract briefing of the features of XML Schema and the benefits which it provides to ETL tools that are built based on the new framework.

5.4 Orchestration Point Specifications

Since the restructuring process of the new ETL framework is highly dependent on SOA, therefore, based on SOA standards (Barai *et al.*, 2008), it is compulsory to orchestrate the services available in the ETL through creating an orchestration point.

It establishes a common point of integration for other components or applications, which enables an implemented orchestration point to be the main integration point. In addition, the orchestration point leads to an increase in the ETL flexibility because:

- The workflow logic encapsulated by an orchestration can be modified or extended in a central location (Lam & Minsky, 2010).
- Positioning an orchestration centrally can significantly ease the merging of ETL processes by abstracting the interfaces that tie the corresponding automation solutions together (Lam & Minsky, 2010).

By establishing a service-oriented integration for ETL architecture, orchestration can support the evolution of ETL framework, because, it is the main factor of any successful ETL tool based on the new ETL framework that contains various components based on different computing platforms. In addition, it can be said that the orchestration is the heart of SOA-based ETL because it establishes means of centralizing and controlling a great deal of inter and intra-ETL logic through a standardized service model. Furthermore, it expresses a body of business process logic that is typically owned by a single ETL tool. Furthermore, it establishes a business protocol that formally defines a business process definition.

The workflow logic within an orchestration is broken down into a series of basic and structured activities that can be organized into sequences and flows. These sequences and flows are explored in Figure 5.2 of section 5.1.

In creating the orchestration point, the whole creation process is up to the ETL designer except some compulsory features which are:

- a. *Create reusable partner services:* make sure that partner services have to be reusable and can be used across different orchestration processes. For example, the *extract* operation of an ETL web service can be used by other orchestration processes in other ETL implementations of the same framework. This is to achieve the reusability feature of the new ETL framework (Newcomer & Lomow, 2004; Lam & Minsky, 2010).
- b. *Asynchronous communication:* Coordinate asynchronous communication between different web services. For example, an ETL administrator's request for the extraction functionality can be done without time sequence limitations, and the same for other ETL services. This helps in achieving the loose coupling of the new ETL framework (Lam & Minsky, 2010).
- c. *Data manipulation:* Data can be manipulated before exchanging between different services. The orchestration process can check, verify, and modify data from the client before sending requests to partner services, which simplifies the administration and troubleshooting of the ETL (Lam & Minsky, 2010).
- d. *Conditional and parallel processing is possible with orchestration services:* If an extraction is already done, the orchestration process can send the request to another ETL partner service, which supports the automation process of the ETL (Almeida *et al.*, 2006; He *et al.*, 2007).

5.5 Specifications of the Composition of Partners and Configurations Based on SOA

Based on SOA standards (Jerstad *et al.*, 2005; Sneed, 2006; Barai *et al.*, 2008; Maurizio *et al.*, 2008), after developing all the components of the ETL; based on the new ETL framework, these components must be composed in one application. This application has to be deployable as a composite ETL tool ready to be executed and used by ETL administrators (clients).

A composite application is an accepted solution that addresses a specific business problem by bringing together business logic and data sources from multiple systems (Barai *et al.*, 2008; Hau *et al.*, 2008). Typically, a composite application is associated with ETL business processes, and may bring together several process steps, presenting them to the client (ETL administrators) through a single interface that is customized to suit the requirements of the ETL business needs.

SOA describes a category of composite applications composed of service provider and service consumer components that distributes business logic and offers location transparency for the service providers and consumers (Jerstad *et al.*, 2005; Hau *et al.*, 2008). The SOA approach allows the replacement or upgrading of individual components in the application without affecting other components.

Figure 5.1 is the basic architecture specification for the way of composing the ETL application. The numbers from (1) to (9) refer to the flow of actions done starting from (1) the client request to execute an ETL functionality and finishing by (9) the confirmation message to the client that his request is done. It is up to the ETL designers to choose the tools, languages, application servers and containers to design

their own ETL tool, but the specifications shown in Figure 5.1 should be considered to meet the framework requirements. Based on this, any ETL tool (application) created using the new ETL framework; should be finally deployed as a composite application in the Service Container (SC). The SC Runtime Environment is provided and supported by most of application servers available these days. SC is integrated with the application server as a pre-configured lifecycle module, which means that whenever the application server's instance starts up, the SC runtime will be available.

There is no user interaction required to configure or start the SC runtime. It is just like any other service of the application server. As shown in Figure 5.3, the Service Engine acts as the bridge between ETL components and SC. The distributed modules can be packaged in an SC composite application and deployed as one single entity. SC is based on a web services model in most of the available application servers (Barai *et al.*, 2008), and provides a pluggable architecture for a container that hosts service producer and consumer components. ETL administrators (client) and ETL services connect to the container via binding components or can be hosted inside the container as parts of a service engine.

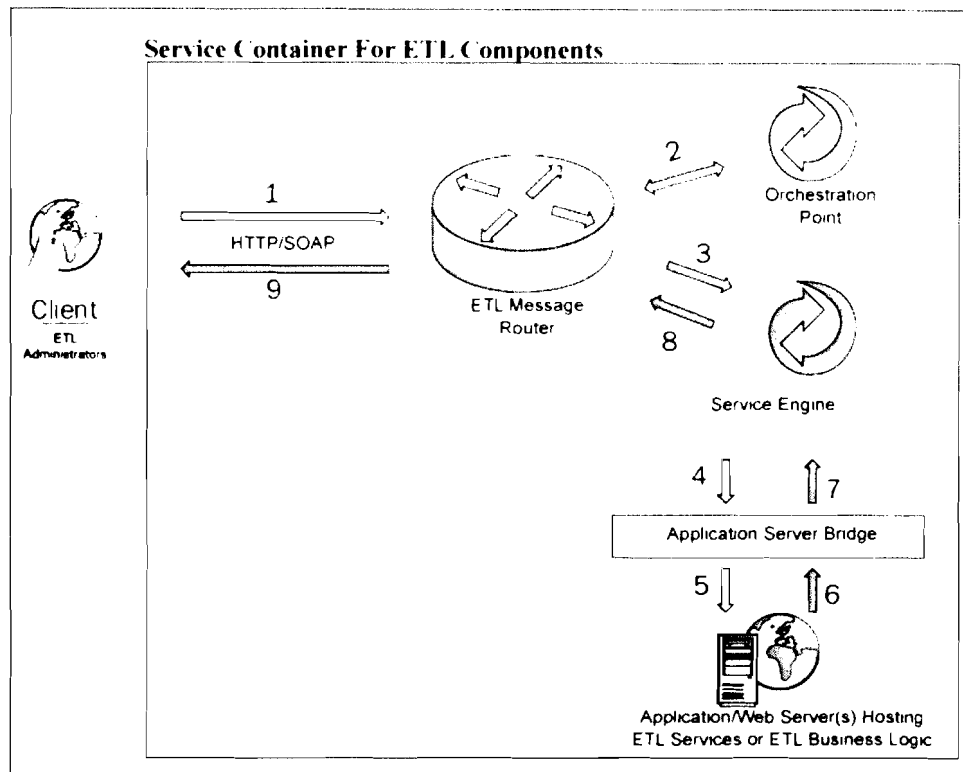


Figure 5.3: ETL Composition Architecture Adopted From (Salter & Jennings, 2008)

Furthermore, a Service Container (SC) is a container for structuring business integrated components. It defines an environment for plug-in components that interact using a services model based directly on WSDL (Barai *et al.*, 2008). A SC defines a packaging for ETL composite applications that are composed of ETL service consumers and providers. Individual service units are deployable to components. Groups of components are gathered together into a service assembly. The service assembly includes meta-data for combining the service units together, as well as binding service units to external services. This provides a simple mechanism for performing composite application assembly using services.

5.6 Specifications of Extending the New ETL Framework

Sometimes, there is a need to add an additional component to the basic ETL framework to meet special business needs other than extraction, transformation and loading. In this case, the new ETL framework provides the extensibility feature to make it possible and simple to extend the framework to suit these special business needs. Then, if it is needed to add a new component to the basic ETL framework, there are few steps to be followed, which are:

- a. Prepare a distributable module for the extra component.
- b. Design and develop the business logic of the extra component.
- c. Encapsulate the extra component in a Web service.
- d. Register the Web service in the orchestration point as a partner service.
- e. Reconfigure the orchestration configuration files to suit the new extra component.
- f. Reconfigure the composite ETL application to involve the new extra component.
- g. Redeploy the whole stuff and test the application again to assure that other components are not affected by the extra one.

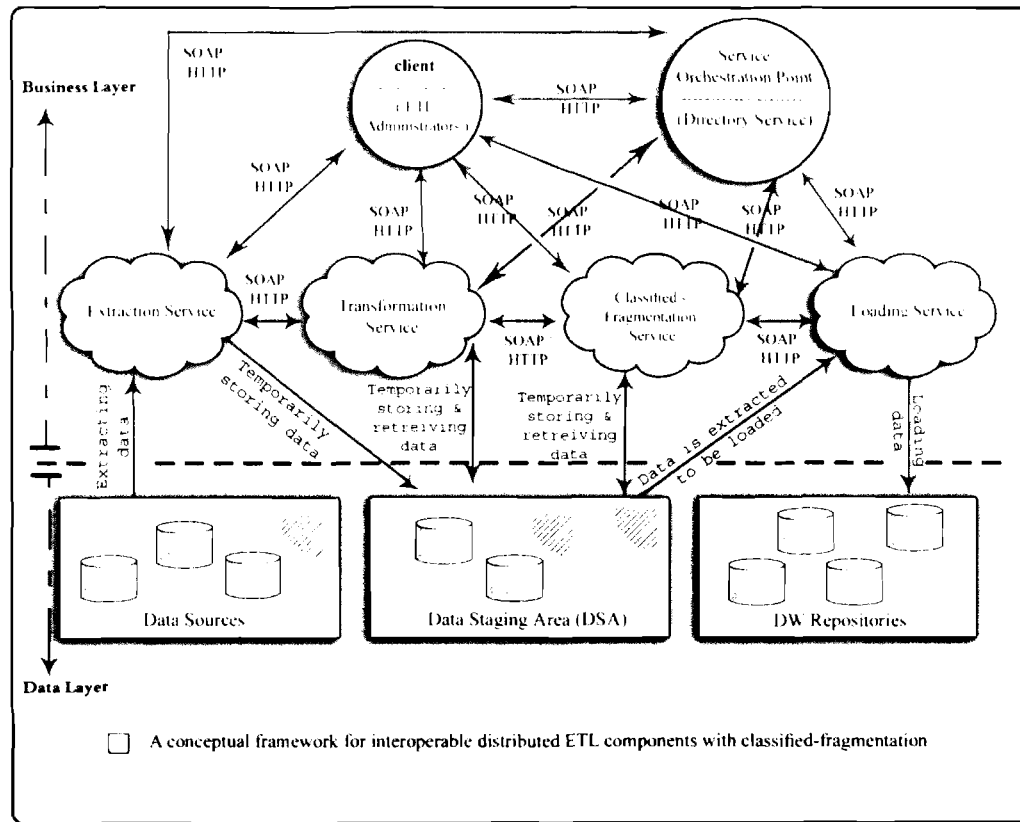


Figure 5.4: A Framework for Interoperable Distributed ETL Components with Classified - Fragmentation

An additional component to extend the basic framework is added. This component is a classification-fragmentation component that is added as an additional component to the enhanced ETL framework to speed up the report generation, and to show the simplicity and flexibility in adding any new component as an extension to the restructured ETL framework without affecting other components. These are the distribution and interoperability features of the framework, which leads to loosely-coupled ETL components. This allows easy to reuse, expand, extend, or add any new component to any loosely-coupled software framework (Roach *et al.*, 2008; Laskey & Laskey, 2009). As shown in Figure 5.4, after adding the classified-fragmentation

component, the framework is the same as Figure 5.1 except the new loosely-coupled classified-fragmentation component and its relations with other components and with the orchestration point.

Chapter 6 includes a complete prototype as a validation to the theoretical framework. A small part of this prototype implements the classified-fragmentation component based on the framework of Figure 5.4, and on the specifications provided in this section. This part of the prototype is developed as a proof of concept to prove that by following the framework standards, any other components can be added to the framework without any complications. Prior to developing the prototype, a meta-model for the new ETL framework is designed to specify the abstract design of the framework.

5.7 Meta-Model for the New ETL Framework

The new ETL framework meta-model is designed in this section to support and represent the new ETL framework and its concepts. A modeling language can be either a graphical or textual language (Kobryn, 2000; Yong Xia, 2002; OMG, 2003).

For the purpose of designing the new ETL framework, Unified Modeling Language (UML) is used as the graphical modeling language (Kobryn, 2000; Lujanmora, 2004), while Extended Backus-Naur Form (EBNF) is also used as a textual modeling language to support the understanding of the graphical meta-model by following its flexible features and symbols of describing meta-models textually (ISO, 1996; Yong Xia, 2002; OMG, 2003; Gargantini, 2007).

UML is formally defined by a meta-model (or semantic model) and it is used to represent software design since it is widely used for modeling both research and industry works (Kobryn, 2000; Kruchten & Selic, 2001; Atkinson, 2002; OMG, 2011).

Object Management Group (OMG) has defined the Common Warehouse Meta-model (CWM) (OMG, 2003) that provides specifications to implement low level DW processes but it does not provide specifications to the component level design (OMG, 2003). On the other hand, UML provides notations for specifying the packaging of a logical design into components that represent a distributed computing architecture and this can be modeled using UML component diagram (Kobryn, 2000; Lujanmora, 2004; OMG, 2011). Furthermore, according to (Lujanmora, 2004), several approaches have been proposed to model different aspects of a DW, such as the conceptual model of the DW, the design of the ETL processes, and the derivation of the DW models from the enterprise data models. However, few efforts have been dedicated to the modeling of the components' physical design of a DW from the early stages of a DW project. (Lujanmora, 2004) used UML component diagram for physical design of DW components to highlight the high level component architecture of the DW projects.

Component diagram supported by other UML notations is used to create a meta-model for the new ETL framework. The components of the meta-model and the relationships among these components are explored in this section. Figure 5.5 represents a meta-model for the new ETL framework.

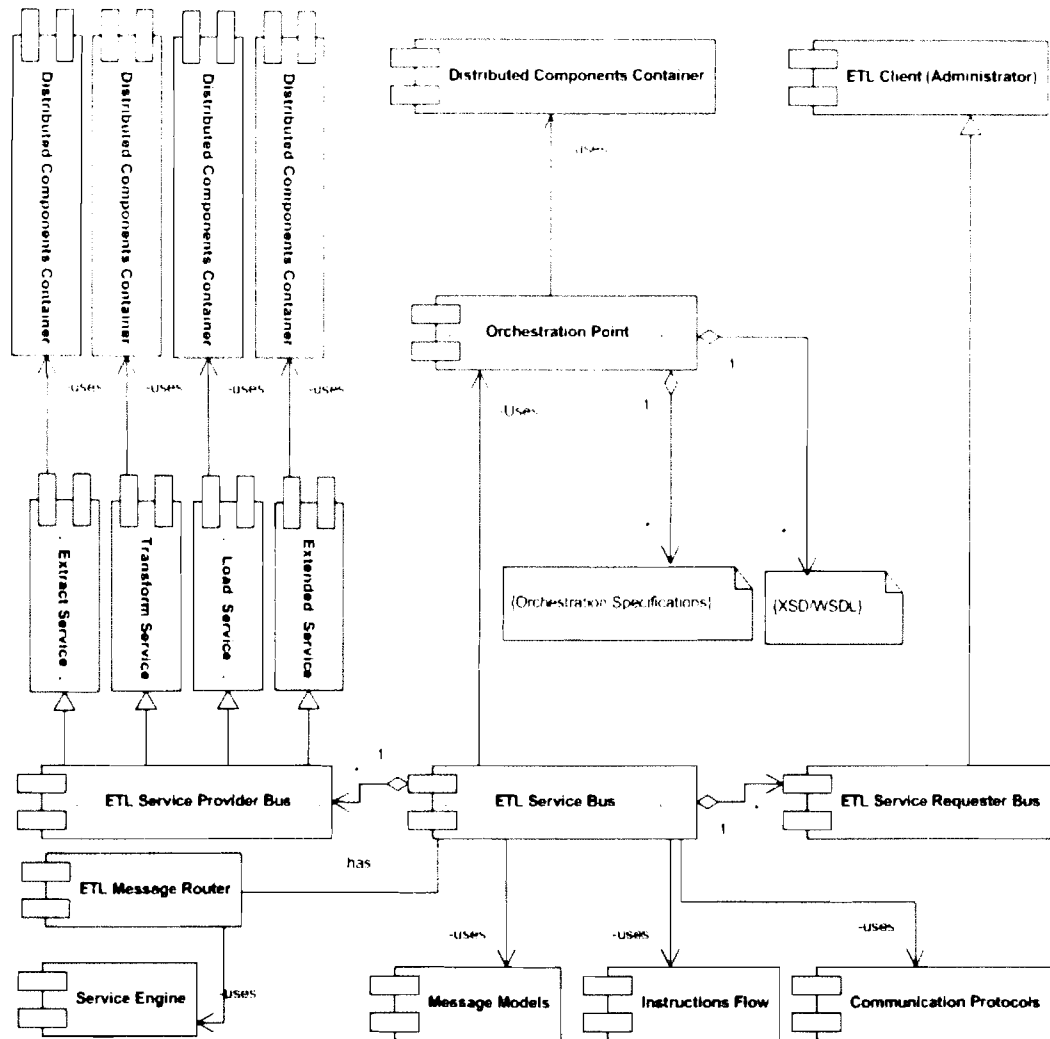


Figure 5.5: Meta-Model for Interoperable and Distributed ETL Framework Components Based on SOA

The complete EBNF definitions for the meta-model shown in Figure 5.5 are listed in section A.3 of Appendix A. This section presents the specific EBNF definitions for the main components of the meta-model. Table 5.1 presents the meaning of some EBNF symbols that are included in the EBNF definitions of the new ETL framework.

Table 5.1: EBNF Symbols Summary (ISO, 1996; Yong Xia, 2002; Gargantini, 2007)

Symbol	Description
<code>::=</code>	The element to the left of the symbol is defined by the constructs on the right.
<code>*</code>	The preceding construct may occur zero or more times.
<code>{...}</code>	The constructs within the curly braces are grouped together.
<code>[...]</code>	The constructs within the square brackets are optional.
<code> </code>	An exclusive OR.

The new ETL framework is composed of main components. These are: Extraction component, Transformation component Load component, optional extended components, and Orchestration point.

```
ETL ::= Extract_Service ,
      Transform_Service, Load_Service,
      [Extended_Service],Orchestration_P
oint, ETL_Service_Bus;
```

The E-T-L components are described by the following EBNF definitions followed by the EBNF definitions of the functions every component does regarding data

queries, while the complete EBNF definitions regarding these ETL functions is included in section A.3 of Appendix A:

```
Extract_Service      ::= Extraction_Queries, (Distrib
                        uted_Component_Container |
                        User_Distribution_Settings)
                        ;
```

```
Transform_Service    ::= Transformation_Queries, (Dis
                        tributed_Component_Containe
                        r |
                        User_Distribution_Settings)
                        ;
```

```
Load_Service         ::= Loading_Queries, (Distribute
                        d_Component_Container |
                        User_Distribution_Settings)
                        ;
```

```
Extended_Service     ::= [Query], (Distributed_Compon
                        ent_Container |
                        User_Distribution_Settings)
                        ;
```

```
Extraction_Queries   ::= extract_from_dataSource,
                        update_to_DataStagingArea;
```

```
Transformation_Queries ::= extract_from_DataStagingAre
                        a,
                        update_to_DataStagingArea;
```

```
Loading_Queries      ::= extract_from_DataStagingAre
```

```

a,
update_to_finalDwRepository
,
delete_temp_DataStagingArea
_data;

```

In addition, the syntax of Orchestration Point component is specified by the following EBNF definitions:

```

Orchestration_Point      ::= Orchestration_Specifica
                             tions,{Orchestration_Sp
                             ecifications},
                             XSD_WSDL,
                             {XSD_WSDL},(Distributed
                             _Component_Container |
                             User_Distribution_Setti
                             ngs);

```

The new ETL framework components exchange data and messages through an enterprise service bus as shown in Figure 5.5. An enterprise service bus represents an environment designed to foster sophisticated interconnectivity between services. It establishes an intermediate layer of processing that can help overcome common problems associated with reliability, scalability, and communications disparity (Weerawarana et al., 2005; Salter & Jennings, 2008). The ETL components of the new ETL framework use the application server service bus to interchange messages

and data among ETL components. This service bus is mentioned as ETL_Service_Bus in the EBNF description of this meta-model.

The ETL Service Bus component uses the Orchestration Point component as a dynamic look-up mechanism to provide information about ETL services. The Orchestration Point component thus enables optimized access to service meta-data and management of service interactions and policies. It also supports the integration of other standard registries and repositories. In its most elemental state, it is composed of service meta-data artifacts documents, such as XML Schema Definition Language (XSD) or Web Services Description Language (WSDL) files. These service meta-data files are managed by the Orchestration Point component. The syntax of The ETL Service Bus component and other components that have direct relations with ETL Service Bus component are specified by the following EBNF definitions:

```
ETL_Service_Bus ::= {ETL_Service_Provider_Bus |
                    ETL_Service_Requester_Bus},
                    ETL_MessageRoute,
                    Message_Model ,
                    {Message_Model},
                    Instructions_Flow,
                    Communication_Protoco,
                    {Communication_Protocol},
                    Orchestration_Point;

ETL_Service_Provider_Bus ::= {Extract_Service |
```

```

Transform_Service |
Load_Service |
[Extended_Service] |
{Extended_Service});

ETL_Service_Requester_Bus ::= ETL_Client_Web_Service,
                               {ETL_Client_Web_Service};

```

The ETL Service Bus component represents the connectivity of the new ETL framework components, and it aggregates with ETL Service Provider Bus and ETL Service Requester Bus components to control the flow of messages and instructions between the ETL Service Requester and the ETL Service Provider. In addition, it provides support to communication protocols used for the communication among the distributed ETL framework components. The ETL Service Bus component uses Message Models, Instructions Flow, and Communication Protocols components. The Message Models component enables The ETL Service Bus to support different types of message models flowing between the service provider and service requestor. The syntax of Communication Protocols component is specified by the following EBNF definition:

```

Communication_Protocol ::= (SOAP | HTTP),
                             (Request_Response |
                             Publish_Subscribe |
                             Synchronous_Asynchronou

```

s) ;

The Communication Protocols component provides support for different communication protocols, such as SOAP and HTTP to connect the service providers with the service consumers (The communication protocols support several interactions patterns, such as request/response, publish/subscribe, and synchronous/asynchronous.). The Instructions flow component contains interfaces to invoke the mediation flows to perform mediation between ETL service requestor and ETL service provider, and to provide references to external services.

The syntax of ETL Message Router component is specified by the following EBNF definitions:

```
ETL_MessageRouter ::= [Service_Engine],  
                        {ETL_Client_Request,  
                        Input_Extraction,  
                        Response_Redirect};
```

The ETL Service Bus component has the ETL Message Router component routes requests to specific business services based on defined criteria. It's conditional routing, and it usually requires the extraction of some piece of data from the message upon which routing can be based.

There are three steps that are done in the ETL Message Router component:

1. ETL Client sends request to the router
2. The router extracts the appropriate input
3. The router directs the response to the appropriate ETL service based on the value of the input extracted in step 2.

In addition, ETL Message Router component optionally uses Service Engine component to provide business logic and transformation services to ETL components, as well as consume ETL services.

The Orchestration Point component and ETL distributed components use Distributed Component Containers as run environments. The container in its general definition is the interface between component functionalities and the application server functionalities that support the component (Armstrong *et al.*, 2004). In the field of distributed application development, all of the application servers released by the popular vendors like Microsoft, Oracle, Sun Microsystems and IBM; include containers for distributed components (Coulouris *et al.*, 2001). If the open source GlassFish Application server (Sun-Microsystems, 2010) that supports J2EE distributed components is taken as an example, then the distributed components are EJB (Enterprise JavaBeans) components that are deployed to the EJB container. The EJB container handles these functionalities:

- The J2EE security model that allows the configuration of a web component or a distributed enterprise bean component so that system resources are accessed only by authorized users.

- The J2EE transaction model that allows specifying relationships among methods those make up a single transaction so that all methods in one transaction are treated as a single unit.
- Lookup services that provide a unified interface to multiple naming and directory services in the enterprise so that application components can access naming and directory services.
- The J2EE remote connectivity model that manages low-level communications between clients and enterprise beans. After an enterprise bean is created, a client invokes methods on it as if it were in the same virtual machine.

Because the J2EE architecture provides configurable services, distributed components within the same J2EE application can behave differently based on where they are deployed. For example, an enterprise bean can have security settings that allow it a certain level of access to database data in one production environment and another level of database access in another production environment. Figure 5.6 shows the Distributed Components container (EJB container) position in the open source GlassFish application server as an example to Distributed Components containers in application servers. As shown in Figure 5.6, The J2EE GlassFish server provides EJB and web containers. Enterprise JavaBeans (EJB) container manages the execution of distributed enterprise beans components for J2EE applications. Enterprise beans and their container run on the J2EE server. On the other hand, the Web container manages the execution of JSP page and servlet components for J2EE applications.

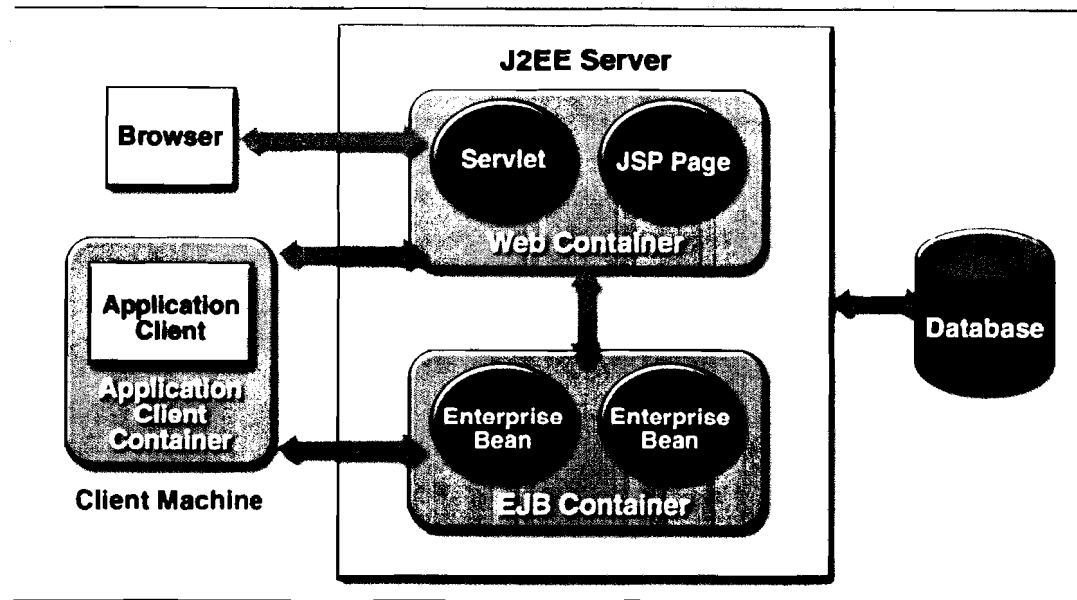


Figure 5.6: GlassFish Server and Containers (Armstrong et al., 2004)

5.8 Feedback from the Experts Regarding the Theoretical Framework

In addition to validating the theoretical framework by the prototype that was designed, developed and evaluated based on the theoretical framework specifications and discussed in chapter 6 and chapter 7, this section discusses the feedback and opinions of experts regarding the theoretical framework using structured interviews. This evaluation was conducted to validate the theoretical solution of the highlighted problems and the advantages of this solution before developing the prototype.

In addition, it consolidates the theoretical steps and specifications of the framework before the prototype development. These interviews were with experts from five companies which are: Sybase, SAS (Statistical Analysis System), TM (Telekom Malaysia), EIGC (Emirates Industrial Gases Co.), and PIT (Professional Information Technology Co.). The number of respondents (N) is 6. Two respondents from SAS

Company and one respondent from each other company. Details of the interviews are attached in Appendix C.

The main assumption during these interviews is that the expert opinions regarding the theoretical framework solution and advantages depict the expected actual solution and advantages of the new ETL framework. The feedback collection method for consolidating the theoretical solution and its advantages through the expert opinions; is a structured interview questionnaire covering basic aspects of the problems (section 2 of the questionnaire), solution (section 3 of the questionnaire) and advantages of the solution (section 4 of the questionnaire).

The results of the structured interview questionnaire are analyzed using SPSS. N, Mean and Std. Deviation were the three fields for analyzing the results. N represents the number of respondents involved in the structured interview questionnaires. Mean represents the average of all responses regarding a certain point or it is the sum of all of the data values of the corresponding questionnaire item divided by the number of these data values. Standard deviation (Std. Deviation) is the average distance between any score in a distribution and the mean of the distribution.

Table 5.2 shows the result of experts' satisfaction level for the problems of the traditional ETL framework. The number of respondents (N) is 6, while the Mean and Std. deviation are shown for every point. The total mean is 3.81 (given that the likert scale used in designing the structured interview questionnaire is a 5-point scale as shown in appendix C).

Table 5.2: Problems Satisfaction

	N	Mean	Std. Deviation
PR1	6	3.50	1.225
PR2	6	3.83	0.408
PR3	6	3.67	0.516
PR4	6	3.67	0.816
PR5	6	4.33	0.816
PR6	6	3.67	0.516
PR7	6	3.83	0.408
PR8	6	4.00	0.632
Valid N (listwise)	6	3.81	

Table 5.3 shows the result of experts' satisfaction level for the solutions proposed for the problems of the traditional ETL framework. The number of respondents (N) is 6, while the Mean and Std. deviation are shown for every point. The total mean is 3.73 (given that the likert scale used in designing the structured interview questionnaire is a 5-point scale as shown in appendix C).

Table 5.3: Solutions Satisfaction

	N	Mean	Std. Deviation
SO1	6	4.00	0.000
SO2	6	3.67	0.516
SO3	6	3.33	0.516
SO4	6	4.00	0.632
SO5	6	3.67	0.516
SO6	6	4.00	0.632
SO7	6	3.67	0.516
SO8	6	3.50	0.548
Valid N (listwise)	6	3.73	

Table 5.4 shows the result of experts' satisfaction level for the advantages of the new proposed ETL framework over the traditional ETL framework. The number of respondents (N) is 6, while the Mean and Std. deviation are shown for every point.

The total mean is 3.77 (given that the likert scale used in designing the structured interview questionnaire is a 5-point scale as shown in appendix C).

Table 5.4: Advantages Satisfaction

	N	Mean	Std. Deviation
AD1	6	3.50	0.548
AD2	6	3.67	0.816
AD3	6	4.00	0.000
AD4	6	3.67	0.516
AD5	6	4.17	0.408
AD6	6	3.33	0.516
AD7	6	3.83	0.408
AD8	6	4.00	0.000
Valid N (listwise)	6	3.77	

The experts' feedback shows that they agree that the problems presented in this research are existing problems that need to be solved. In addition, as shown in the structured interview questionnaires attached in Appendix C and in the results presented in this section, majority of the experts agree on the main points of the theoretical solution and its advantages.

5.9 Conclusion

This chapter has listed mandatory specifications and recommendation for the conceptual (theoretical) framework for interoperable distributed ETL components. These specifications have covered distribution, Web services involvement, orchestration point, composing the ETL components, and extending the framework. In addition, a meta-model using both UML and EBNF is designed abstractly for consolidating the design of the new ETL framework. Furthermore, this chapter included experts' feedback regarding the theoretical framework (before doing the prototype) through structured interviews with experts from the industry. Chapter 6

discusses the development of an SOA-based ETL prototype based on the newly defined theoretical framework.

CHAPTER SIX

SOA-BASED ETL PROTOTYPE

Chapter 5 has specified the conceptual (theoretical) framework for interoperable distributed ETL components. The specifications of the new framework constitute a base and a blueprint for developing ETL tools based on the new framework. Based on these specifications, this chapter discusses the development of an SOA-based ETL prototype, which was based on the new theoretical framework. Scrum methodology was adopted for the development of the SOA-based ETL prototype.

6.1 Analysis

Requirement analysis is the first activity that was performed in the life cycle of this prototype development to determine the whole backlog of the prototype. The method considered for gathering the requirements of the prototype is the theoretical framework specifications and the meta-model explored in chapter 5.

Scrum methodology includes the term “Product Owners” (Sutherland *et al.*, 2007; Marchenko & Abrahamsson, 2008; Sutherland *et al.*, 2008; Uy & Ioannou, 2008). The product owners in this research prototype are industry experts. Once those product owners (industry experts) have an idea about ETL framework by working with DW customers, they have the ability to determine the requirements for the basic ETL activities. Based on that, the feedback of the industry experts is considered when the requirements are determined based on theoretical framework specifications and the literature.

Considering priority, the whole requirements (product backlog) of the prototype are divided through several rounds (sprints) by the team. The complete clarity of the requirement items is reached at the starting point of every sprint. Furthermore, in the first part of the sprint planning meeting, the requirement analysis continues, and sometimes, the requirements are well understood by that time. However, since items are selected in the sprint planning, then, the team is sure that they are working on those specified items only for the current sprint. The team has identified further points to be clarified such as exact scope and sad path scenario. Moreover, the requirement analysis activity did not stop in the first part of the sprint planning meeting, but continued in the second part and during the actual sprint. In the second part of the sprint planning meeting, while the ETL tasks were decomposed, sometimes, the complexity in requirements appeared again and has been clarified.

During the sprint, each of tests, requirement discussion while designing and coding and discovery from testing needed some additional elements of requirement analysis. Furthermore, in every sprint review, some additional requirements appeared through feedback and suggestions.

6.1.1 Requirements (Prototype Backlog) Determination

The whole prototype backlog (requirements) is the development of an SOA-based ETL prototype that is based on the theoretical framework described in chapter 5. This prototype periodically extracts data from source systems, transforms the data into a common format, and then loads the data into the target data store, usually a data warehouse. ETL processes bring together and combine data from multiple source systems into a data warehouse enabling all users to work on a single

integrated set of data. This process is to be done in a distributed and interoperable specification of the ETL components. The prototype contains three main components (Extract, Transform and Load) and one extended component (Classified-Fragmentation). The details of these components are as follows:

- a. *Extract*: the process of reading data from a specified source database and extracting a desired subset of data.
- b. *Transform*: the process of converting the extracted data from its previous form into the form it needs to be in, so that it can be placed into another database. Transformation occurs by using rules or lookup tables or by combining with other data.
- c. *Load*: the process of loading the data into the target DW repository.
- d. *Classified-Fragmentation*: a component to be added as an additional component to the enhanced ETL framework to speed up the report generation, and to show the simplicity and flexibility in adding any new component as an extension to the restructured ETL framework without affecting other components.

6.1.2 Backlog Division

After determining the whole product backlog as described in section 6.1.1, the total product backlog is divided into items. Each item (or set of items) is carried out in one sprint which took from 2 to 4 weeks cumulative work of the research effort. The items of this product backlog are estimated or sized, because, knowing the size of the items is a cost indicator. It helps prioritize the product backlog and facilitates planning the prototype. In addition, detailed task-level estimates are determined in

the sprint planning meetings of the research team (the main researcher and his supervisor). Furthermore, tasks and their estimates are captured in the sprint backlog.

The prototype backlog evolved, and its contents changed frequently. New items are discovered and added to the prototype backlog based on experts' feedback. Moreover, existing items are modified, reprioritized, refined, or removed. That is why the prototype product backlog was a dynamic artifact.

Then, all sprints of the prototype backlog were prioritized. The most important and highest-priority sprints were implemented first. These were found at the top of the prototype backlog. Once a sprint was done, it was removed from the prototype backlog. Scrum does not mandate how the prototype backlog is prioritized. However, some prioritization factors are considered by the researcher to be useful, and these are:

- Sprint value: a sprint is considered valuable if it is necessary for achieving functional requirements of the prototype. If that is not the case, then, the sprint is irrelevant; and it is excluded from the top priority sprints. The Scrum team either reprioritizes the sprint and places it at the bottom of the prototype backlog or better, discards it. The second option keeps the prototype backlog simple and the researcher focused.
- Knowledge, uncertainty, and risk: Because risk and uncertainty influence prototype success, uncertain and risky sprints occupied high priorities. This speeded up the generation of new knowledge of ETL functional requirements of the prototype. In addition, it removed uncertainty, and reduced risk.

- Dependencies: Dependencies in the product backlog are an available fact. Functional requirements, for instance, often depend on other functional and even nonfunctional requirements, for example the extraction sprint came before the transformation one.

Prioritization directed the team's work by focusing the team on the most important sprints of the prototype. It also enhances the management of the sprints.

Based on prioritization factors, the whole prototype backlog is divided into sprints in this order:

- a. Database Sprint
- b. Coding ETL Components Sprint
- c. Distributed Components and Web Services Sprint
- d. BPEL Creation Sprint
- e. Sprint of Assembling Prototype Components into One Composite Application
- f. Unit Testing Sprint
- g. Web Service Testing Sprint
- h. Compatibility Testing Sprint
- i. Classified-Fragmentation Speed and Scalability Testing Sprint
- j. End To End Testing Sprint

After dividing the prototype backlog into sprints, the requirements of every sprint are detailed in a meeting at the beginning of every sprint. In the same meeting, the general design of the sprint items is discussed among the research team and detailed design continued in the sprint. The team organized frequent discussion meetings

while coding, and coding itself led to include some design aspects. The design is considered done only when the sprint is done successfully. As mentioned in section 6.1.2, database sprint occupied the highest priority among other sprints. Section 6.2 describes in details the database design, creation, and other query aspects.

6.2 Database Sprint

The health domain is chosen for the prototype, because normally health centers have huge numbers of records in their databases (Patasiene *et al.*, 2007; Badoiu *et al.*, 2008; Maskat & Shamsudin, 2008; Zhou *et al.*, 2008) especially historical data (Katifori *et al.*, 2008). This data are usually used in data warehouse and business intelligence activities (Bugatti *et al.*, 2008; Johansen *et al.*, 2008; Roy *et al.*, 2008; Theodosi & Tsihrintzis, 2008; Information, 2009).

This prototype uses a CLINIC database schema. This database was used as a sample for developing, deploying and testing the ETL components of the new ETL framework. The CLINIC database consists of a number of tables (such as PATIENTS, PHYSICIANS, DEPARTMENTS, TESTOPERATIONS, DISEASE, MEDICINES, GENDER and CITY). These tables store data for patients, physicians, test operations, and other related data. The detailed explanation about every table and every field in this schema is described in Appendix A.

During the execution of the ETL processes, necessary databases are generated. These databases could be prepared before the execution, but in this prototype, the databases generation is automated to make the testing process more simple and clear. The generated databases are: Figure 6.1 (CLINIC database), Figure 6.2 (EXTRACT

TEMP STORAGE database), Figure 6.3 (TRANSFORM TEMP STORAGE database), Figure 6.4 (CLASSIFICATION database) and Figure 6.5 (LOAD database).

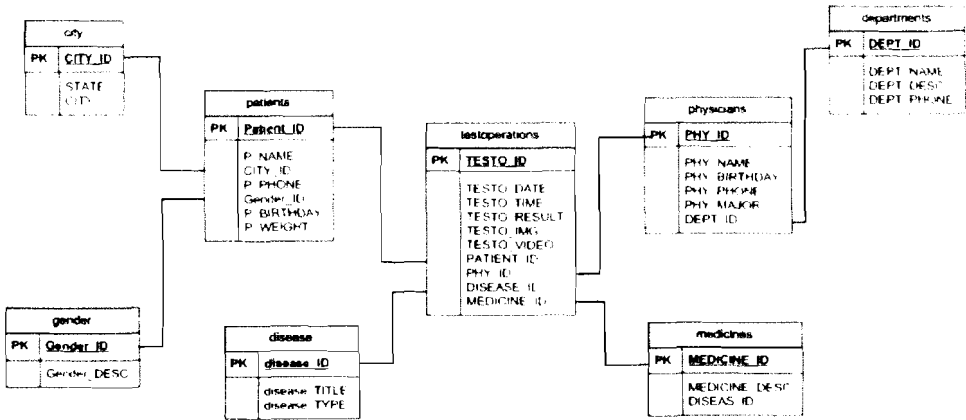


Figure 6.1: CLINIC Database

a. *EXTRACT TEMP STORAGE database*

An EXTRACT TEMP STORAGE database is a temporary storage database, which is generated after the extraction process of CLINIC database tables. The schema of this database is a star schema that consists of one fact table, and four dimension tables. The detailed explanation about every table and every field in this schema is described in Appendix A.

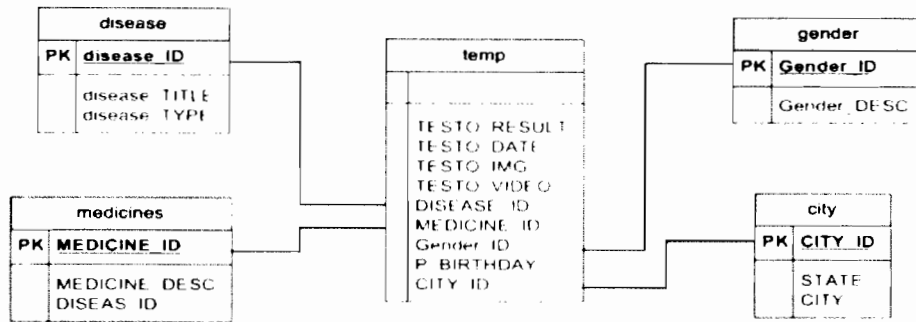


Figure 6.2: EXTRACT TEMP STORAGE Database

b. TRANSFORM TEMP STORAGE database

A TRANSFORM TEMP STORAGE database is a temporary storage that is generated as a result of the transformation process done on the EXTRACT TEMPSTORAGE database. The schema of this database is a star schema that consists of one fact table, and four dimension tables. The detailed explanation about every table and every field in this schema is described in Appendix A.

The transformation of data that is done as a sample of transforming data; is to transform P_BIRTHDAY to P_AGE.

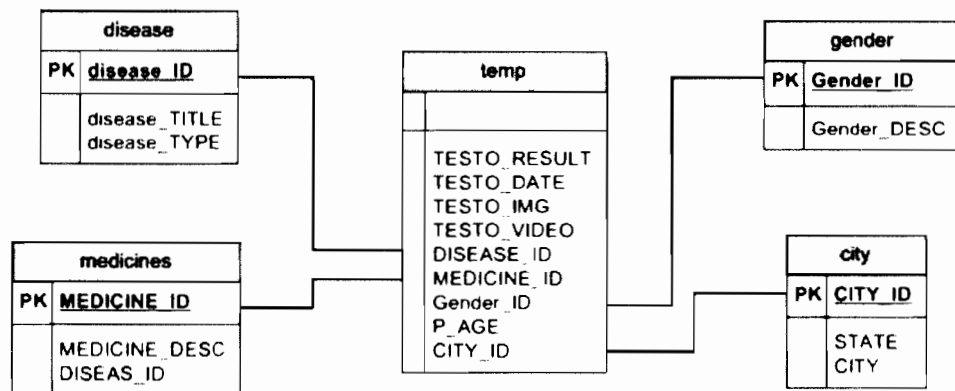


Figure 6.3: TRANSFORM TEMP STORAGE Database

c. *CLASSIFICATION database*

A CLASSIFICATION database is a temporary storage database, which is generated after the classification process of TRANSFORM TEMP STORAGE database tables. The schema of this database is a star schema that consists of six fact tables, and four dimension tables. During the classification process, tables are fragment based on date and type. The detailed explanation about every table and every field in this schema is described in Appendix A.

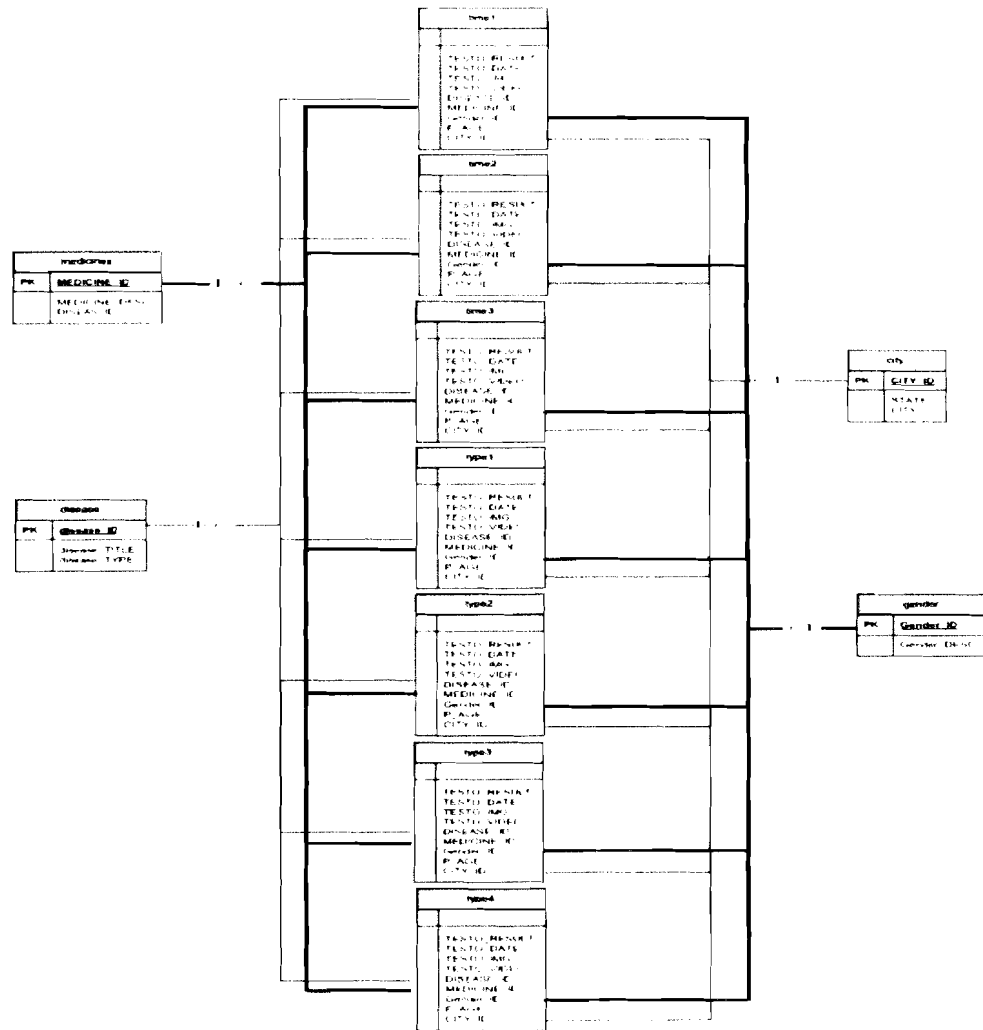


Figure 6.4: CLASSIFICATION Database

d. *LOAD database*

A LOAD database is a final storage that is generated after load process was executed on CLASSIFICATION database tables. The schema of this database is a star schema that consists of a number of fact and dimension tables depending on the fragmentation criteria. The detailed explanation about every table and every field in this schema is described in Appendix A.

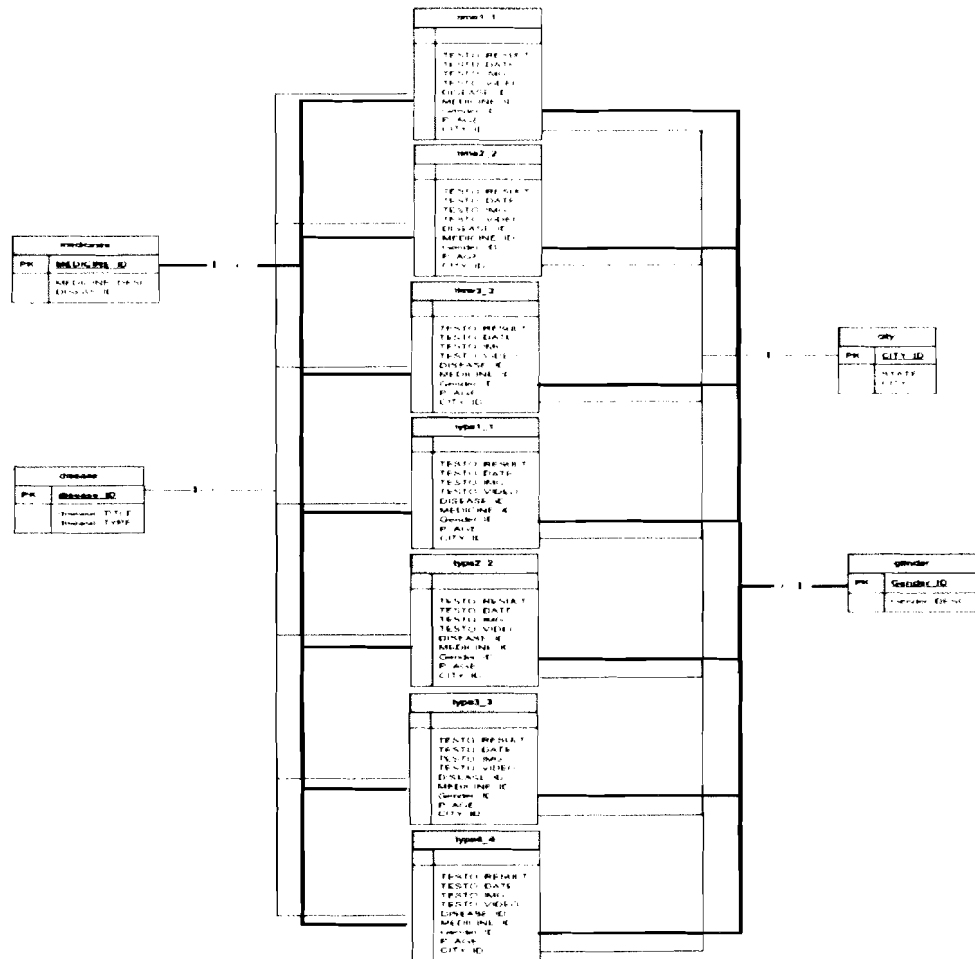


Figure 6.5: LOAD Database

6.3 Coding ETL Components Sprint

After completing the database sprint successfully, this sprint is done to accomplish the business logic coding of the ETL components. Java programming language is chosen as the core language to implement this prototype, because it has some advantages over other competitor languages, since it is an open source and platform independent language (Armstrong *et al.*, 2004). In conducting this sprint, there are two options to meet the theoretical framework specifications: The first is to do coding using JavaBeans and JSP, then to wrap them in an EJB multi tier distributed architecture, while the other option is to code the components completely under EJB specifications. For this sprint, the first option is selected, because it enables the migration of legacy components to the new framework whenever needed (Channabasavaiah *et al.*, 2003). Therefore, the following tools are used to code the business logic of the ETL components:

- Java SE Development Kit (JDK) 6.0.
- MySQL database 5.0.
- Apache Tomcat web Server 5.5.9.
- Jcreator Pro 3.2 IDE.
- Web browsers: Internet Explorer, Mozilla Firefox and Google Chrome.

Before describing the configuration and coding of this sprint, the design of this sprint was performed and is represented by a class diagram as shown in Figure 6.6. Attributes and methods of the class diagram of Figure 6.6 are listed in section A.2 of Appendix A.

The database sprint is completed successfully before starting the coding of this sprint because the database tables created are necessary to deploy and test this sprint. Therefore, it is a must to configure the database connection between the coded components of this sprint and the database. This can be done by going to *config.txt* file that has the JDBC (Java Database Connectivity) connection variables, which is located in WEB-INF folder of the Apache Tomcat web Server. Figure 6.7 is a screen shot of *config.txt* file.

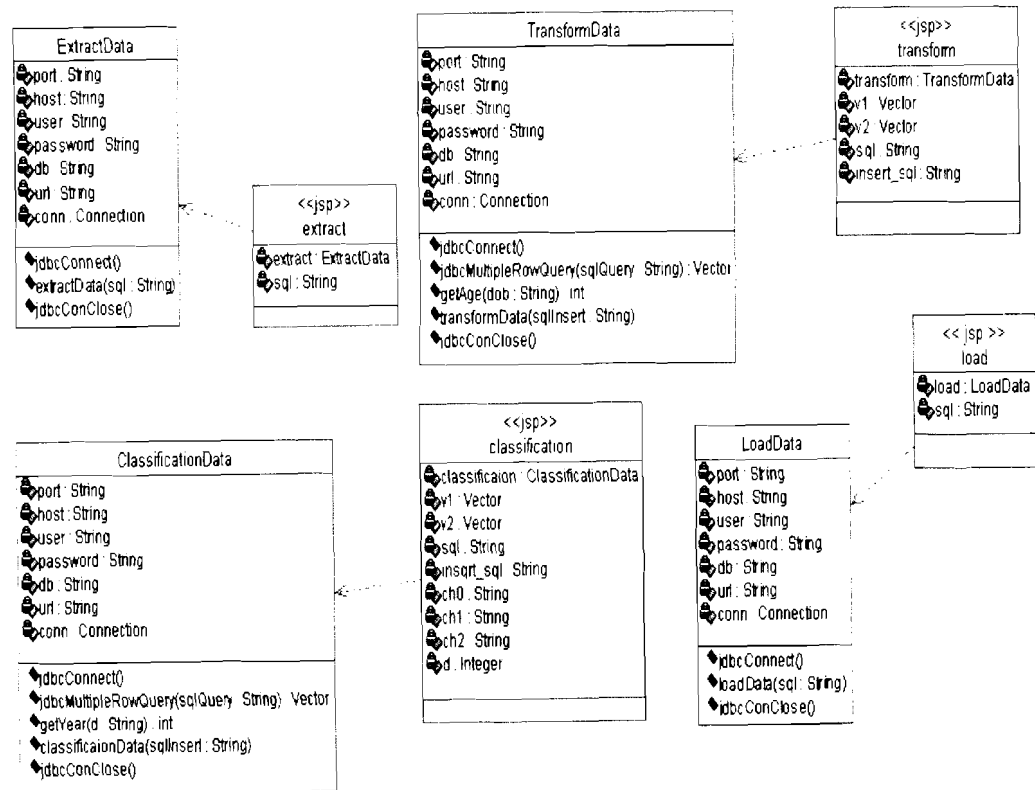


Figure 6.6: Class Diagram of ETL Components

```

// MySQL hostname
host=localhost

// MySQL port
port=3306

// MySQL user
user=root

// MySQL password
password=

// MySQL db-name
db=

```

Figure 6.7: *config.txt (JDBC Connection Variables)*

The *config.txt* file has a copy in each component of the ETL prototype, because every component is done separately as a distributed project to meet the distribution specifications.

6.3.1 Extraction Task

In addition to the class diagram discussed in section 6.3, this section presents the extraction task in more details including a sequence diagram and the coding of this task. The sequence diagram in Figure 6.8 shows that ETL admin can launch the extract page to start extraction operations. First, a connection to a data source has to be established, and then the extraction of data to the extract temp storage has to be done. Then, termination of database connection is done. Finally, " Extract...Done " expression appears. The complete code of core class of the extraction component is available in Appendix B.

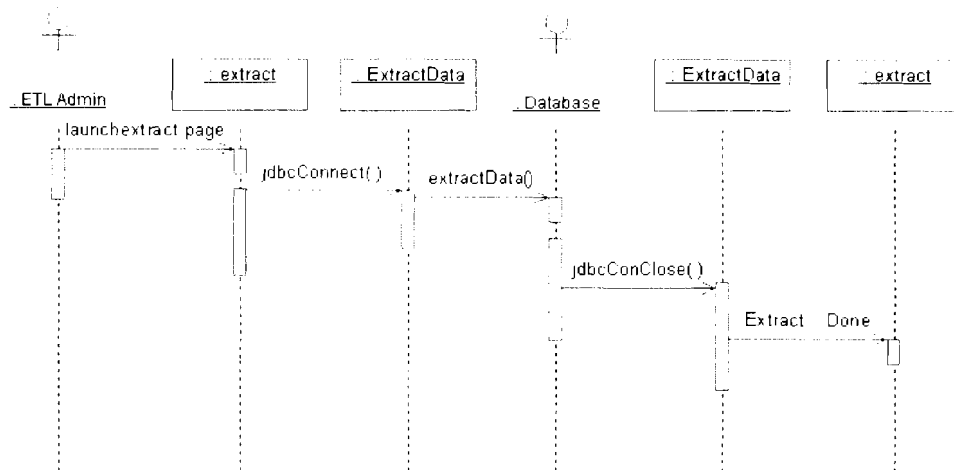


Figure 6.8: Extraction Sequence Diagram

To configure the extraction operation, some steps are done to the `extract.jsp` file that is located in the `jsp` folder of the `extract` project, those steps are:

a. *Creation of extractTempStorage database*

The database name is determined (step (1) of Figure A.1).

b. *Creation of fact table*

A query is written. This query creates the fact table and extracts data from the main data source into the `extractTempStorage` database (step (2) of Figure A.1).

c. *Creation of dimension tables*

A query is written. This query creates dimension (*look up*) tables and extracts its data from the main data source into the `extractTempStorage` database (stage (3) of Figure A.1).

In addition, if a new extract method is needed, it can be added to *ExtractData.java* file, that is located in *\WEB-INF\classes*. The code of this java class is shown in Table B.1 of Appendix B.

6.3.2 Transformation Task

In addition to the class diagram discussed in section 6.3, this section presents the transformation task in more details including a sequence diagram.

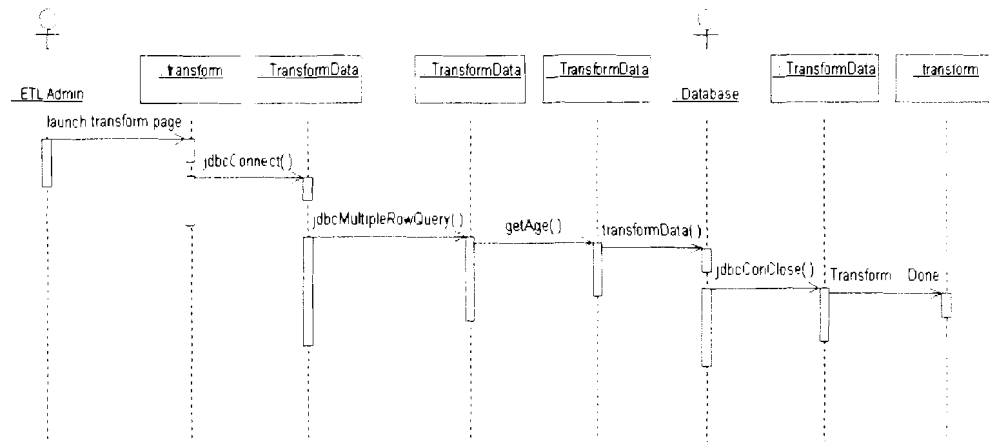


Figure 6.9: Transformation Sequence Diagram

The sequence diagram in Figure 6.9 shows that ETL admin can launch the transform page to start transformation operations. First, a connection to the extract temp storage has to be established, and then a selection of extracted data and a transformation data from “*birthday*” to “*age*” has to be done. Then, then the transformed data is inserted into the transformation temp storage. Followed by

termination of database connection is done. Finally, " Transform...Done " expression appears.

The complete code of core class of the transformation component is shown in Table B.2 of Appendix B.

To configure the transformation operation, some steps are done to the transform.jsp file that is located in the *jsp* folder of the *transform* project, those steps are:

a. *Creation of transform temp storage*

Specifies the database name that is found at the tranformed data (step (1) of Figure A.2).

b. *Creation of fact table*

Creates empty fact table as a copy from Temp fact table, that is located in the extractTempStorage database (step (2) of Figure A.2).

c. *Update fact table*

Alters the empty fact table to change any field name or datatype in order to be compatible with the tranformed data (step (3) of Figure A.2).

d. *Transform data*

Writes a query to select extracted data from the extractTempStorage database, then transform the selected data such as "calculate age from birthday by *getAge()*" method. Then, insert the transformed data into the Temp fact table, which is located in the transformTempStorage database (step (4) of Figure A.2). Also another transform method like *getAge()* can be added by going to *TransformData.java* file that is located in \WEB-

INF\classes and adding the new transform method, then using it at this step.

TransformData.java is available in Appendix B.

e. *Creation of dimension tables*

Writes queries that create dimension (*look up*) tables and copy its data from the *extractTempStorage* database into the *transformTempStorage* database (step (5) of Figure A.2).

6.3.3 Classified-Fragmentation Task

In addition to the class diagram discussed in section 6.3, this section presents the classified-fragmentation task in more details including a sequence diagram.

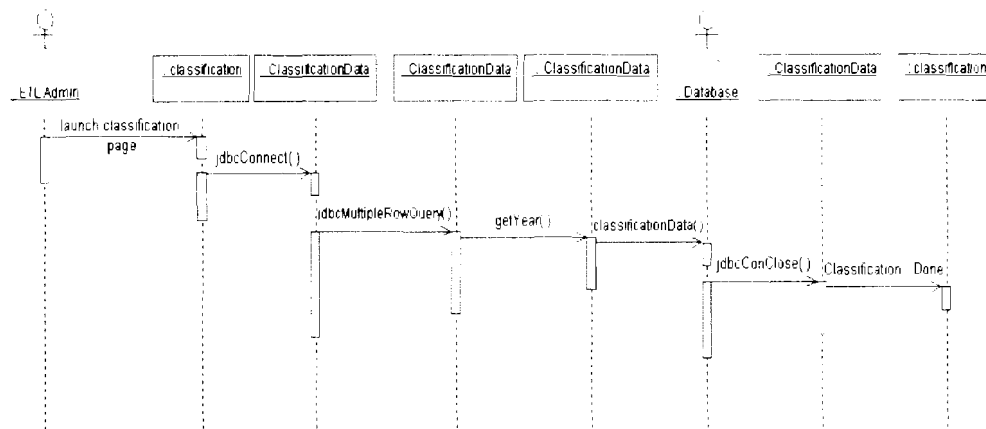


Figure 6.10: Classified-Fragmentation Sequence Diagram

The sequence diagram in Figure 6.10 shows that ETL administrator can launch the classified-fragmentation page to start Classified-fragmentation operations. First, a connection to the transform temp storage is established, and then, a selection of transformed data has to be done, and data classified according to years and type of

data. Then the classified data is inserted into the classification temp storage in different tables according to classification conditions. After that, termination of database connection is completed. Finally, "classified-fragmentation ...Done " expression appears. The complete code of core class of the classified-fragmentation component is shown in Table B.3 of Appendix B.

To configure the classified-fragmentation operation, some steps are done to the classification.jsp file that is located in the *jsp* folder of the Classified-fragmentation project, those steps are:

a. *Creation of classification temp storage*

Specifies the database name that is found at the classified data (step (1) of Figure A.3 (a)).

b. *Creation of fact table*

Creates empty fact tables, as a copy from Temp fact table that is located in the transformTempStorage database. The generation of a number of fact tables according to the classification of data is shown (step (2) of Figure A.3 (a)).

c. *Classifying data*

- To classify data, first of all, creation of a query is needed to select transformed data from the transformTempStorage database (step (3) of Figure A.3 (b)).
- Variables are set with the retrieved data to be prepared for classification (step (3.1) of Figure A.3 (b)).

- In this classification operation, data is classified according to its type, such as, rows include an image or video, and rows do not.
- Insertion of classified data into three different fact tables (step (3.2) of Figure A.3 (b)). Then, data is classified according to another condition, for example, the date of data, such as classifying data that was created before 1970 and inserting it into time1 fact table.
- The `getYear()` method returns the year from the date to use it in the classification operation step (3.3) of Figure A.3 (b)).

d. *Creation of dimension tables*

The creation of queries that generates the dimension (*look up*) tables and copy its data from the transformTempStorage database to the classification database (step (4) of Figure A.3 (b)).

6.3.4 Loading Task

In addition to the class diagram discussed in section 6.3, this section presents the Loading task in more detail including a sequence diagram and the coding of this task.

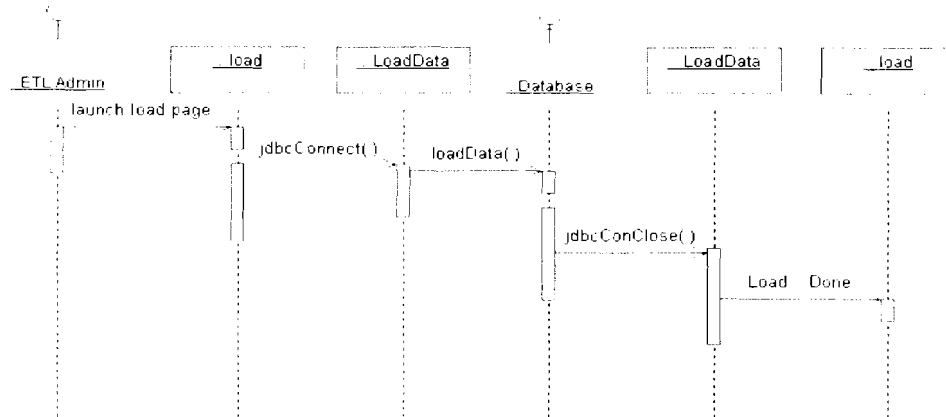


Figure 6.11: Transformation Sequence Diagram

The sequence diagram in Figure 6.11 shows that ETL admin can launch the load page to start load operations. First, a connection to the classification temp storage has to be established. Then, classified data is loaded into load storage. After that, termination of database connection is completed. Finally, " Load...Done " expression appears. The complete code of the core class of the load component is shown in Table B.4 of Appendix B, while the code of the JSP as a presentation layer is described in this section.

To configure the load operation, some steps are done to the load.jsp file that is located in the *jsp* folder of the *load* project, and these steps are:

- a. *Creation of load database*: Specifies the database name (step (1) at Figure A.4).
- b. *Creation of fact tables*: Using queries that creates the fact tables and load its data from the classification database to the load database (step (2) at Figure A.4).

- c. *Creation of dimension tables:* Using queries that creates the dimension (*look up*) tables and loads its data from the classification database into the load data warehouse (step (3) at Figure A.4).

6.4 Distributed Components and Web Services Sprint

Based on the specifications of the theoretical framework, a distributed project is created separately for each component of the ETL prototype. Multi tier architecture was adopted as a base for these projects. The distribution concept is based on the EJB (Enterprise Java Beans) technology, which was built on CORBA and RMI frameworks. GlassFish application server was selected to host the projects and the services of the distributed modules.

After creating a distributed project for each ETL component, a web service is created inside each project to wrap the business logic of the ETL task. Points (a) to (d) show these four web services, their WSDL codes and their XML Schema codes.

- a. *Extract Web service* (The basic code of the web service design is available in Figure A.5)

Based on the theoretical framework specifications, the WSDL code and the XML Schema code of the extract web service are shown in Tables B.5 and B.6 of Appendix B.

- b. *Transform Web service* (The basic code of the web service design is available in Figure A.6)

Based on the theoretical framework specifications, the WSDL code and the XML Schema code of the transform web service are shown in Tables B.7 and B.8 of Appendix B.

- c. *Classify Web service* (The basic code of the web service design is available in Figure A.7)

Based on the theoretical framework specifications, the WSDL code and the XML Schema code of the classify web service are shown in Tables B.9 and B.10 of Appendix B.

- d. *Load Web service* (The basic code of the web service design is available in Figure A.8)

Based on the theoretical framework specifications, the WSDL code and the XML Schema code of the load web service are shown in Tables B.11 and B.12 of Appendix B.

6.5 Business Process Execution Language (BPEL) Creation Sprint

As an implementation to the orchestration point that is a mandatory specification of the theoretical framework listed in chapter 5, the BPEL (Business Process Execution Language) was chosen to implement the orchestration point of the prototype.

The BPEL has rapidly been emerged as a standard for combining a set of services into a number of discrete and long running enterprise processes (Weerawarana *et al.*, 2005; Barai *et al.*, 2008). Most organizations are either using BPEL or planning to use it over their other middleware framework (Barai *et al.*, 2008). The NetBeans IDE (NetBeans, 2010) was used to design and implement the required BPEL

functionalities. In this study, a PBEL was designed and developed using NetBeans IDE and deployed to a separate runtime environment for execution. This runtime is the OpenESB (Sun-Microsystems, 2010) runtime that was integrated with the GlassFish application server.

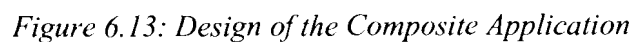
Figure 6.12 shows the design of the BPEL orchestration point based on the theoretical framework specifications. On the left side, a client web service is shown that is eligible for consumption by any ETL administrator application, while on the right side, four partner links for the four ETL web services that are described in section 6.4 are shown. In the middle, the core business logic of the BPEL web service is presented. To have a clearer look at the BPEL orchestration point, its code that is done using XML is listed in Table B.13 of Appendix B.

In Figure 6.12, a client (ETL administrators) Web service can discover services by examining the BPEL. After looking up the required ETL services, the client communicates remotely and directly with any distributed ETL service (Partner Service), in this case the client is the service consumer and the ETL service is the service provider. An ETL service can also be a consumer to another ETL service; in this case it discovers the availability of that service using BPEL service as well.

6.6 Sprint of Assembling Prototype Components in One Composite Application

Based on the theoretical framework specifications, all the components of the prototype are combined in one deployable composite application. In addition to the theoretical framework specifications, the SOA architecture recommends building loosely coupled applications and treating each one of these as independent “service units”. Well-designed composite applications implement this architectural approach by providing an easy way to build business applications. It also provides integration of existing applications with other existing, as well as new applications. This SOA concept of linking together business processes is the hub of composite applications.

There are many tools available today that are used as editors or IDEs for the creation of composite applications. Out of these, NetBeans SOA tools and OpenESB runtime compose proper IDEs for creating and editing the required composite application for this prototype. Figure 6.13 shows the design of the composite application for the prototype parts. It combines the BPEL and other components through the SOAP protocol, and opens a port for each partner link as shown in the same Figure. The final composite application resulted from the design of Figure 6.13 is deployable in the GlassFish application server. The implementation of this design is listed in the code available as shown in Table B.16 of Appendix B.



The testing sprints are done for the purpose of validating and verifying that the prototype meets the requirements that are listed in section 6.1 to validate that the prototype works as expected and can be implemented with the same required characteristics. Figure 6.14 shows the main testing interface that is developed to make testing and executing the ETL components more friendly. This web interface combines all the components of the prototype in one composite application as an interface for the user to do testing and execution of any ETL web service.

6.7.1 Unit Testing Sprint

SOAP Request

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:ExtractOperation xmlns:ns2="http://prototype.phd.awad/">
      <extractParameter>true</extractParameter>
    </ns2:ExtractOperation>
  </S:Body>
</S:Envelope>
```

SOAP Response

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:ExtractOperationResponse xmlns:ns2="http://prototype.phd.awad/">
      <return>Extract Service Invoked ! <p align='center'><b><font size='4'>
    </ns2:ExtractOperationResponse>
  </S:Body>
</S:Envelope>
```

Figure 6.15: GlassFish Tester Result for the Extract Web Service

Unit testing is a method by which individual units of source code are tested to determine if they are fit for use (W3C, 2010). A “unit” in the prototype is the distributed ETL component that is wrapped in a Web service. GlassFish Tester (Sun-Microsystems, 2010) is used to test each Web service individually. The four main Web services passed the test. A screenshot of the SOAP request and response of the Extraction component are shown in Figure 6.15 as a sample of testing results, while the complete testing process is shown in Appendix B.

6.7.2 Classified-Fragmentation Speed and Scalability Testing Sprint

To test the speed of report generation before and after using the Classified-Fragmentation component, and to test that the prototype is scalable for more than one concurrent ETL administrators, the same clinical data of the prototype is used to generate a statistical report. As stated in section 6.2, the data warehouse repository of that clinic consists of a number of tables which are: (PATIENTS, PHYSICIANS, DEPARTMENTS, TESTOPERATIONS, DISEASES, MEDICINES, GENDER and CITY). These tables store data about patients, physicians, test operations, and other related data. The data was classified according to its type (video, text, or image), and then, the original tables are fragmented in which each of the table consists one type of data e.g. video, text, or image. This fragmentation process decreases the number of records in each table, which leads to a faster report generation (Wrembel & Koncilia, 2007).

A report generation speed test was conducted using Apache JMeter tool (Apache, 2010), which is a Java desktop application designed to test software functions and performance, especially for web applications. Testing was done to calculate the time consumed to generate the report shown in Figure 6.16. This testing was done 9 times using fragmented data and 9 times using un-fragmented data, while different numbers of concurrent users are considered. The report shown in Figure 6 16 was re-generated using 100,000 to 300,000 records, and with one to three concurrent users. The time consumed for each of the fragmented and un-fragmented data is shown in Table 6.1 in milliseconds (ms).

Disease/City	Alor Star	Baling	Changlun	Dong Dang	Dusun Kapas	Gua Tinggi	Jitra	Langkawi	Merbok	Tawar	Total
influenza	856	874	864	876	861	870	825	807	810	803	8410
heart disease	795	793	840	850	860	815	840	840	795	845	8273
headache	765	840	823	813	825	801	815	833	872	860	8345
hypertension	801	802	804	826	825	853	803	831	810	805	8403
disease1	1162	1053	1246	1124	1102	1162	1170	1145	1195	1126	11455
disease2	820	752	752	700	771	767	830	720	813	729	7753
disease3	736	751	790	805	792	812	755	782	775	747	7774
disease4	755	675	743	845	745	727	731	796	707	771	7642
disease5	731	764	705	852	751	813	775	751	811	724	7713
disease6	783	764	769	775	761	759	775	711	804	789	7693
Total	8327	8030	8429	8633	8263	8418	8400	8306	8475	8300	83581

Figure 6.16: A Statistical Report Generated from a Clinical DW Repository

As shown in Table 6.1, it is clear that the time required to generate the report shown in Figure 6.16 in the case of fragmented data (highlighted in bold) is less than the time needed for un-fragmented data.

Table 6.1: Time Deference between Fragmented and Un-Fragmented Data for Report Generation

No. of Records	No. of Users	Fragmented data (ms)	Un-fragmented data (ms)
100,000	1	188	297
100,000	2	192	207
100,000	3	255	364
200,000	1	359	515
200,000	2	369	520
200,000	3	369	643
300,000	1	516	688
300,000	2	531	719
300,000	3	625	815

Section A.5 of appendix A; shows the detailed steps of conducting this test with data from 100,000 to 1,000,000 records, however, in this section, only data from 100,000 to 300,000 records is shown in Table 6.1 as a sample of what is shown in Table A.6 of appendix A.

6.7.3 Compatibility Testing Sprint

The compatibility testing was done to test the compatibility of the prototype with different applications and Web servers, as well as with different browsers. Once the prototype is developed using Java, a set of J2EE (Java 2 Enterprise Edition) application and web servers are used to test whether the prototype is compatible or not. These servers are: GlassFish application server, Apache Tomcat Web server and JBoss application servers. The prototype was deployed and executed successfully on all of these servers. Furthermore, Microsoft Internet Explorer, Mozilla Firefox and Google Chrome web browsers were used to test the browser compatibility, and the results showed that it works exactly the same with the three browsers.

6.7.4 End To End Testing Sprint

End To End testing was used to validate the prototype starting from sending the request by the ETL administrator, passing through BPEL orchestration point and invoking the proper Web Service, and finally, finishing by executing the proper ETL functionality.

To conduct the End To End testing, GlassFish Tester was used to create a test case with an XML file as an input and then another XML file is auto generated as an output. The input file acts as an ETL administrator who needs to execute one of the four ETL functionalities. Then, the BPEL forwards the client request to the appropriate Web service according to the parameter included in “<etl:operationParameter> </etl:operationParameter>” tag of the input XML file. For the current test case, “1” means “Extract”, “2” means “Transform”, “3” means

“Classify” and “4” means “Load”. Figures 6.17 and 6.18 respectively are the screenshots of the input and output XML files used in a test case done with parameter “1”, i.e. the input XML file tag is: “<etl:operationParameter>1</etl:operationParameter>”.

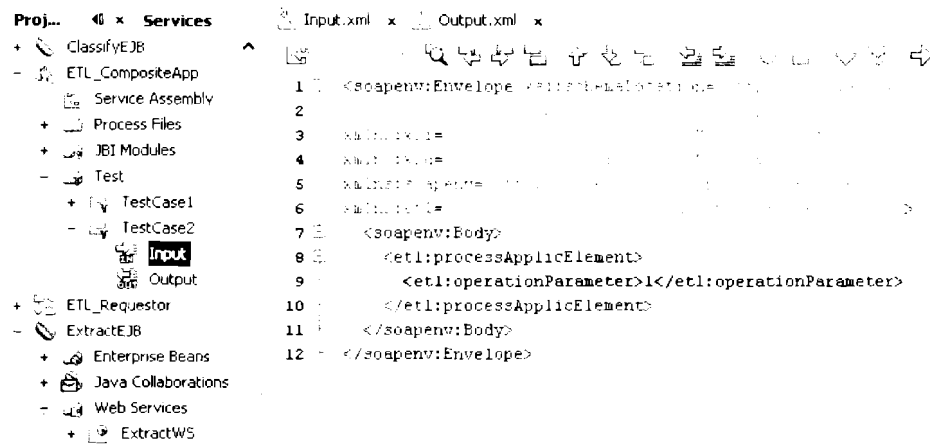


Figure 6.17: End to End Test Case Input File (input.xml)

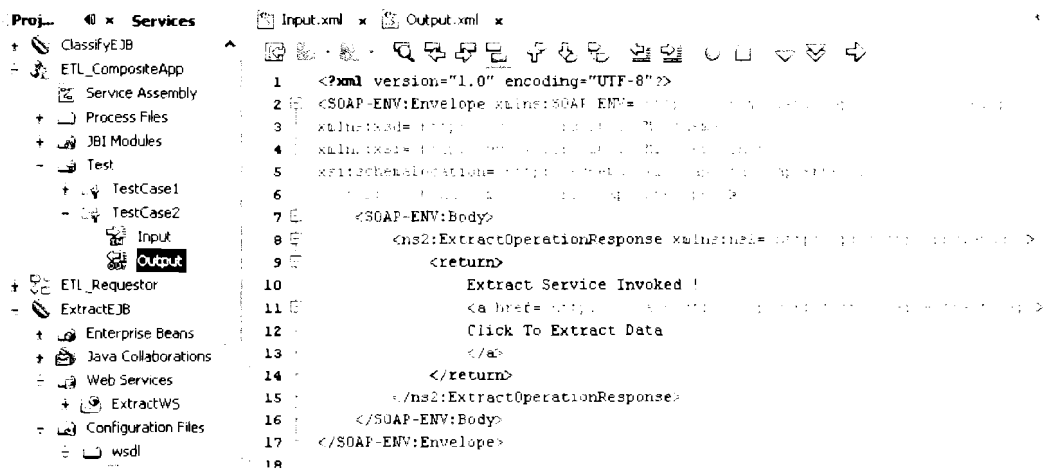


Figure 6.18: Auto generated End to End Test Case Output File (output.xml)

In addition to the “GlassFish Tester” testing, the effect of executing each of the ETL prototype components on the data available in the DW repository tables is verified. For example, the transformation component transforms the date of birth data available in P_BIRTHDAY field of Figure 6.19 to age data available in P_AGE field of Figure 6.20, and instead of having the date of birth of the patient; it calculates his age. Figure 6.19 shows the source data before executing the transformation component, while Figure 6.20 shows the transformed data after executing the transformation component.

DISEASE_ID	MEDICINE_ID	Gender_ID	P_BIRTHDAY	CITY_ID
4	8	1	1941-03-27	10
9	8	1	1977-10-04	6
5	7	1	1942-09-03	9
7	4	0	1943-08-15	6
8	2	1	1989-09-17	3
10	1	1	1989-09-17	3
10	4	1	1989-09-17	3
8	6	1	1950-06-04	9
4	6	1	1950-06-04	9
7	1	0	1945-11-01	9
7	10	0	1945-11-01	9
7	5	0	1945-11-01	9
10	3	1	1997-07-05	3
4	9	1	1997-07-05	3
4	10	1	1997-07-05	3
7	6	0	1959-05-11	7
10	5	0	1959-05-11	7
4	9	0	1959-05-11	7
7	9	1	2007-04-04	3
10	8	0	1996-10-25	3
7	7	0	1996-10-25	3
9	5	1	1975-01-06	7

Figure 6.19: Sample Data *before* Executing the Transformation Component

DISEASE_ID	MEDICINE_ID	Gender_ID	P_AGE	CITY_ID
4	8	1	70	10
9	8	1	34	6
5	7	1	69	9
7	4	0	68	6
8	2	1	22	3
10	1	1	22	3
10	4	1	22	3
8	6	1	61	9
4	6	1	61	9
7	1	0	66	9
7	10	0	66	9
7	5	0	66	9
10	3	1	14	3
4	9	1	14	3
4	10	1	14	3
7	6	0	52	7
10	5	0	52	7
4	9	0	52	7
7	9	1	4	3
10	8	0	15	3
7	7	0	15	3
9	5	1	37	7

Figure 6.20: Sample Data *after* Executing the Transformation Component

6.8 Conclusion

This chapter has explored the analysis, design, development and testing of the SOA-based ETL prototype, which was based on the mandatory specifications and recommendations of the theoretical framework for interoperable distributed ETL components. Furthermore, the Scrum methodology was adopted in the life cycle of developing this prototype. Chapter 7 discusses the evaluation, results, and discussions of the SOA-based ETL prototype.

CHAPTER SEVEN

EVALUATION

In addition to the prototype results and its testing that are explored in chapter 6 for the purpose of testing the prototype. This chapter explores three case studies that are conducted for the purpose of evaluating the prototype of this research that validates the theoretical framework.

For the purpose of the deployment of the ETL components using the SOA-based ETL prototype, GlassFish ESB IDE that is based on NetBeans IDE is used for this purpose and for the purpose of managing the web/application servers. In addition, the GlassFish ESB IDE is used for managing the database connections. A screenshot for this IDE is represented by Figure 7.1.

As shown in the top-left side of Figure 7.1, each component of the ETL (ExtractEJB, TransformEJB, and LoadEJB) is deployed in a separate EJB based distributed application. In addition, each of the BPEL-based orchestration point called (ETL_Requestor) and the composition functionality called (ETL_CompositeApp) is deployed in a separate web service. This decomposition of deploying the prototype components is done for the purpose of applying the distribution concept among the prototype components.

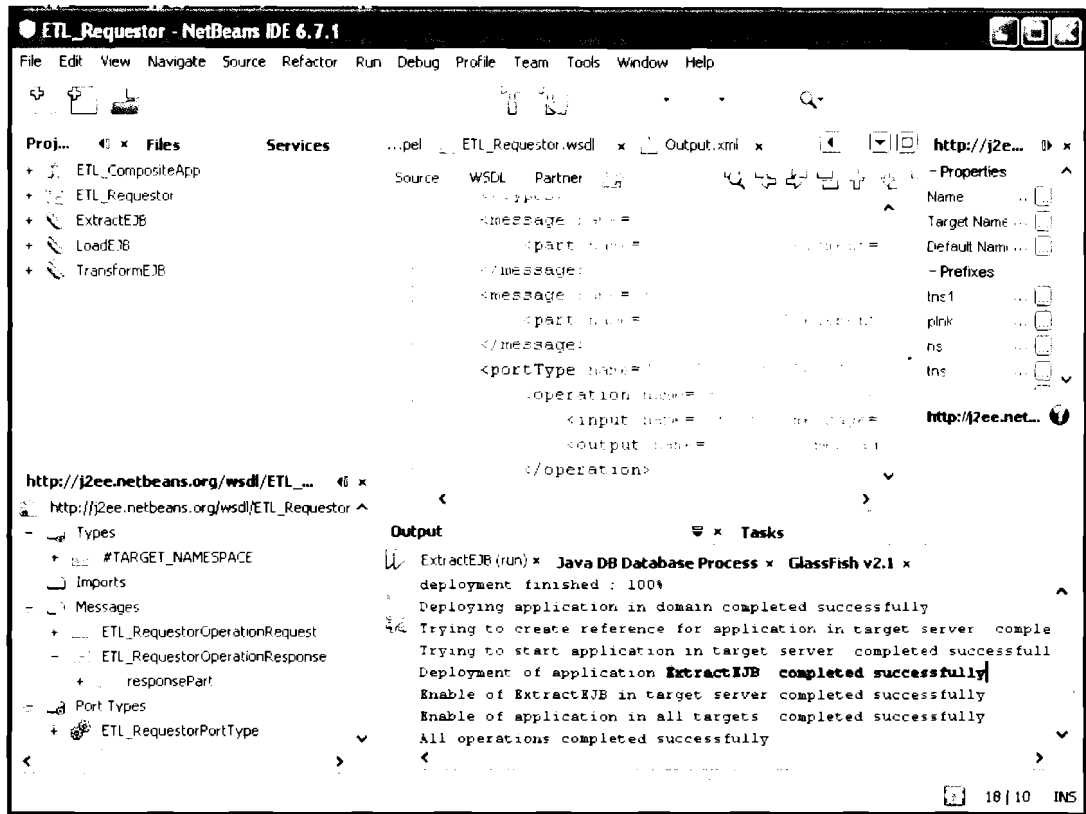


Figure 7.1: GlassFish ESB Based on NetBeans IDE for Managing the SOA-based ETL Prototype

Traditional ETL tools share similar steps that are performed to execute ETL functionalities. These steps are summarized by Figures 7.2, 7.3 and 7.4. As shown in these Figures, first, the data source needs to be identified and an SQL query editor is provided for the transformation process. After that, the destination is determined, and finally the data transfer is executed from the source to the destination.

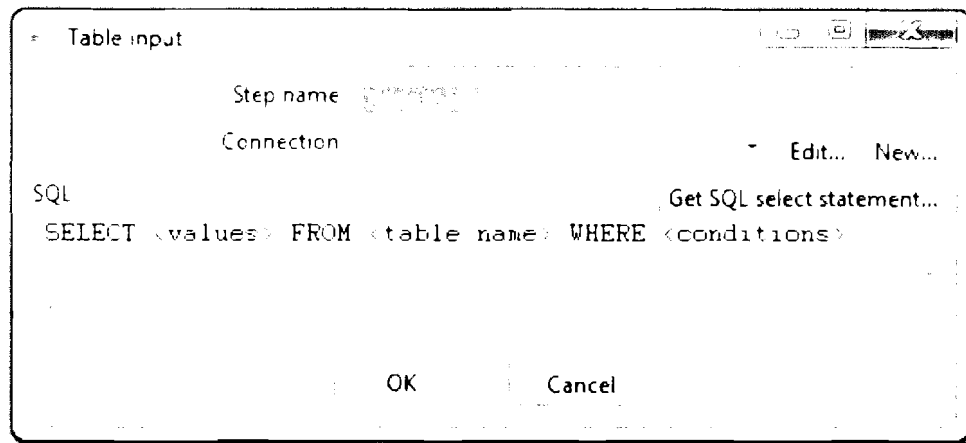


Figure 7.2: First Step in Executing Traditional ETL Tools

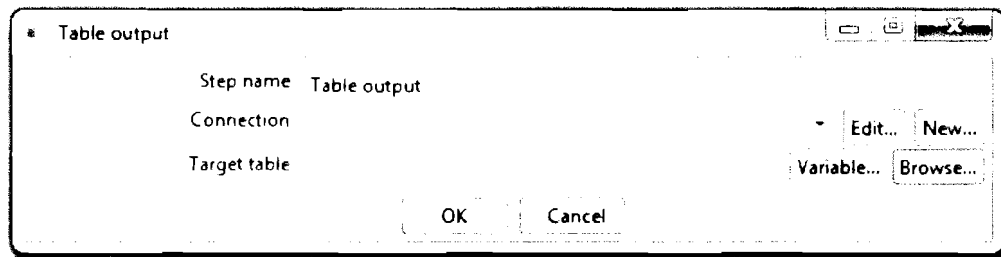


Figure 7.3: Second Step in Executing Traditional ETL Tools

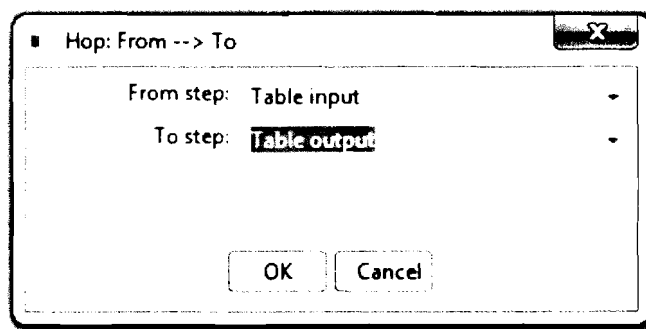


Figure 7.4: Third Step in Executing Traditional ETL Tools

7.1 Case Study 1: Applying ETL Functionalities on Palestine Electric Company (PEC) using Traditional and New ETL Tools

Palestine Electric Company (PEC) is the only company that operates power generation in Palestine as the operating company for the power plant. PEC, which was formed in Gaza under the laws of Palestine to develop, own and operate the first power plant in the Palestinian Territories. The power plant is a 140MW combined cycle power plant based on four Gas Turbines and two Steam Turbines in a configuration of two blocks. Only four turbines are operating since 2007, while the other two are broken down and not maintained till the time of conducting this case study. The turbines are designed to allow for dual fuel system and could burn both natural gas and fossil fuel. The fuel used by the power plant is stored in two big storage tanks with a capacity of 20 million liters, and the average daily consumption of fuel by the power plant is approximately 700,000 liters at full load operation of 140 MW.

7.1.1 ETL Business Needs

PEC does many statistical reports for the purpose of the decision making such as financial reports, the number of hours every turbine is operating, the amount of generated power in Megawatt (MW) over time, statistics of blackouts due to the Israeli occupation impediments and due to technical problems, and statistics for the fuel consumption by each turbine to make notifications regarding the available and the required amounts of fuel.

Doing these types of statistical reports requires extracting certain data fields from the source databases, sometimes making some transformations to the extracted data, and

loading data to a final data warehouse repository for the purpose of generating statistical reports.

7.1.2 Extracting Required Fields for Data Warehouse Star-Schema

This case study had applied the ETL functionalities on sample partial source DB schema of the PEC Company shown in Figure 7.5, and sample screenshots of auto generated data are explored in Figures 7.6 and 7.7.

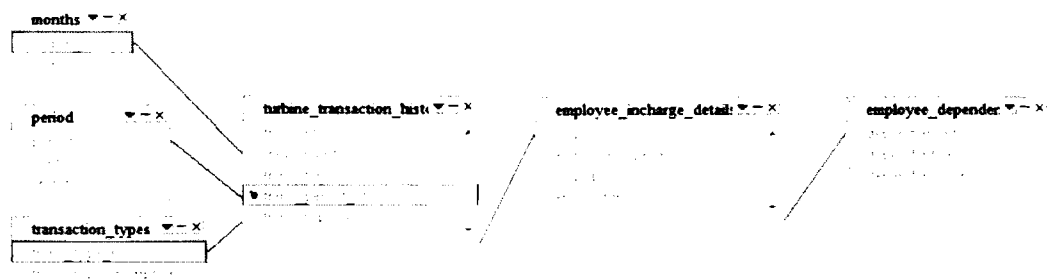


Figure 7.5: Sample Partial Source PEC DB Schema

trans_id	trans_month	trans_year	trans_period_id	trans_type_id	emp_incharge_id
1	2	2010	3	2	2
2	1	2010	1	2	1
3	4	2010	3	2	4
4	3	2010	4	2	3
5	2	2010	1	2	2
6	1	2010	3	2	1

Figure 7.6: Screenshot of Auto-generated Data in Turbine_transaction_history Table

trans_type_id	trans_type_description
1	Turbine Switching
2	Electricity Blackout
3	Maintenance Turn Off

Figure 7.7: Screenshot of Auto-generated Data in Transaction_types Table

The star schema of the data warehouse fields extracted from the PEC database is explored in Figure 7.8. The ETL functionalities are applied to the sample PEC table schema and to its auto generated data using both the traditional ETL tool that is used by PEC Company (MS SQL Server Integration Services (SSIS)) and the SOA-based ETL prototype. The last four fields of the fact table (elec_brkout) represent the measures of the fact table.

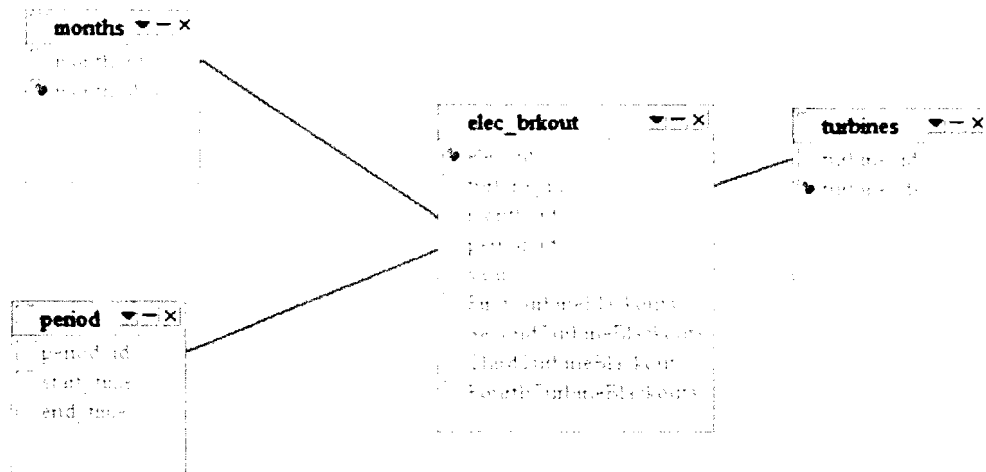


Figure 7.8: Star Schema of the Fact and Dimension Tables Extracted from PEC Database

7.1.3 Applying ETL Functionalities Using the Traditional ETL Tool

Using the traditional ETL tool simulated the current situation at PEC Company where the database is created and managed by Microsoft SQL Server DBMS that includes its own ETL tool which is SQL Server Integration Services (SSIS). This tool is reviewed and discussed in section 2.3 of this thesis. In PEC Company, the SSIS is installed on one server for the purpose of extracting certain fields from the source database and loading it in a data warehouse repository.

In this case study SSIS is used to extract the fields needed for the creation of the business process that enables the generation of a statistical report about blackouts to present the blackouts of the electricity for the year 2010. The fields are shown in the star schema of Figure 7.8. Although the generation of the statistical report is outside the scope of the ETL functionalities since it belongs to the presentation layer of the Business Intelligence, however, Table 7.1 and Figure 7.9 show summary and graph reports for the blackouts of the electricity for the year 2010. These two reports are shown as samples for reports that are generated over the data loaded to the final PEC DW repository using both traditional and SOA-based ETL tools. One server with Windows 2003 server operating system is used for running the ETL functionalities and the SSIS is installed as a desktop application on the server.

Table 7.1: Summary Report for 2010 Electricity Blackouts

Month/ Blackout	First Turbine Blackouts	Second Turbine Blackouts	Third Turbine Blackouts	Fourth Turbine Blackouts	Total
Jan	2	3	3	1	9
Feb	2	0	0	4	6
Mar	0	0	0	2	2
Apr	3	1	0	5	9
May	3	2	1	4	10
Jun	0	4	1	1	6
Jul	2	1	2	0	5
Aug	2	2	3	0	7
Sep	1	1	3	3	8
Oct	4	4	2	2	12
Nov	1	1	3	1	6
Dec	5	0	0	2	7
Total	25	19	18	25	87

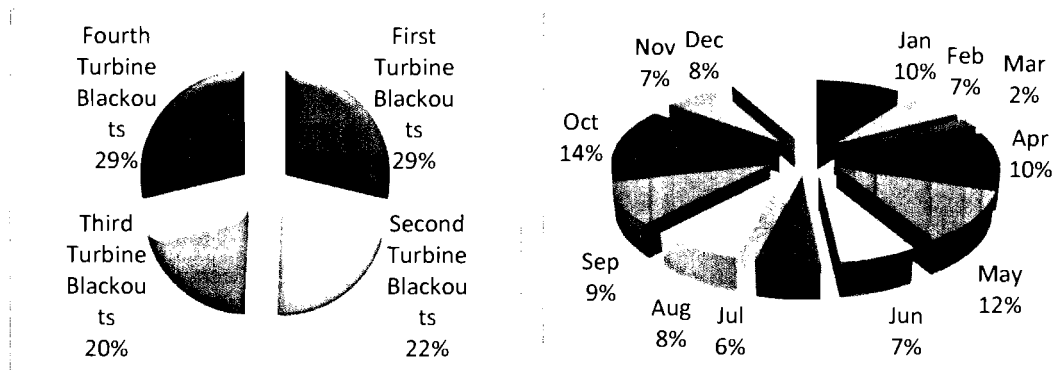


Figure 7.9: Graph Report for 2010 Electricity Blackouts

7.1.4 SOA-based ETL Prototype

After using the traditional ETL tool, this section describes using SOA-based ETL prototype. The SOA-based ETL prototype is deployed over four different servers two of them with Redhat Linux operating systems and the other two with Windows 2003 server operating systems. JBoss application servers as runtime environments are installed over the servers with Redhat Linux operating systems, and open source GlassFish application server is installed over the servers with Windows 2003 server operating systems. Each of the four application servers has its own distributed components container which is EJB container.

The Extraction component of the SOA-based ETL prototype is deployed separately on one of the JBoss application servers; the Transformation component of the SOA-based ETL prototype (as a language reformatting transformation) is deployed separately on the other JBoss application server; the Loading component of the SOA-based ETL prototype is deployed separately on one of the GlassFish application servers; the Orchestration point (BPEL Service) is deployed on the other

GlassFish application server; and the ETL administrator (Client) component is deployed separately on the same application server where the Orchestration point is deployed on. After that, the execution of the ETL functionalities is done using browsers of normal client PCs. Therefore, SOA-based ETL prototype is used here to extract, transform and load the fields needed for the creation of the business process that enables the generation of a statistical report about blackouts to present the blackouts of the electricity for the year 2010.

7.1.5 Goals Achieved

Based on the new ETL framework specifications explored in chapter 5, there are two goals that should be achieved. The first goal is related to the main functionality of ETL tools in general while the second goal is related to the usage of the distribution and interoperability of the ETL framework components. Both goals are listed in this section, and the results of the possibility of achieving each goal using both traditional ETL tool and the new ETL framework of this case study are listed. The results of achieving both goals were obtained through observation.

- a. Goal 1:* Extracting, transforming and loading the data in its final format to the data warehouse repository to be suitable for the presentation layer of the business intelligence activities.

This goal was achieved in both the traditional ETL tool (SSIS) and the SOA-based ETL prototype, and the extracted data, the transformations and the loaded data that is necessary for generating a statistical report about electricity blackouts to present the blackouts of the electricity for the year 2010 is

obtained exactly similar in both using the traditional and SOA-based ETL tools. Figures 7.10, 7.11, 7.12 and 7.13 show the data of the fact and dimension tables of the star schema while Figure 7.14 shows the effect of the translation component on the data since it changes the time units from English to Arabic.

elec_id	turbine_id	month_id	period_id	
1	1	1	1	5
2	1	1	2	5
3	3	3	2	5
4	2	2	2	1
5	2	2	3	1
6	4	4	3	2
7	1	1	4	2
8	4	4	5	4
9	4	4	6	3
10	3	3	7	1
11	3	3	8	1
12	1	1	9	1
13	3	3	9	5
14	4	4	10	3
15	2	2	10	3
16	2	2	11	2

Figure 7.10: A Sample of the Auto Generated Data of the Fact Table (*elec_brkout*) of the Star Schema Explored in Figure 7.8

month_id	month_desc
1	Jan
2	Feb
3	Mar
4	Apr
5	May
6	Jun
7	Jul
8	Aug
9	Sep
10	Oct
11	Nov
12	Dec

Figure 7.11: The Data of the Dimension Table (*months*) of the Star Schema Explored in Figure 7.8

period_id	start_time	end_time
1	03:00 PM	06:00 PM
2	06:00 PM	09:00 PM
3	01:00 AM	04:00 AM
4	07:00 PM	10:00 AM
5	08:00 PM	11:00 PM

Figure 7.12: The Data of the Dimension Table (periods) of the Star Schema Explored in Figure 7.8

turbine_id	turbine_desc
1	FirstTurbine
2	SecondTurbine
3	ThirdTurbine
4	FourthTurbine

Figure 7.13: The Data of the Dimension Table (turbines) of the Star Schema Explored in Figure 7.8

period_id	start_time	end_time
1	03:00 مساءً	06:00 مساءً
2	06:00 صباحاً	09:00 مساءً
3	01:00 صباحاً	04:00 صباحاً
4	07:00 مساءً	10:00 صباحاً
5	08:00 مساءً	11:00 مساءً

Figure 7.14: The Data of the Dimension Table (periods) of the Star Schema Explored in Figure 7.8 (after executing the translation component)

b. Goal 2: Achieving Distribution and Interoperability among ETL components

By comparing both ways of using the traditional ETL tool (SSIS) and the SOA-based ETL prototype, it is observed that goal #2 is achieved in using SOA-based

ETL prototype, while using the traditional ETL tool does not achieve most of the goal parts. By using the traditional ETL tool, the SSIS tool is a tightly coupled tool and its components cannot be distributed. Furthermore, the whole tool is installed on one server because of the impossibility of distributing its components. Therefore, Components' Distribution and Interoperability among ETL components could not be achieved when the traditional ETL tool is used.

On the other hand, by following the SOA-based ETL prototype, each component of the SOA-based ETL prototype is distributed and deployed over a separated EJB container on a different application server, and these components were interoperable to each others because all of the components interoperated to achieve the Extraction, Transformation and Loading functionalities. Therefore, Components' Distribution and Interoperability is achieved among the ETL components. Table 7.2 shows a checklist that summarizes the observation results of conducting this case study.

Table 7.2: Observation Results Checklist

	Using Traditional ETL Tool	Using SOA-based ETL Prototype
Extraction Performed	Yes	Yes
Transformation Performed	Yes	Yes
Loading Performed	Yes	Yes
Distribution Achieved	No	Yes
Interoperability Achieved	No	Yes

Achieving the Distribution and Interoperability has enabled the achievement of Flexibility to extend the ETL tool. The SSIS tool is a tightly coupled tool and its components cannot be extended easily. On the other hand, a component for translating some symbols to Arabic language during the transformation process of ETL processes is encapsulated in a separated web service and plugged to the SOA-based ETL prototype as language reformatting transformation component. Furthermore, the reusability is achieved by using the SOA-based ETL prototype. A component for translating symbols to Arabic language during the transformation process of ETL processes is reused by encapsulating it in a separated web service and reusing it within the SOA-based ETL prototype. On the other hand, the SSIS tool is a tightly coupled tool and its components are not reusable.

7.2 Case Study 2: Applying ETL Functionalities on Limkokwing University of Creative Technology (LUCT) using Traditional and New ETL Tools

Limkokwing University of Creative Technology (LUCT) is private Malaysian university that has six faculties, which are: Faculty of Architecture & the Built Environment; Faculty of Business Management & Globalization; Faculty of Communication, Media & Broadcasting; Faculty of Design Innovation; Faculty of Information & Communication Technology; and Faculty of Multimedia Creativity.

7.2.1 ETL Business Needs

LUCT does many statistical reports for the purpose of the decision making such as students' performance reports, staff performance reports, financial reports, and

foreign students' visa tracing reports. Doing these types of statistical reports requires extracting certain data fields from the LUCT main database, sometimes making some transformations to the extracted data, and loading data to a final data warehouse repository for the purpose of generating statistical reports.

7.2.2 Extracting Required Fields for Data Warehouse Star-Schema

This case study had applied the ETL functionalities on sample partial source DB schema of LUCT University shown in Figure 7.15. The star schema of the data warehouse fields extracted from the LUCT database is explored in Figure 7.16. The last field of the fact table (student_performance) of Figure 7.16 represents a measure of the fact table. The ETL functionalities are applied to the sample LUCT table schema and to its auto generated data using both the traditional ETL tool that is used by LUCT Company (MS SQL Server Integration Services (SSIS)) and the SOA-based ETL prototype.

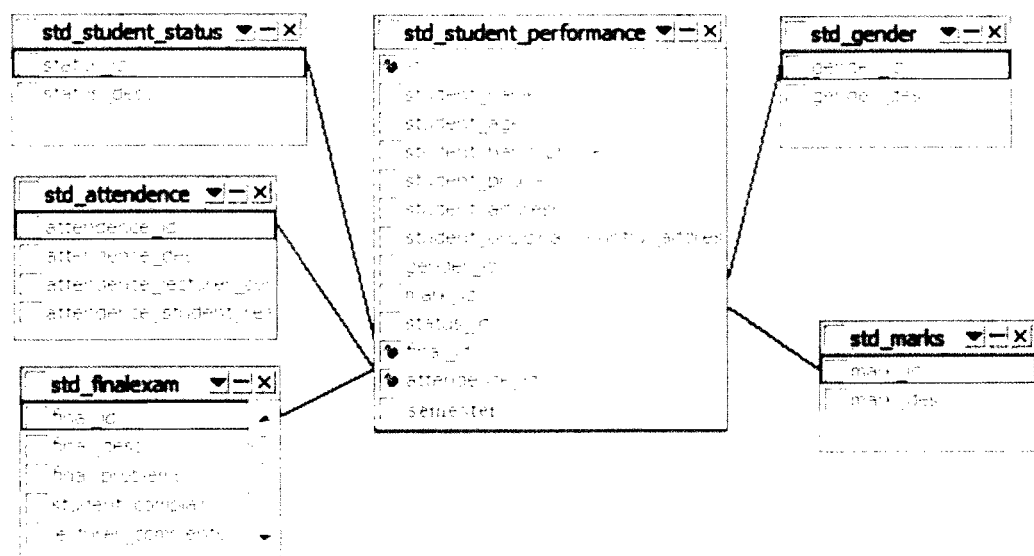


Figure 7.15: Sample Partial Source DB Schema of LUCT

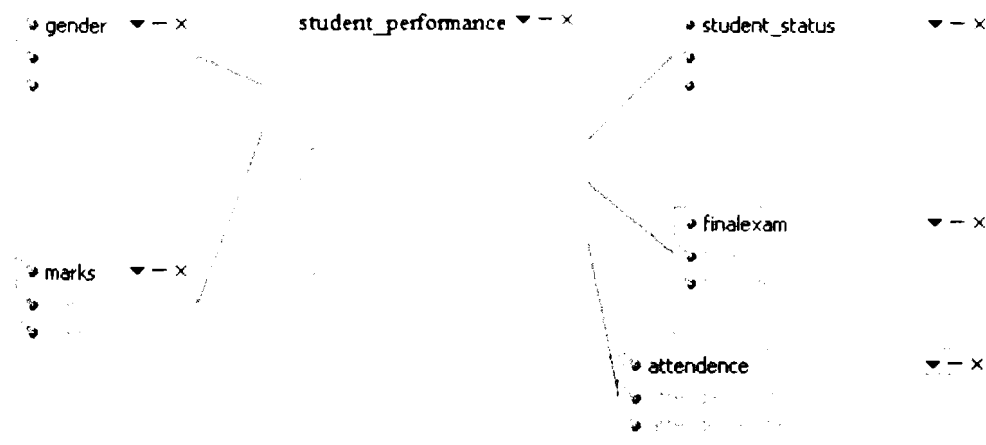


Figure 7.16: Star Schema of the Fact and Dimension Tables Extracted from LUCT Database

7.2.3 Applying ETL Functionalities Using the Traditional ETL Tool

Using the traditional ETL tool simulated the current situation at LUCT where the database is created and managed by Microsoft SQL Server DBMS that includes its own ETL tool which is SQL Server Integration Services. This tool is reviewed and discussed in section 2.3 of this thesis. In LUCT, the SSIS is installed on one server for the purpose of extracting certain fields from the source database and loading it in a data warehouse repository.

In this case study SSIS is used to extract the fields needed for the creation of the business process that enables the generation of a statistical report about the students performance in Java II subject for February-June, 2011 semester. The fields are shown in the star schema of Figure 7.16. Although, the generation of the statistical report is outside the scope of the ETL functionalities since it belongs to the

presentation layer of the Business Intelligence, however, Table 7.3 and Figure 7.17 show summary and graph reports for the students' performance in Java II for Feb-June, 2011 semester. These two reports are shown as samples for reports that are generated over the data loaded to the final LUCT DW repository using both traditional and SOA-based ETL tools. One server with Windows 2003 server operating system is used for running the ETL functionalities and the SSIS is installed as a desktop application on the server.

Table 7.3: Summary Report for Students' Performance in Java II for Feb-June, 2011 Semester

Grade	Student Nos.
A	23
A-	14
B+	15
B	16
B-	30
C+	28
C	14
C-	12
D	12
F	5
Total	169

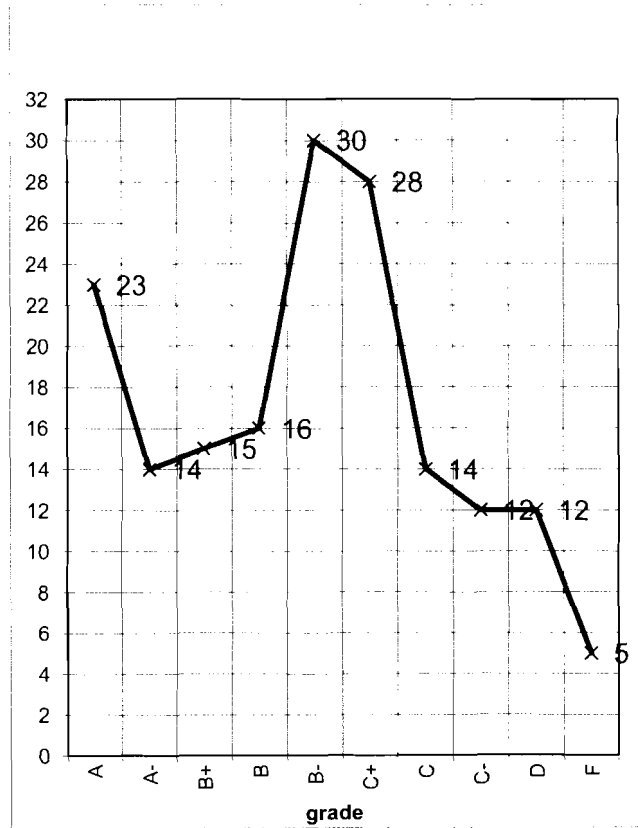


Figure 7.17: Summary Report for Students' Performance in Java II for Feb-June, 2011 Semester

7.2.4 SOA-based ETL Prototype

After using the traditional ETL tool, this section describes using SOA-based ETL prototype. The SOA-based ETL prototype is deployed over four different servers two of them with Redhat Linux operating systems and the other two with Windows 2003 server operating systems. Apache Tomcat Jakarta Web servers as runtime environments are installed over the servers with Redhat Linux operating systems, and Open source GlassFish application server is installed over the servers with

Windows 2003 server operating systems. Each of the four application servers has its own distributed components container which is EJB container.

The Extraction component of the SOA-based ETL prototype is deployed separately on one of the Apache Tomcat Jakarta Web servers; the Transformation component of the SOA-based ETL prototype is deployed separately on the other Apache Tomcat Jakarta Web server; the Loading component of the SOA-based ETL prototype is deployed separately on one of the GlassFish application servers; the Orchestration point (BPEL Service) is deployed on the other GlassFish application server; and the ETL administrator (Client) component is deployed separately on the same application server where the Orchestration point deployed on. After that, the execution of the ETL functionalities is done using browsers of normal client PCs. Therefore, SOA-based ETL prototype is used here to extract the fields needed for the creation of the business process that enables the generation of a statistical report about the students performance in Java II for February-June, 2011 semester.

7.2.5 Goals Achieved

Based on the new ETL framework specifications explored in chapter 5, there are two goals that should be achieved. The first goal is related to the main functionality of ETL tools in general while the second goal is related to the usage of the distribution and interoperability of the ETL framework components. Both goals are listed in this section, and the results of the possibility of achieving each goal using both traditional ETL tool and the new ETL framework of this case study are listed. The results of achieving both goals were obtained through observation.

- a. *Goal 1:* Extracting, transforming and loading the data in its final format to the data warehouse repository to be suitable for the presentation layer of the business intelligence activities.

This goal was achieved in both the traditional ETL tool (SSIS) and the SOA-based ETL prototype. The extracted and the loaded data that is necessary for generating a statistical report about the students' performance in Java II for February-June, 2011 semester is obtained exactly similar in both using the traditional and SOA-based ETL tools. Transformation in this case study is just a step that does nothing, because no transformations required in LUCT. Therefore, it is redundant in the traditional ETL tool, and skipped in the SOA-based ETL prototype. Figures 7.18, 7.19, 7.20, 7.21, 7.22 and 7.23 show the data of the fact and dimension tables of the star schema.

id	gender_id	mark_id	status_id	final_id	attendance_id
1	1	5	1	1	1
2	1	4	1	1	1
3	1	2	1	1	1
4	2	2	1	1	1
5	2	3	1	1	1
6	1	3	2	1	1
7	2	4	1	1	1
8	2	1	1	1	1
9	2	2	1	1	1
10	2	8	1	1	1
11	1	8	1	1	1
12	1	10	1	2	2
13	2	1	1	1	1
14	1	5	1	1	1
15	2	6	1	1	1
16	1	7	1	1	1
17	1	7	1	1	1
18	2	1	1	1	1

Figure 7.18: A Sample of the Auto Generated Data of the Fact Table (student_performance) of the Star Schema Explored in Figure 7.16

mark_id	mark_desc
1	A
2	A-
3	B+
4	B
5	B-
6	C+
7	C
8	C-
9	D
10	F

Figure 7.19: The Data of the Dimension Table (marks) of the Star Schema Explored in Figure 7.16

attendance_id	attendance_desc
1	more than 80 %
2	less than 80%

Figure 7.20: The Data of the Dimension Table (attendance) of the Star Schema Explored in Figure 7.16

status_id	status_desc
1	old_scheme_student
2	new_scheme_student

Figure 7.21: The Data of the Dimension Table (student_status) of the Star Schema Explored in Figure 7.16

gender_id	gender_desc
1	male
2	female

Figure 7.22: The Data of the Dimension Table (gender) of the Star Schema Explored in Figure 7.16

final_id	final_desc
1	attended
2	did not attend

Figure 7.23: The Data of the Dimension Table (finalexam) of the Star Schema Explored in Figure 7.16

b. *Goal 2: Achieving Distribution and Interoperability among ETL components*

By comparing the usage of the traditional ETL tool (SSIS) versus the SOA-based ETL prototype, it is observed that goal #2 is achieved in using SOA-based ETL prototype, while using the traditional ETL tool does not achieve most of the goal parts. By using the traditional ETL tool, the SSIS tool is a tightly coupled tool and its components cannot be distributed. Furthermore, the whole tool is installed on one server because of the impossibility of distributing its components. Therefore, Components' Distribution and Interoperability among ETL components could not be achieved when the traditional ETL tool is used.

On the other hand, by following the SOA-based ETL prototype, each component of the SOA-based ETL prototype is distributed and deployed over a separated EJB container on a different application server, and these components were interoperable to each others because all of the components interoperated to achieve the Extraction, Transformation and Loading functionalities. Therefore, Components' Distribution and Interoperability is achieved among the ETL components. Table 7.4 shows a checklist that summarizes the observation results of conducting this case study.

Table 7.4: Observation Results Checklist

	Using Traditional ETL Tool	Using SOA-based ETL Prototype
Extraction Performed	Yes	Yes
Transformation Performed	No	No
Loading Performed	Yes	Yes
Distribution Achieved	No	Yes
Interoperability Achieved	No	Yes

7.3 Case Study 3: Applying ETL Functionalities on Professionals Information Technology (PIT) Company using Traditional and New ETL Tools

The Professionals Information Technology Co. is a premier technology consultancy and training center in Gaza Strip, Palestine. PIT has been in business since 1997 and has grown over 100% each year, with 7 branches in the year 2011. Furthermore, PIT put much emphasis in hands on training so that its trainees can join the IT workforce with expertise and confidence. In addition, PIT consultants provide specialized services to its clients ranging from small business system design and integration to large government contract management.

PIT provides technical and non-technical training services for the public and private sector and its instructors are not only certified but also experienced in the subject matters they teach. Most make their daily living working in the IT field. Currently PIT has 40 trainers; some are employed by PIT Co, while others are PIT partners.

7.3.1 ETL Business Needs

PIT does statistical reports for the purpose of the decision making such as trainees' performance reports and trainers' performance reports. Doing these types of statistical reports requires extracting certain data fields from the PIT main database, sometimes making some transformations to the extracted data, and loading data to a final data warehouse repository for the purpose of generating statistical reports.

7.3.2 Extracting Required Fields for Data Warehouse Star-Schema

This case study had applied the ETL functionalities on sample partial source DB schema of PIT Company shown in Figure 7.24. The star schema of the data warehouse fields extracted from the PIT database is explored in Figure 7.25. The last field of the fact table (trainer_evaluation) of Figure 7.25 represents a measure of the fact table. The ETL functionalities are applied to the sample PIT table schema and to its auto generated data using both the traditional ETL tool that is used by PIT Company (Oracle Warehouse Builder (OWB)) and the SOA-based ETL prototype.

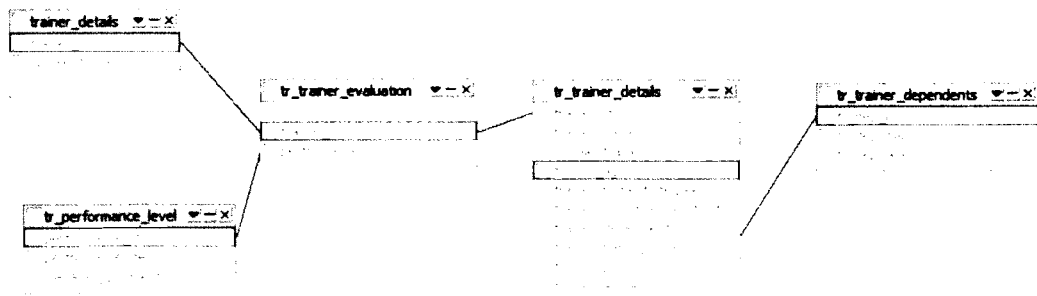


Figure 7.24: Sample Partial Source DB schema of PIT

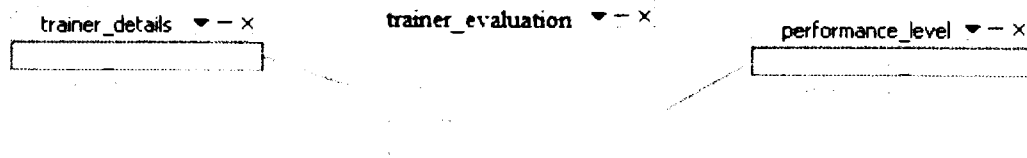


Figure 7.25: Star Schema of the Fact and Dimension Tables Extracted from PIT Database

7.3.3 Applying ETL Functionalities Using the Traditional ETL Tool

Using the traditional ETL tool simulated the current situation at PIT where the database is created and managed by Oracle DBMS that includes its own ETL tool which is Oracle Warehouse Builder. This tool is reviewed and discussed in section 2.3 of this thesis. In PIT, the OWB is installed on one server for the purpose of extracting certain fields from the source database and loading it in a data warehouse repository.

In this case study OWB is used to extract the fields needed for the creation of the business process that enables the generation of a statistical report about the trainers' performance in training advanced IT courses. The fields are shown in the star schema of Figure 7.25. Although, the generation of the statistical report is outside the scope of the ETL functionalities since it belongs to the presentation layer of the Business Intelligence, however, Table 7.5 and Figure 7.26 show summary and graph reports for the trainers' performance in training IT courses. These two reports are shown as samples for reports that are generated over the data loaded to the final PIT

DW repository using both traditional and SOA-based ETL tools. One server with Windows 2003 server operating system is used for running the ETL functionalities and the OWB is installed as a desktop application on the server.

Table 7.5: Summary Report for Trainers' Performance in Training IT Courses.

Performance Level	No. of Trainees
Excellent	31
Very Good	50
Good	12
Acceptable	5
Not Acceptable	10
Total	108

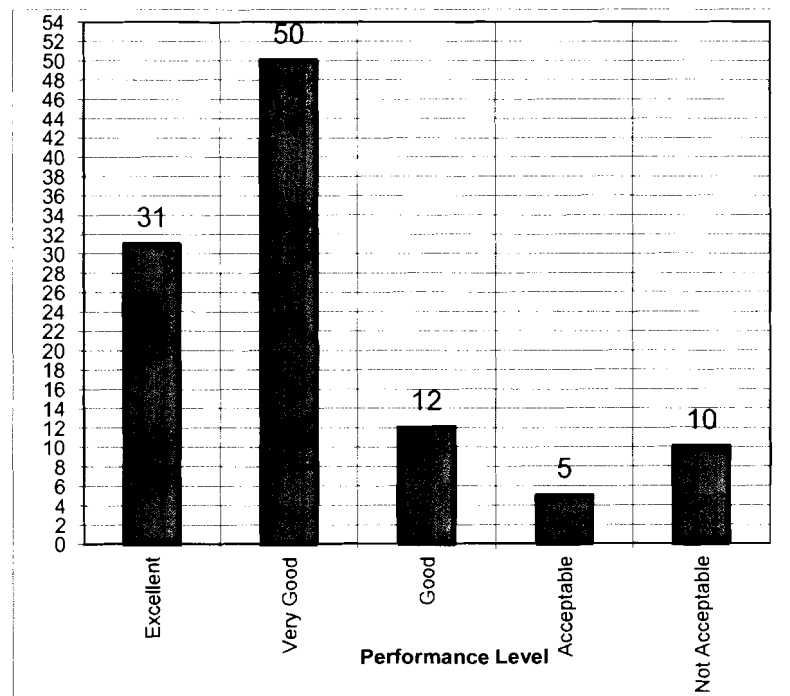


Figure 7.26: Graph Report for Trainers' Performance in Training IT Courses.

7.3.4 SOA-based ETL Prototype

After using the traditional ETL tool, this section describes using SOA-based ETL prototype. The SOA-based ETL prototype is deployed over four different servers two of them with Redhat Linux operating systems and the other two with Windows 2003 server operating systems. Apache Tomcat Jakarta Web servers as runtime environments are installed over the servers with Redhat Linux operating systems, and Open source GlassFish application server is installed over the servers with Windows 2003 server operating systems. Each of the four application servers has its own distributed components container which is EJB container.

The Extraction component of the SOA-based ETL prototype is deployed separately on one of the Apache Tomcat Jakarta Web servers; the Transformation component of the SOA-based ETL prototype is deployed separately on the other Apache Tomcat Jakarta Web server; the Loading component of the SOA-based ETL prototype is deployed separately on one of the GlassFish application servers; the Orchestration point (BPEL Service) is deployed on the other GlassFish application server; and the ETL administrator (Client) component is deployed separately on the same application server where the Orchestration point deployed on. After that, the execution of the ETL functionalities is done using browsers of normal client PCs. Therefore, SOA-based ETL prototype is used here to extract the fields needed for the creation of the business process that enables the generation of a statistical report about the trainers' performance in training advanced IT courses.

7.3.5 Goals Achieved

Based on the new ETL framework specifications explored in chapter 5, there are two goals that should be achieved. The first goal is related to the main functionality of ETL tools in general while the second goal is related to the usage of the distribution and interoperability of the ETL framework components. Both goals are listed in this section, and the results of the possibility of achieving each goal using both traditional ETL tool and the new ETL framework of this case study are listed. The results of achieving both goals were obtained through observation.

- a. *Goal 1:* Extracting, transforming and loading the data in its final format to the data warehouse repository to be suitable for the presentation layer of the business intelligence activities.

This goal was achieved in both the traditional ETL tool (OWB) and the SOA-based ETL prototype. The extracted and loaded data that is necessary for generating a statistical report about the trainers performance in training advanced IT courses; is obtained exactly similar in both using the traditional and SOA-based ETL tools. . Transformation in this case study is just a step that does nothing, because no transformations required in PIT Company. Therefore, it is redundant in the traditional ETL tool, and skipped in the SOA-based ETL prototype. Figures 7.27, 7.28, and 7.29 show the data of the fact and dimension tables of the star schema.

id	trainer_id	performance_id	
1	2009008	1	
2	2009020	2	
3	2009014	2	
4	2009008	1	
5	2009020	1	
6	2009021	1	
7	2009011	1	
8	2009008	2	
9	2009021	1	
10	2009014	3	
11	2009009	1	
12	2009011	1	
13	2009031	1	
14	2009008	2	
15	2009009	2	
16	2009031	3	
17	2009014	1	
18	2009008	1	
19	2009031	1	
20	2009009	1	

Figure 7.27: A Sample of the Auto Generated Data of the Fact Table (trainer_evaluation) of the Star Schema Explored in Figure 7.25

trainer_id	course_trained
2009008	Microsoft Certified System Engineer (MCSE)
2009009	Microsoft Certified System Administrator (MCSA)
2009011	Oracle Certified Administrator (OCA)
2009014	Sun Certified Java Programmer (SCJP)
2009020	International Computer Driving License (ICDL)
2009021	Adobe Photoshop
2009031	Microsoft Certified System Developer (MCSD)

Figure 7.28: The Data of the Dimension Table (trainer_details) of the Star Schema Explored in Figure 7.25

performance_id	performance_desc
1	Excellent
2	Very Good
3	Good
4	Acceptable
5	Not Acceptable

Figure 7.29: The Data of the Dimension Table (performance_level) of the Star Schema Explored in Figure 7.25

b. *Goal 2: Achieving Distribution and Interoperability among ETL components*

By comparing the usage of the traditional ETL tool (OWB) versus the SOA-based ETL prototype, it is observed that goal #2 is achieved in using SOA-based ETL prototype, while using the traditional ETL tool does not achieve most of the goal parts. By using the traditional ETL tool, the OWB tool is a tightly coupled tool and its components cannot be distributed. Furthermore, the whole tool is installed on one server because of the impossibility of distributing its components. Therefore, Components' Distribution and Interoperability among ETL components could not be achieved when the traditional ETL tool is used.

On the other hand, by following the SOA-based ETL prototype, each component of the SOA-based ETL prototype is distributed and deployed over a separated EJB container on a different application server, and these components were interoperable to each others because all of the components interoperated to achieve the Extraction, Transformation and Loading functionalities. Therefore, Components' Distribution and Interoperability is achieved among the ETL components. Table 7.6 shows a checklist that summarizes the observation results of conducting this case study.

Table 7.6: Observation Results Checklist

	Using Traditional ETL Tool	Using SOA-based ETL Prototype
Extraction Performed	Yes	Yes
Transformation Performed	No	No
Loading Performed	Yes	Yes
Distribution Achieved	No	Yes
Interoperability Achieved	No	Yes

7.4 Conclusion

This chapter has evaluated the SOA-based ETL prototype by conducting three case studies for the purpose of evaluating the prototype of this research that validates the theoretical framework. The three case studies are conducted with three different organizations that use the ETL tools. The observed results of conducting the case studies have proved the distribution and interoperability among the new ETL framework components.

CHAPTER EIGHT

CONCLUSION AND FUTURE WORK

This chapter concludes the thesis by discussing the research work carried out in supporting the thesis proposition. It highlights the problems of the current ETL framework, defines the new ETL theoretical framework, and discusses the development and evaluation of the SOA-based ETL prototype that validates the framework. In addition, the contribution of this research is concluded, and the chapter ends with proposing directions for future work in the areas of SOA-based ETL and business intelligence.

8.1 Conclusion

The goal of this research was to define and validate a framework for distributable and interoperable ETL components. The research has succeeded in filling the existing gaps of the current ETL framework regarding distribution and interoperability of ETL components. The research discussed how to distribute the Extraction, Transformation and Loading components so as to achieve distribution and interoperability of these ETL components. In addition, it explored how the ETL framework can be extended easier as a proof of the extensibility concept. Service Oriented Architecture was adopted to address the mentioned missing features of distribution and interoperability by restructuring the current ETL framework. After defining the new ETL framework as a solution to the research problems, the framework was validated by developing a prototype based on the new framework design and specifications. Therefore, this research has succeeded in defining a

validated framework for interoperable and distributed Extraction-Transformation-Loading (ETL) based on service oriented architecture.

8.1.1 Problems of Current ETL Framework

An intensive literature review about ETL and related technologies was conducted at the beginning of this research journey. The literature review showed that there are gaps in the ETL area, which are lack of components distribution and interoperability, and these gaps contributed to problems of the current ETL framework.

In the current ETL framework, data warehouse gets its data from many sources available in different “geographically isolated” locations. Each data source at times needs a complete ETL tool to perform the Extraction process. If the Extraction is done at data source locations, an ETL tool including all of its features as tightly-coupled software needs to be installed in each data source location. Here, the Transformation and Loading features are redundant as only the Extraction process is required at a data source location. In addition, the two redundant features are only required at the location of the data warehouse destination.

Therefore, in a data warehouse project each data source requires an ETL license to extract data from the sources. This increases the number of licenses whenever a new data source is included into the data warehouse project. Thus, the cost of the data warehouse project increases. However, if the ETL components are distributed and hosted on application server(s), any data source administrator can remotely use the same Extraction component to push the data to the Transformation phase of ETL. This signals a cheaper option when new data sources are used.

Managing many ETL tools to extract data from many different sources needs extra administration and maintenance effort, because the administrators are often different persons from one location to another. On the contrary, when the ETL components are distributed into interoperable components and hosted on application server(s) as web services, the administration of these components is centralized amongst all ETL administrators of the project.

Furthermore, at times, due to the complexity, long learning curve of the available ETL tools, and difficulty to achieve some extensibility in terms of additional functionalities, many organizations prefer to adopt the in-house development to perform ETL tasks, and this increases the project time and effort.

8.1.2 Proposing the New ETL Framework

The research problems motivated for more research to bridge the gaps that have been left in the ETL field regarding distribution and interoperability. Literature about distributed systems, web services, SOA and software architecture was essential in determining the methods and concepts that contributed in bridging these gaps.

The solution of the research problems is a new ETL framework that is based on Service Oriented Architecture. SOA was selected as the main architecture for proposing the new ETL framework due to its central role in resolving components distribution and interoperability issues. SOA was supported by literatures about distributed systems, web services and software architectures for the new ETL framework proposition.

8.1.3 Defining the New ETL Framework

The proposed ETL framework was refined and enhanced to constitute a base in answering the research questions and achieving the research objectives. The framework was defined as “*Framework for Interoperable and Distributed Extract-Transform-Load (ETL) Based on Service Oriented Architecture*”. The new ETL framework is defined in a meta-model using both UML for diagrammatic description and EBNF for textual definition. The meta-model and the specifications have consolidated that interoperability and distribution of the ETL components. In particular, ETL vendors or developers who are interested to adopt the new ETL framework as a base to produce their own ETL tools will have to consider a set of specifications regarding: Distribution, Web services, SOA orchestration processes, XML schemas, WSDL documents, SOA-based composition, and the framework extensibility. The new framework has advantages over the traditional ETL framework, which are interoperability, flexibility, reusability, scalability and cost efficiency.

8.1.4 Validating the New ETL Framework

The research methodology used in this research adopts both DSA and Scrum methodologies. DSA is adopted as the main and comprehensive methodology of the research, while Scrum is adopted only for the prototype that was developed as a proof of the framework concepts. The thesis included the analysis, design, development, testing, and evaluation of the SOA-based ETL prototype. Analyzing the prototype requirements is based on the theoretical framework design and specifications. Scrum methodology was adopted in the life cycle of developing this

prototype due to its strength as a development methodology, and the same methodology is used to manage the testing tasks.

Four types of testing are done for the purpose of verifying that the prototype meets the framework design and specifications. A web interface combined all the components of the prototype in one composite application as an interface for the user to do testing and execution of any ETL web service. The types of testing carried out for the purpose of validating the prototype are unit testing, Web services testing, compatibility testing, classified-fragmentation speed and scalability testing, and end to end testing.

Moreover, three case studies were conducted for the purpose of evaluating the prototype of this research that validates the new ETL framework in real data warehouse environments. The case studies were conducted using the data of three different organizations that adopt data warehouse solutions for the purpose of generating statistical reports. The observed results of conducting the case studies have proved that the new framework enables the distribution and interoperability among the ETL components.

8.2 Research Contributions

This research contributes to the field of ETL by adding the distribution and interoperability concepts to the ETL framework, and restructuring this framework based on SOA. Therefore, the new ETL framework provided the ability of using each ETL component separately by considering the loose coupling feature of the new ETL framework.

Furthermore, the new ETL framework adds contributions towards the area of data warehousing and business intelligence because ETL is a core concept in this area. This contribution has provided distributable and interoperable ETL framework that can be used in the design and implementation of data warehouse and business intelligence projects. Therefore, data warehouse and business intelligence projects can now take advantages of this contribution by including any ETL component as a service in the main portal of a data warehouse or business intelligence project.

In addition, the research contributes to the ETL vendors (companies that develop ETL tools), since it enables them to develop new releases of ETL tools that incorporate the distribution and interoperability features based on the new ETL framework. This enables them to save time and effort in developing, extending and reusing the ETL components. The developers can follow the new ETL framework definition as a base for their future ETL tools development, thus reducing the licensing cost on the users' side.

The research contributes also to the ETL administrators (users) by simplifying the administration process of a DW project, because in this case, the administrators only need to know the steps of using the required component rather than knowing the steps of using the complete ETL tool. In addition, the research contributes to the ETL customers (companies that buy non-free ETL tools to implement DW projects); by reducing the number of licenses needed for implementing a DW project. They only buy the components they require or they can just share the same components of a portal with other customers, in case that the components are deployed on a remote application server.

8.3 Limitation

This research included the classified-fragmentation component as an extension to the new ETL framework, and did not include other components to suit other specific business needs. In addition, the types of data used for the SOA-based ETL prototype implementation are limited to text, image or/and video. Furthermore, the framework definition is limited to ETL components and did not include other business intelligence components. These limitations motivate for more research that could add more contributions to the body of knowledge.

8.4 Future Work

While conducting this research, possible directions for future works have been identified. These directions are discussed in sections 8.3.1 and 8.3.3.

8.4.1 Enhancing the Speed of the ETL Framework Using Other Techniques

A classified-fragmentation component to speed up the report generation of a data warehouse project was discussed in section 5.6. This component is added to the research as a proof to the extensibility concept of the framework, and did not provide sophisticated algorithms to handle the performance issue. Future research work can be dedicated to handle the performance problems of the ETL framework in general, and to handle the problems of the report generation speed as an advance research topic.

8.4.2 Extending the New ETL Framework with New Components and New Data Types

The new ETL framework is extended by adding a classified-fragmentation component to speed up the report generation of a data warehouse project. In addition, the data types used in conducting this research are limited to text, image and video. Future research works can be conducted to extend the framework by other components to suit other specific ETL business needs. In addition, data types other than text, image and video can be included according to the specific business needs of these future research works.

8.4.3 Combining the ETL Framework with a Complete SOA-Based Business Intelligence Framework

Business intelligence technologies include many data warehousing technologies such as ETL, reporting, ad-hoc querying, online analytical processing (OLAP) and others. A potential future research work is to define a theoretical framework for distributing the components of all these business intelligence technologies into interoperable distributed components, and to add the ETL framework developed in this research as a model in the complete business intelligence framework.

8.5 Final Remarks

This research has identified the gaps and the problems of the current ETL framework. It solved the problems and filled the gaps of the current ETL framework by providing the new SOA based ETL framework. The framework was defined diagrammatically and textually to ease the understanding of the framework

components. In addition, a tested and evaluated prototype has validated the theoretical framework.

Although many technologies can be used to implement an ETL tool based on the new ETL framework, however all of the technologies used in implementing the prototype of this research are open source technologies. Open source technologies are adopted to enable free distribution and implementation of the SOA-based ETL prototype, so that developers can use it as a base for their future ETL tools development that are based on the new ETL framework. Open source technologies used to implement the prototype are: Java, J2EE, MySQL, Apache Tomcat web server, and GlassFish application server. On top of free availability, open source technologies are more flexible because the source code is available in case that there is a need to extend or enhance any part of the code of those technologies.

Although this thesis provides “framework for interoperable and distributed extraction-transformation-loading (ETL) based on service oriented architecture”, it has not only filled the existing distribution and interoperability gaps. The research has also contributed to an increased understanding of how a loosely coupled software framework can be built using service oriented architecture. This is an important issue as more software frameworks need to be redefined to eliminate the problems that arise due to the tight coupling feature of the software framework components.

REFERENCES

- Abrahiem, R. (2007). A New Generation of Middleware Solutions for a Near-Real-Time Data Warehousing Architecture. *Proceedings of Electro/Information Technology International Conference*.
- Agrawal, H., Chafle, G., Goyal, S., Mittal, S., & Mukherjea, S. (2008). An Enhanced Extract-Transform-Load System for Migrating Data in Telecom Billing. *Proceedings of the 2008 IEEE 24th International Conference on Data Engineering*.
- Agrawal, R., Bayardo, R. J., Gruhl, D., & Papadimitriou, S. (2002). Vinci: A service-oriented architecture for rapid development of web applications. *Computer Networks*, 39(5), 523-539.
- Albrecht, A., & Naumann, F. (2008). Managing ETL processes. *Proceedings of New Trends in Information Integration (NTII) Workshop*.
- Almeida, J., Almeida, V., Ardagna, D., Francalanci, C., & Trubian, M. (2006). Resource management in the autonomic service-oriented architecture. *Proceedings of IEEE International Conference on Autonomic Computing, 2006. ICAC'06*.
- Almeida, M. S., Ishikawa, M., Reinschmidt, J., & Roeber, T. (1999). *Getting started with Data Warehouse and Business Intelligence* (1st ed.). San Jose, California, USA: International Business Machines Corporation.
- Apache. (2010). Apache JMeter. Retrieved 22/9/2009, from <http://jakarta.apache.org/jmeter/>.
- Armstrong, E., Ball, J., Bodoff, S., Carson, D. B., Evans, I., Green, D., *et al.* (2004). *The J2EE™ 1.4 Tutorial* (2nd ed.). San Antonio Road Palo Alto, CA, USA: Sun Microsystems.
- Atkinson, T. K. H. (2002). Rearchitecting the UML Infrastructure. *ACM Transactions on Modeling and Computer Simulation*, 12(4), 290-321.
- Ault, M. (2003). *Oracle Data Warehouse Management: Secrets of Oracle Data Warehousing* (1st ed.). North Carolina, USA: Rampant Techpress.
- Badoiu, A., Petrescu, S., Vlad, V., & Botu, A. (2008). Information System for the Management of the Health Services in Romania. *Proceedings of IEEE International Conference for Robotics Automation, Quality and Testing*.
- Bala, H., Venkatesh, V., Venkatraman, S., Bates, J., & Brown, S. H. (2009). Disaster Response in Health Care: A Design Extension for Enterprise Data Warehouse. *communications of the acm*, 1(52), 136-140.
- Bâra, A., Lungu, I., Velicanu, M., Diaconita, V., & Botha, I. (2008). Improving query performance in virtual data warehouses. *WSEAS Transactions on Information Science and Applications*, 5(5), 632-641.
- Barai, M., Binildas, & Caselli, V. (2008). *Service Oriented Architecture with Java* (1st ed.). Birmingham, UK: Packt Publishing.

- Barton, B., & Campbell, E. (2007). Implementing a Professional Services Organization Using Type C Scrum. *Proceedings of 40th Hawaii International Conference on System Sciences*.
- Bertrand, F., Bramley, R., Sussman, A., Bernholdt, D. E., Kohl, J. A., Larson, J. W., *et al.* (2005). Data redistribution and remote method invocation in parallel component architectures. *Proceedings of 19th IEEE International Parallel and Distributed Processing Symposium*.
- Blair, G. S., Coulson, G., Robin, P., & Papathomas, M. (2009). An architecture for next generation middleware. *Proceedings of the IFIP International Conference on Distributed Systems Platforms and Open Distributed Processing*, 191-206.
- Bonifati, A., Casati, F., Dayal, U., & Shan, M. C. (2001). Warehousing workflow data: Challenges and opportunities. *Proceedings of the International Conference on Very Lagr Databases*, 649-652.
- Brace, I. (2008). *Questionnaire Design: How to Plan, Structure and Write Survey Material for Effective Market Research* (2nd ed.). London, UK: Kogan Page.
- Brown, A., Johnston, S., & Kelly, K. (2003). Using service-oriented architecture and component-based development to build web service applications. *interactions*, 1(1), 2-16.
- Bruckner, R. M., List, B., & Schiefer, J. (2002). Striving towards near real-time data integration for data warehouses. *Lecture notes in computer science*, 1(1), 317-326.
- Bugatti, P. H., Ribeiro, M. X., Traina, A. J. M., & Jr, C. T. (2008). Content-based Retrieval of Medical Images by Continuous Feature Selection. *Proceedings of 21st IEEE International Symposium on Computer-Based Medical Systems*.
- Channabasavaiah, K., Holley, K., & Tuggle, E. (2003). Migrating to a service-oriented architecture. *IBM DeveloperWorks*, 1(1), 1-23.
- Cheng, H. K., Tang, Q. C., & Zhao, a. J. L. (2006). Web Services and Service-Oriented Application Provisioning: An Analytical Study of Application Service Strategies. *Proceedings of IEEE International Conference on Transactions on Engineering Management*.
- Chester, T. M. (2001). Cross-Platform Integration with XML and SOAP. *IT Pro Journal*, 3(5), 26 - 34.
- Cleveland, F. M. (2002). Information Exchange Modeling (IEM) and extensible Markup Language (XML) Technologies. *Proceedings of IEEE Power Engineering Society Winter Meeting, 2002*.
- Coulouris, G., Dallimore, J., & Kindberg, T. (2001). *Distributed systems: concepts and design* (3rd ed.). Shanghai,China: China Machine Press.

- Cristal, M., Wildt, D., & Prikladnicki, R. (2008). Usage of SCRUM Practices within a Global Company. *Proceedings of IEEE International Conference on Global Software Engineering*.
- Curbera, F., Duftler, M., Khalaf, R., Nagy, W., Mukhi, N., & Weerawarana, S. (2002). Unraveling the Web services web: an introduction to SOAP, WSDL, and UDDI. *IEEE Internet Computing*, 6(2), 86-93.
- Darmont, J., & Boussaid, O. (2006). *Processing and managing complex data for decision support* (1st ed.). Hershey, USA: IGI Global.
- Darmont, J., Boussaid, O., Ralaivao, J. C., & Aouiche, K. (2005). An architecture framework for complex data warehouses. *Proceedings of 7th International Conference on Enterprise Information Systems*.
- Derong, S., Ge, Y., Yu, C., Yue, K., & Tiezheng, N. (2005). An Effective Web Services Discovery Strategy for Web Services Composition. *Proceedings of the 2005 The Fifth International Conference on Computer and Information Technology (CIT'05)*.
- Dessloch, S., Hemaidezt, M. A., Wisneskyl, R., Radwan, A., & Zhou, J. (2008). Orchid: Integrating Schema Mapping and ETL. *Proceedings of IEEE Data Engineering 24th International Conference*.
- Dou, A. J., Lin, S., & Kalogeraki, V. (2008). Real-Time Querying of Historical Data in Flash-equipped Sensor Devices. *Real-Time Systems Symposium*.
- Du, D., & Raghavendra, C. (2005). *Distributed Network Systems* (2nd ed.). California, USA: Springer.
- Du, T. C., & Wong, J. (2004). Designing Data Warehouses for Supply Chain Management. *Proceedings of IEEE International Conference on E-Commerce Technology*.
- Dung, T. Q., & Kameyama, W. (2007). A Proposal of Ontology-based Health Care Information Extraction System: VnHIES. *Proceedings of Research, Innovation and Vision for the Future, 2007 IEEE International Conference*.
- Erl, T. (2004). *Service-oriented architecture: a field guide to integrating XML and web services* (1st ed.). NJ, USA: Prentice Hall.
- Gao, A., Yang, D., & Tang, S. Web service composition based on message schema analysis. *Advances in Databases: Concepts, Systems and Applications*, 4443(5), 918-923.
- Gargantini, E. R., and Patrizia Scandurra. (2007). Deriving a textual notation from a metamodel by University of Bergamo. Retrieved 1/12/2011, from https://doc.telin.nl/dsweb/Get/Rendition-50041/3M4MDA_2006_online_proceedings.pdf#page=41.

- Green, P. F., Indulska, M. K., Rosemann, M., & Weber, R. A. (2003). Will XML technologies and web services solve the interoperability problem? *Proceedings of International Workshop on Utility, Usability and Complexity of Emergent IS.*, 103-115.
- Hau, T., Ebert, N., Hochstein, A., & Brenner, W. (2008). Where to Start with SOA Criteria for Selecting SOA Projects. *Proceedings of 41st Hawaii International Conference on System Sciences.*
- He, B., Wang, R., Chen, Y., Lelescu, A., & Rhodes, J. (2007). BIwTL: a business information warehouse toolkit and language for warehousing simplification and automation. *Proceedings of the 2007 ACM SIGMOD international conference on Management of data.*
- Heinzl, S., Mathes, M., Friese, T., Smith, M., & Freisleben, B. (2006). Flex-SwA: Flexible Exchange of Binary Data Based on SOAP Messages with Attachments. *Proceedings of IEEE International Conference on Web Services (ICWS'06).*
- Henry, S., Hoon, S., Hwang, M., Lee, D., & DeVore, M. D. (2005). Engineering Trade Study: Extract, Transform Load Tools for Data Migration. *Proceedings of the 2005 Systems and Information Engineering Design Symposium.*
- Hevner, A. R., & Chatterjee, S. (2010). *Design Research in Information Systems* (1st ed.). CA, USA: Springer.
- Hevner, A. R., & March, S. T. (2003). The Information Systems Research Cycle. *MIS Quarterly* (November), 36(11), 111 - 113.
- Holzer, S., Tafazzoli, A. G., Altmann, U., Wachter, W., & Dudeck, J. (1999). *Data warehousing as a tool for quality management in oncology* (1st ed.). NY, USA: IOS Press.
- IBM. (2010). InfoSphere DataStage. Retrieved 15/2/2010, from <http://www-01.ibm.com/software/data/infosphere/datastage/>.
- Information, C. I. f. H. (2009). Regrouping Historical Data CIHI Reference Document. *Canadian Institute for Health Information* Retrieved 12/2/2010, from http://www.cihi.ca/CIHI-ext-portal/pdf/internet/REGROUPING_HIST_DATA_REF_EN.
- Inmon, W. H. (2005). *Building the Data Warehouse* (4th ed.). Indiana, USA: Wiley Publishing, Inc.
- Iqbal, T., & Daudpota, N. (2006). XML based framework for ETL processes for relational databases. *WSEAS Transactions on Information Science and Applications*, 3(7), 1402-1406.

- ISO. (1996). *Extended Backus-Naur Form (EBNF)* Retrieved 30/10/2011, from http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=26153.
- Issarny, V., Kloukinas, C., Zarras, A., & Architectures, M. (2008). Management Group's Common Object Request Broker (CORBA). *Microsoft's Distributed Component Object Model*.
- Jerstad, I., Dustdar, S., & Thanh, D. V. (2005). A service oriented architecture framework for collaborative services. *Proceedings of 14th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprise, 2005*.
- Johansen, M. A., Scholl, J., Hasvold, P., Ellingsen, G., & Bellika, J. G. (2008). "Garbage In, Garbage Out"- Extracting Disease Surveillance Data from EPR Systems in Primary Care. *Proceedings of 2008 ACM conference on Computer supported cooperative work*.
- Judy, K. H., & Krumins-Beens, I. (2008). Great Scrums Need Great Product Owners: Unbounded Collaboration and Collective Product Ownership. *Proceedings of 41st Hawaii International Conference on System Sciences*.
- Katifori, A., Torou, E., Vassilakis, C., & Halatsis, C. (2008). Supporting Research in Historical Archives: Historical Information Visualization Modeling Requirements. *Proceedings of 12th International Conference Information Visualisation*.
- Kimball, R., & Caserta, J. (2004). *The Data Warehouse ETL Toolkit* (2nd ed.). Indiana, USA: Wiley Publishing, Inc.
- Klmek, J., Kopenec, L., Loupal, P., & Mal, J. (2010). XCase - A Tool for Conceptual XML Data Modeling. *Proceedings of Advances in Databases and Information Systems Conference*.
- Kobryn, C. (2000). Modeling Components and Frameworks with UML. *Communications of the ACM*, 43(10).
- Kruchten, W. K., Bran, & Selic. (2001). Describing Software Architecture with UML. *Rational Software*.
- Kshemkalyani, A. D., & Singhal, M. (2008). *Distributed Computing Principles, Algorithms, and Systems* (1st ed.). Cambridge, UK: Cambridge University Press.
- Kumari, G. P., Kandan, B., & Mishra, A. K. (2008). Experience sharing on SOA based Heterogeneous Systems Integration. *IEEE Congress on Services 2008 - Part I*, 11(6), 107 - 108.
- Lam, T., & Minsky, N. (2010). Regulating Orchestration in SOA-Based Systems. *Proceedings of 2010 Seventh International Conference on Information Technology*, 690-695.

- Laskey, K. B., & Laskey, K. (2009). *Service oriented architecture*. CA, USA: Computational Statistics.
- Li, G., Muthusamy, V., & Jacobsen, H. A. (2010). A distributed service-oriented architecture for business process execution. *ACM Transactions on the Web (TWEB)*, 4(1), 1-33.
- Liao, H., Padmanabhan, S., Srinivasan, S., Lau, P., Shan, J., & Wisnesky, R. (2008). Bringing Business Objects into Extract-Transform-Load (ETL) Technology. *Proceedings of IEEE International Conference on e-Business Engineering*.
- Louridas, P. (2006). SOAP and Web Services. *IEEE Software*.
- Lujanmora, J. T. (2004). Physical Modeling of Data Warehouses using UML. *ACM Journal*.
- Maassen, J., Nieuwpoort, R., Veldema, R., Bal, H. E., & Plaat, A. (2008). *Java Remote Method Invocation provides an unusually flexibility*. Indiana, USA: Wiley Publishing, Inc.
- Mahboubi, H., & Darmont, J. e. o. (2009). Enhancing XML Data Warehouse Query Performance by Fragmentation. *2009 ACM symposium on Applied Computing*.
- Marchenko, A., & Abrahamsson, P. (2008). Scrum in a Multiproject Environment: An Ethnographically-Inspired Case Study on the Adoption Challenges. *Proceedings of Agile 2008 Conference*.
- Maskat, R., & Shamsudin, M. F. (2008). Proposing a Physical Model for Malaysian Health Plan Data. *Proceedings of International Conference on Advanced Computer Theory and Engineering*.
- Massachusetts, E. D. (2008). *Introduction to the Data Warehouse* (1st ed.). Malden, USA: Massachusetts.
- Matsumura, I., Ishida, T., Murakami, Y., & Fujishiro, Y. (2006). Situated Web Service: Context-Aware Approach to High-Speed Web Service Communication. *Proceedings of IEEE International Conference on Web Services (ICWS'06)*.
- Maurizio, A., Sager, J., Jones, P., Corbitt, G., & Girolami, L. (2008). Service Oriented Architecture: Challenges for Business and Academia. *Proceedings of 41st Hawaii International Conference on System Sciences*.
- McCabe, M. C., & Grossman, D. (1996). The role of tools in development of a data warehouse. *Proceedings of the 4th International Symposium on Assessment of Software Tools*.
- Microsoft. (2009). Microsoft Business Intelligence. Retrieved 5/8/2009, from <http://www.microsoft.com/bi/>.
- Microsoft. (2010). SQL Server Integration Services. Retrieved 15/1/2010, from <http://msdn.microsoft.com/en-us/library/ms141026.aspx>.

- Mitchell, S., Blake, M. B., Cunningham, D., & Gopalan, S. (2008). A SOA-Driven Content Discovery and Retrieval Platform. *Proceedings of 10th IEEE Conference on E-Commerce Technology*.
- Morris, H., Liao, H., Padmanabhan, S., Srinivasan, S., Lau, P., Shan, J., *et al.* (2008). Bringing Business Objects into Extract-Transform-Load (ETL) Technology. *Proceedings of the 2008 IEEE International Conference on e-Business Engineering*.
- Mrunalini, M., Kumar, T. V. S., & Kanth, K. R. (2009). Simulating Secure Data Extraction in Extraction Transformation Loading (ETL) Processes. *Proceedings of the 2009 Third UKSim European Symposium on Computer Modeling and Simulation*.
- Mulik, S., Ajgaonkar, S., & Sharma, K. (2008). Where Do You Want to Go in Your SOA Adoption Journey? *IEEE Computer Society Washington, DC, USA*.
- Mundy, J., Thornthwaite, W., & Kimball, R. (2006). *The Microsoft data warehouse toolkit: with SQL Server 2005 and the Microsoft Business Intelligence toolset*: Wiley Pub.
- Muñoz, L., Mazón, J.-N., & Trujillo, J. (2009). Measures for ETL processes models in data warehouses. *Proceedings of the first international workshop on Model driven service engineering and data quality and security*.
- Mykknen, J., Porrasmaa, J., Rannanheimo, J., & Korpela, M. (2003). A process for specifying integration for multi-tier applications in healthcare. *International journal of medical informatics*, 70(2-3), 173-182.
- Natis, Y. (2003). Service-oriented architecture scenario. *Gartner, Inc., Stamford*.
- Nelson, G., & Wright, J. (2005). Real time decision support: creating a flexible architecture for real time analytics. *DSSResources. COM*, 11(4), 18-32.
- NetBeans. (2010). NetBeans IDE. Retrieved 25/4/2010, from <http://netbeans.org/>.
- Newcomer, E. (2002). *Understanding Web Services: XML, Wsdl, Soap, and UDDI*: Addison-Wesley Professional.
- Newcomer, E., & Lomow, G. (2004). *Understanding SOA with Web Services (Independent Technology Guides)* (1st ed.). USA: Addison-Wesley Professional.
- Niehaves, B., & Becker, J. (2006). Design Science Perspective on IT-Consulting. *Tagungsband / der Multikonferenz Wirtschaftsinformatik*.
- OMG. (2003). *Common Warehouse Metamodel (CWM) Specification* (1st ed. Vol. 1). MA, USA: OMG Headquarters.
- OMG. (2011). Unified Modeling Language. Retrieved 29/10/2011, from <http://www.uml.org/>.

- Oracle. (2009). Oracle Enterprise Performance Management and Business Intelligence. Retrieved 10/8/2009, from http://www.oracle.com/solutions/business_intelligence/index.html.
- Oracle. (2010). Oracle Warehouse Builder. Retrieved 3/2/2010, from <http://www.oracle.com/technetwork/developer-tools/warehouse/overview/index.html>.
- Papazoglou, M. P. (2003). Service-oriented computing: Concepts, characteristics and directions. *Proceedings of the Fourth International Conference on Web Information Systems Engineering*.
- Patasiene, I., Kregzdyte, R., Patasius, M., Patasius, J., & Kazakeviciute, A. (2007). Integrating Global Data into Local Health Data Base. *Proceedings of 29th Annual International Conference of the IEEE EMBS Cité Internationale*.
- Pentaho. (2006). Pentaho Data Integration: Spoon 2.3.1, User Manual. Retrieved 3/1/2009 from <http://www.pentaho.org>.
- Pentaho. (2009). Pentaho Business Intelligence. Retrieved 2/7/2009, from <http://www.pentaho.com>.
- Perin, F. (2009). Enabling the Evolution of J2EE Applications through Reverse Engineering and Quality Assurance. *Proceedings of the 2009 16th Working Conference on Reverse Engineering*.
- Phan, T., Han, J., Schneider, J.-G., Ebringer, T., & Rogers, T. (2008). A survey of policy-based management approaches for Service Oriented Systems. *Proceedings of 19th Australian Conference on Software Engineering*.
- Priebe, T., & Pernul, G. (2003). Towards integrative enterprise knowledge portals. *Proceedings of the twelfth international conference on Information and knowledge management*.
- Qiu, B., Liu, Y., Ong, Y. S., Gooi, H. B., & Chen, S. (2002). Managing Metadata over the WWW using eXtensible Markup Language (XML). *Power Engineering Society Winter Meeting, 2002*.
- Rayhan, S. H., & Haque, N. (2008). Incremental Adoption of Scrum for Successful Delivery of an IT Project in a Remote Setup. *Proceedings of Agile 2008 Conference*.
- Roach, T., Low, G., & D'Ambra, J. (2008). CAPSICUM - A Conceptual Model for Service Oriented Architecture. *IEEE Congress on Services 2008 - Part I*.
- Roy, N., Pallapa, G., & Das, S. K. (2008). An Ontology-Driven Ambiguous Contexts Mediation Framework for Smart Healthcare Applications. *Proceedings of the 1st international conference on Pervasive Technologies Related to Assistive Environments*.

- Sahama, T. R., & Croll, P. R. (2007). A Data Warehouse Architecture for Clinical Data Warehousing. *Proceedings of 2007 Conference of Research and Practice in Information Technology*.
- Salter, D., & Jennings, F. (2008). *Building SOA-Based Composite Applications Using NetBeans IDE 6* (1st ed.). USA: PACKT Publishing.
- Santos, R. J., & Bernardino, J. (2008). Real-Time Data Warehouse Loading Methodology. *Proceedings of the 2008 international symposium on Database engineering & applications*.
- SAS. (2010). SAS Enterprise Data Integration Server. Retrieved 15/1/2010, from <http://www.sas.com/technologies/dw/entdiserver/index.html>.
- Schwaber, K. (2009, 2/3/2009). Scrum Development Process. from <http://jeffsutherland.com/oops/schwapub.pdf>.
- Sellis, T. (2006). Formal specification and optimization of ETL scenarios. *Proceedings of the 9th ACM international workshop on Data warehousing and OLAP*.
- Sen, A., & Sinha, A. P. (2007). Toward Developing Data Warehousing Process Standards: An Ontology-Based Review of Existing Methodologies. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions* 37(1), 17 - 31.
- Shah, R. C., Roy, S., Jain, S., & Brunette, W. (2003). Data mules: Modeling and analysis of a three-tier architecture for sparse sensor networks. *Ad Hoc Networks*, 1(2-3), 215-233.
- Shaikh, M. U., Malik, S. U. R., Qureshi, A., & Yaqoob, S. (2010). Intelligent Decision Making Based on Data Mining Using Differential Evolution Algorithms and Framework for ETL Workflow Management. *Proceedings of the 2010 Second International Conference on Computer Engineering and Applications - Volume 01*.
- Shani, U., Sela, A., Akilov, A., Skarbovski, I., & Berk, D. (2006). A scalable heterogeneous solution for massive data collection and database loading. *Proceedings of the 1st international conference on Business intelligence for the real-time enterprises*.
- Shil, A. B., & Ahmed, M. B. (2006). Additional Functionalities to SOAP, WSDL and UDDI for a Better Web Services' Administration. *Proceedings of Information and Communication Technologies Conference*.
- Silvers, F. (2008). *Building and Maintaining a Data Warehouse* (1st ed.). Philadelphia, USA: Taylor & Francis Group, LLC.
- Simitsis, A., Skoutas, D., & Castellanos, M. (2010). Representation of conceptual ETL designs in natural language using Semantic Web technology. *Data & Knowledge Engineering: Elsevier Science Publishers*, 69(1), 96 -15.

- Simitsis, A., Vassiliadis, P., Terrovitis, M., & Skiadopoulos, S. (2005). Graph-based modeling of ETL activities with multi-level transformations and updates. *Lecture notes in computer science*, 3589(1), 43-61.
- Simitsis, A., Wilkinson, K., Castellanos, M., & Dayal, U. (2009). QoX-driven ETL design: reducing the cost of ETL consulting engagements. *SIGMOD '09: Proceedings of the 35th SIGMOD international conference on Management of data*.
- Siqueira, T. L. L., Ciferri, R. R., Times, V. C., & Ciferri, C. D. d. A. (2009). A Spatial Bitmap-based Index for Geographical Data Warehouses. *Proceedings of the 2009 ACM symposium on Applied Computing*.
- Skoutas, D., Simitsis, A., & Sellis, T. (2009). Ontology-driven conceptual design of ETL processes using graph transformations. *Journal on Data Semantics XIII*, 5530(1), 120-146.
- Sneed, H. M. (2006). Integrating legacy Software into a Service oriented Architecture. *Proceedings of the 10th European Conference on Software Maintenance and Reengineering*.
- Sprott, D., & Wilkes, L. (2004). Understanding service-oriented architecture. Retrieved 25/3/2011, from <http://msdn.microsoft.com/en-us/library/aa480021.aspx>.
- Stal, M. (2006). Using architectural patterns and blueprints for service-oriented architecture. *IEEE software*, 23(2), 54-61.
- Stojanovic, Z., Dahanayake, A., & Sol, H. (2004). Modeling and design of service-oriented architecture. *Proceedings of 2004 IEEE International Conference on Systems, Man and Cybernetics*.
- Stonebraker, M., & Hellerstein, J. M. (2001). Content integration for e-business. *ACM SIGMOD Record*, 30(2), 552-560.
- Sulaiman, T., Barton, B., & Blackburn, T. (2006). Proceedings of AgileEVM – Earned Value Management in Scrum Projects. *AGILE 2006 Conference*.
- Sullins, B. G., & Whipple, M. B. (2005). *EJB Cookbook* (1st ed.). Philadelphia, USA: Manning.
- Sun-Microsystems. (2010). Glassfish. Retrieved 25/4/2010, from <https://glassfish.dev.java.net/>.
- Sutherland, J., Schoonheim, G., & Rijk, M. (2009). Fully Distributed Scrum: Replicating Local Productivity and Quality with Offshore Teams. *Proceedings of 42nd Hawaii International Conference on System Sciences*.
- Sutherland, J., Schoonheim, G., Rustenburg, E., & Rijk, M. (2008). Fully Distributed Scrum: The Secret Sauce for Hyperproductive Offshored Development Teams. *Proceedings of Agile 2008 Conference*.

- Sutherland, J., Viktorov, A., Blount, J., & Puntikov, N. (2007). Distributed Scrum: Agile Project Management with Outsourced Development Teams. *Proceedings of 40th Hawaii International Conference on System Sciences*.
- Suzumura, T., Yasue, T., & Onodera, T. (2010). Scalable performance of system S for extract-transform-load processing. *Proceedings of the 3rd Annual Haifa Experimental Systems Conference*.
- Tam, L. N. (2010). *IBM Data Warehouse and Business Intelligence Solutions* (1st ed.). Vietnam: IBM Software Group.
- Tanenbaum, A. S., & Van Steen, M. (2002). *Distributed systems* (2nd ed.). New Jersey, USA: CiteSeer.
- Tellis, W. (1997). *Application of a Case Study Methodology* (3rd ed.). CA: Sage Publishing.
- Temenos. (2005). *Data Warehouse* (2nd ed.). NY, USA: Temenos USA Inc.
- Theodosi, A. D., & Tsihrantzis, G. A. (2008). Using Agents for Feature Extraction: Content Based Image Retrieval for Medical Applications. *Manuscript*.
- Thomsen, C., & Pedersen, T. B. (2009). A powerful programming framework for extract-transform-load programmers. *Proceedings of the ACM twelfth international workshop on Data warehousing and OLAP*.
- Trujillo, J., & Lujnmore, S. (2003). A UML based approach for modeling ETL processes in data warehouses. *Lecture Notes in Computer Science*, 22(5), 307-320.
- Tsai, W. T., Paul, R., Wang, Y., Fan, C., & Wang, D. (2002). Extending WSDL to Facilitate Web Services Testing. *Proceedings of the 7th IEEE International Symposium on High Assurance Systems Engineering (HASE'02)*.
- Tsenov, M. (2007). Example of communication between distributed network systems using web services. *Proceedings of the 2007 international conference on Computer systems and technologies*.
- Tziovara, V., Vassiliadis, P., & Simitsis, A. (2007). Deciding the Physical Implementation of ETL Workflows. *Proceedings of ACM tenth international workshop on Data warehousing and OLAP*.
- Urgaonkar, B., Pacifici, G., Shenoy, P., Spreitzer, M., & Tantawi, A. (2005). *An analytical model for multi-tier internet services and its applications*.
- Uy, E., & Ioannou, N. (2008). Growing and Sustaining an Offshore Scrum Engagement. *Proceedings of Agile 2008 Conference*.
- Vaishnavi, V., & Kuechler, B. (2004). Design Research in Information Systems. from <http://home.aisnet.org/displaycommon.cfm?an=1&subarticlenbr=279#designResearchMethodology>.

- Vara, J. M., Castro, V. d., & Marcos, E. (2005). WSDL automatic generation from UML models in a MDA framework. *Proceedings of the International Conference on Next Generation Web Services Practices (NWeSP'05)*.
- Vassiliadis, P., Simitsis, A., & Skiadopoulos, S. (2002). Conceptual modeling for ETL processes. *Proceedings of the 5th ACM international workshop on Data Warehousing and OLAP*.
- Vassiliadis, P., Simitsis, A., Terrovitis, M., & Skiadopoulos, S. (2005). Blueprints and measures for ETL workflows. *Lecture notes in computer science*, 28(3), 380-385.
- Vitt, E., Luckevich, M., & Misner, S. (2002). *Business intelligence*: Microsoft Press.
- Voth, G. R., Kindel, C., & Fujioka, J. (1998). Distributed application development for three-tier architectures: Microsoft on Windows DNA. *IEEE Internet Computing*, 2(2), 41-45.
- W3C. (1999). XML Schema Requirements. Retrieved 10/12/2009, from <http://www.w3.org/TR/NOTE-xml-schema-req>.
- W3C. (2010). The amazing em unit and other best practices. Retrieved 1-1-2010, from <http://www.w3.org/WAI/GL/css2em.htm>.
- Wang, C., & Liu, S. (2008). SOA Based Electric Power Real-time Data Warehouse. *Proceedings of Workshop on Power Electronics and Intelligent Transportation System*.
- Watson, H. J., & Wixom, B. H. (2007). The current state of business intelligence. *COMPUTER-IEEE COMPUTER SOCIETY-*, 40(9), 96.
- Weerawarana, S., Curbera, F., Leymann, F., Storey, T., & Ferguson, D. F. (2005). *Web Services Platform Architecture: SOAP, WSDL, WS-Policy, WS-Addressing, WS-BPEL, WS-Reliable Messaging and More* (1st ed.). NY, USA: Prentice Hall.
- Wehrle, P., Miquel, M., & Tchounikine, A. (2005). A Model for Distributing and Querying a Data Warehouse on a Computing Grid. *Proceedings of 2005 11th International Conference on Parallel and Distributed Systems*.
- Wehrle, P., Miquel, M., & Tchounikine, A. (2007). A Grid Services-Oriented Architecture for Efficient Operation of Distributed Data Warehouses on Globus. *Proceedings of 21st International Conference on Advanced Networking and Applications*.
- Werner, C., Buschmann, C., & Fischer, S. (2004). Compressing SOAP Messages by using Differential Encoding. *Proceedings of the IEEE International Conference on Web Services (ICWS'04)*.
- Wolter, R. (2001). XML Web services basics. *Microsoft Developer Network*, 12(1), 66-86.

- Wrembel, R., & Koncilia, C. (2007). *Data Warehouses and OLAP: Concepts, Architectures and Solutions* (1st ed.). NY, USA: IRM Press.
- Wu, L., Barash, G., & Bartolini, C. (2007). A Service-oriented Architecture for Business Intelligence. *Proceedings of IEEE International Conference on Service-Oriented Computing and Applications (SOCA'07)*.
- Xi, X., & Hongfeng, X. (2009). Developing a Framework for Business Intelligence Systems Integration Based on Ontology. *International Conference on Networking and Digital Society*.
- Yang, C. L., Chang, Y. K., & Chu, C. P. (2008). A Gateway Design for Message Passing on The SOA Healthcare Platform. *IEEE International Symposium on Service-Oriented System Engineering*.
- Yin, R. (1994). *Case study research: Design and methods* (2nd ed.). Thousand Oaks, CA: Sage Publishing.
- Yingying, X., Hao, T., & Peiren, Z. (2003). An Advanced Text-To-Speech Server System Based on SOAP Protocol. *Proceedings of International Conference on Acoustics, Speech, and Signal Processing (ICASSP '03)*.
- Yong Xia, M. G. (2002). Rigorous EBNF-based Definition for a Graphic Modeling Language. *Winterthurerstr. 190, CH-8057 Zurich, Switzerland*.
- Zhang, & Gracanin, D. (2008). Service-Oriented-Architecture based Framework for Multi-User Virtual Environments. *Proceedings of 2008 Winter Simulation Conference*.
- Zhang, & Wang, S. (2008). A Framework Model Study for Ontology-driven ETL Processes. *Proceedings of 4th International Conference on Wireless Communications, Networking and Mobile Computing*.
- Zhang, L. J., Zhou, N., Chee, Y. M., Jalaldeen, A., Ponnalagu, K., Sindhgatta, R. R., *et al.* (2008). SOMA-ME: A platform for the model-driven design of SOA solutions. *IBM Systems Journal*, 47(3), 397-413.
- Zhang, Q., Li, K., & Yu, J.-h. (2006). Application of Multi-Agent System On Web-Based Data Warehouse for Pricing System of Power Supplier. *Proceedings of Power Systems Conference and Exposition, 2006. PSCE '06*.
- Zhou, X., Liu, B., Wang, Y., Zhang, R., Li, P., Chen, S., *et al.* (2008). Building Clinical Data Warehouse for Traditional Chinese Medicine Knowledge Discovery. *Proceedings of International Conference on BioMedical Engineering and Informatics*.
- Zhu, Y., An, L., & Liu, S. (2008a). Data Updating and Query in Real-time Data Warehouse System. *Proceedings of International Conference on Computer Science and Software Engineering*.

Zhu, Y. Q., Min, B., & Wei, H. (2008b). The Research of Methodology in Models Mapping for ETL Processes Based on Model Driven. *Information Science and Engineering, 2008. ISISE '08. International Symposium*.

Appendix A

Details of the Prototype Design

A.1 Details of Database Design

A.1.1 Data Source Schema (CLINIC Database)

a. PATIENTS TABLE

The personal details of every patient are recorded in PATIENTS table that has the following fields:

- **PATIENT_ID:** Patient Id, the data type is NUMBER, this field is a primary key on the table and it is a sequence that increases field value when a new patient is added.
- **P_NAME:** Patient Name, the data type is TEXT, this field stores patient's name, and it must be not NULL.
- **CITY_ID:** City Id, the data type is NUMBER, this field is a foreign key belong to CITY table that specify the patient's STATE and CITY, and it must be not NULL.
- **P_PHONE:** Patient Phone, the data type is TEXT, this field stores patient's phone, and it must be not NULL.
- **GENDER_ID:** Gender Id, the data type is NUMBER, this field is a foreign key belong to GENDER table that specify the patient's sex, and it must be not NULL.
- **P_BIRTHDAY:** Patient Birthday, the data type is TEXT, this field stores patient's birthday, and it must be not NULL.
- **P_WEIGHT:** Patient Weight, the data type is TEXT, this field store patient's weight, and it must be not NULL.

b. PHYSICIANS TABLE

The personal details of every physician are recorded in Physicians table that has the following fields:

- **PHY_ID:** Physician Id, the data type is NUMBER, this field is a primary key on the table and it is a sequence that increases field value when a new physician is added.
- **PHY_NAME:** Physician Name, the data type is TEXT, this field stores physician's name, and it must be not NULL.
- **PHY_BIRTHDAY:** Physician Age, the data type is TEXT, this field stores physician's birthday, and it must be not NULL.
- **PHY_PHONE:** Phone Id, the data type is TEXT, this field stores physician's phone, and it must be not NULL.
- **PHY_MAJOR:** Major Id, the data type is TEXT, this field stores physician's major, and it must be not NULL.
- **DEPT_ID:** Department Id, this field is a foreign key belong to DEPARTMENTS table that specify the physician's Department, and it must be not NULL.

c. **DEPARTMENTS TABLE**

This table stores data about clinic departments, Departments table has the following fields:

- **DEPT_ID:** Department Id, the data type is NUMBER, this field is a primary key on the table and it is a sequence that increases field value when a new department is added.
- **DEPT_NAME:** Department Name, the data type is TEXT, this field stores department name, and it must be not NULL.
- **DEPT_DESC:** Department description, the data type is TEXT, this field stores description about department, and it must be not NULL.
- **DEPT_PHONE:** Department Phone, the data type is TEXT, this field stores department phone, and it must be not NULL.

d. **TESTOPERATIONS TABLE**

This table stores data about detail patient's test in detail. A number of tests may be conducted for each patient. Each patient is assigned to one physician. TESTOPERATIONS table has the following fields:

- **TESTO_ID:** Test Operation Id, the data type is NUMBER, this field is a primary key on the table and it is a sequence that increases field value when a new test is added.
- **TESTO_DATE:** Test Operation Date, the data type is TEXT, this field stores date of test performed, and it must be not NULL.
- **TESTO_TIME:** Test Operation Time, the data type is TEXT, this field stores time of test performed, and it must be not NULL.
- **TESTO_RESULT:** Test Operation Result, the data type is TEXT, this field stores patient's case in detail, and it must be not NULL.
- **TESTO_IMG:** Test Operation Image, the data type is TEXT, this field stores URL of image that belongs to patient's test, like X-ray image.
- **TESTO_VIDEO:** Test Operation Video, the data type is TEXT, this field stores URL of video that belongs to patient's test, like ultrasound video.
- **PATIENT_ID:** Patient Id, the data type is NUMBER, this field is a foreign key belong to PATIENTS table that specify the patient personal details, and it must be not NULL.
- **PHY_ID:** Physician Id, the data type is NUMBER, this field is a foreign key belong to PHYSICIANS table that specify the physician personal details, and it must be not NULL.
- **DISEASE_ID:** Disease Id, the data type is NUMBER, this field is a foreign key belong to DISEASE table that specify the patient's disease, and it must be not NULL.
- **MEDICINE_ID:** Medicine Id, the data type is NUMBER, this field is a foreign key belong to MEDICINES table that specify the medicine that treat disease, and it must be not NULL.

e. **DISEASES TABLE**

This table stores data about Diseases. This table has the following fields:

- **DISEASE_ID:** Disease Id, the data type is NUMBER, this field is a primary key on the table and it is a sequence that increases field value when a new disease is added.
- **DISEASE_TITLE:** Disease Title, the data type is TEXT, this field stores disease title, and it must be not NULL.
- **DISEASE_TYPE:** Disease Type, the data type is TEXT, this field indicates disease type, and it must be not NULL.

f. **MEDICINES TABLE**

This table stores data about Medicines each medicine belong to disease. MEDICINES table has the following fields:

- **MEDICINE_ID:** Medicine Id, the data type is NUMBER, this field is a primary key on the table and it is a sequence that increases field value when a new medicine is added.
- **MEDICINE_DESC:** Medicine description, the data type is TEXT, this field stores description about medicine, and it must be not NULL.
- **DISEASE_ID:** Disease Id, this field is a foreign key belong to DISEASE table that specify the patient's disease, and it must be not NULL.

g. **GENDER TABLE**

This table stores data about gender. GENDER table has the following fields:

- **GENDER_ID:** Gender Id, the data type is NUMBER, this field is a primary key on the table and it is a sequence that increases field's value when a new data is added.

- **GENDER_DESC:** Gender description, the data type is TEXT, this field stores description about gender (such as MALE, FEMALE), and it must be not NULL.

h. CITY TABLE

This table stores data about gender. CITY table has the following fields:

- **CITY_ID:** CITY Id, the data type is NUMBER, this field is a primary key on the table and it is a sequence that increases field value when a new city is added.
- **STATE:** the data type is TEXT, this field stores the state name that has the city, and it must be not NULL.
- **CITY:** the data type is TEXT, this field stores the name of city, and it must be not NULL.

A.1.2 EXTRACT TEMP STORAGE Database

a. TEMP TABLE

This table contains many fields that collected from clinic database table.

TEMP table has the following fields:

- **TESTO_RESULT:** Test Operation Result, the data type is NUMBER, the data type is TEXT, this field views patient's case in detail, and it must be not NULL.
- **TESTO_DATE:** Test Operation Date, the data type is TEXT, this field store date of test performed, and it must be not NULL.
- **TESTO_IMG:** Test Operation Image, the data type is TEXT, this field store URL of image that belongs to patient's test, like X-ray image.
- **TESTO_VIDEO:** Test Operation Video, the data type is TEXT, this field store URL of video that belongs to patient's test, like ultrasound video.

- **DISEASE_ID:** Disease Id, the data type is NUMBER, the data type is NUMBER, this field is a foreign key belong to DISEASE table that specify the patient's disease, and it must be not NULL.
- **MEDICINE_ID:** Medicine Id, the data type is NUMBER, this field is a foreign key belong to MEDICINES table that specify the medicine that treat disease, and it must be not NULL.
- **GENDER_ID:** Gender Id, the data type is NUMBER, this field is a foreign key belong to GENDER table that specify the patient's sex, and it must be not NULL.
- **P_BIRTHDAY:** Patient Birthday, the data type is TEXT, this field store patient's birthday, and it must be not NULL.
- **CITY_ID:** City Id, the data type is NUMBER, this field is a foreign key belong to CITY table that specify the patient's STATE and CITY, and it must be not NULL.

b. DISEASES TABLE

This table stores data about Diseases. This table has the following fields:

- **DISEAE_ID:** Disease Id, the data type is NUMBER, this field is a primary key on the table and it is a sequence that increases field value when a new disease is added.
- **DISEAE_TITLE:** Disease Title, the data type is TEXT, this field stores disease title, and it must be not NULL.
- **DISEAE_TYPE:** Disease Type, the data type is TEXT, this field indicates disease type, and it must be not NULL.

c. MEDICINES TABLE

This table stores data about Medicines each medicine belong to disease.

MEDICINES table has the following fields:

- **MEDICINE_ID:** Medicine Id, the data type is NUMBER, this field is a primary key on the table and it is a sequence that increases field's value when a new medicine is added.

- **MEDICINE_DESC:** Medicine description, the data type is TEXT, this field stores description about medicine, and it must be not NULL.
- **DISEAS_ID:** Disease Id, this field is a foreign key belong to DISEASE table that specify the patient's disease, and it must be not NULL.

d. GENDER TABLE

This table stores data about gender. GENDER table has the following fields:

- **GENDER_ID:** Gender Id, the data type is NUMBER, this field is a primary key on the table and it is a sequence that increases field value when a new data is added.
- **GENDER_DESC:** Gender description, the data type is TEXT, this field stores description about gender (such as MALE, FEMAIL), and it must be not NULL.

e. CITY TABLE

This table stores data about gender. CITY table has the following fields:

- **CITY_ID:** CITY Id, the data type is NUMBER, this field is a primary key on the table and it is a sequence that increases field value when a new city is added.
- **STATE:** the data type is TEXT, this field stores the state name that has the city, and it must be not NULL.
- **CITY:** the data type is TEXT, this field stores the name of city, and it must be not NULL.
-

A.1.3 TRANSFORM TEMP STORAGE Database

a. TEMP TABLE

This table contains many fields that collected from clinic database table.

TEMP table has the following fields:

- **TESTO_RESULT:** Test Operation Result, the data type is NUMBER, the data type is TEXT, this field views patient's case in detail, and it must be not NULL.
- **TESTO_DATE:** Test Operation Date, the data type is TEXT, this field store date of test performed, and it must be not NULL.
- **TESTO_IMG:** Test Operation Image, the data type is TEXT, this field store URL of image that belongs to patient's test, like X-ray image.
- **TESTO_VIDEO:** Test Operation Video, the data type is TEXT, this field store URL of video that belongs to patient's test, like ultrasound video.
- **DISEASE_ID:** Disease Id, the data type is NUMBER, the data type is NUMBER, this field is a foreign key belong to DISEASE table that specify the patient's disease, and it must be not NULL.
- **MEDICINE_ID:** Medicine Id, the data type is NUMBER, this field is a foreign key belong to MEDICINES table that specify the medicine that treat disease, and it must be not NULL.
- **GENDER_ID:** Gender Id, the data type is NUMBER, this field is a foreign key belong to GENDER table that specify the patient's sex, and it must be not NULL.
- **P_AGE:** Patient AGE, the data type is TEXT, this field store patient's age, and it must be not NULL.
- **CITY_ID:** City Id, the data type is NUMBER, this field is a foreign key belong to CITY table that specify the patient's STATE and CITY, and it must be not NULL.

b. DISEASES TABLE

This table stores data about Diseases. This table has the following fields:

- **DISEASE_ID:** Disease Id, the data type is NUMBER, this field is a primary key on the table and it is a sequence that increases field's value when a new disease is added.

- **DISEASE_TITLE:** Disease Title, the data type is TEXT, this field stores disease's title, and it must be not NULL.
- **DISEASE_TYPE:** Disease Type, the data type is TEXT, this field indicates disease's type, and it must be not NULL.

c. **MEDICINES TABLE**

This table stores data about Medicines each medicine belong to disease. MEDICINES table has the following fields:

- **MEDICINE_ID:** Medicine Id, the data type is NUMBER, this field is a primary key on the table and it is a sequence that increases field's value when a new medicine is added.
- **MEDICINE_DESC:** Medicine description, the data type is TEXT, this field stores description about medicine, and it must be not NULL.
- **DISEASE_ID:** Disease Id, this field is a foreign key belong to DISEASE table that specify the patient's disease, and it must be not NULL.

d. **GENDER TABLE**

This table stores data about gender. GENDER table has the following fields:

- **GENDER_ID:** Gender Id, the data type is NUMBER, this field is a primary key on the table and it is a sequence that increases field's value when a new data is added.
- **GENDER_DESC:** Gender description, the data type is TEXT, this field stores description about gender (such as MALE, FEMALE), and it must be not NULL.

e. **CITY TABLE**

This table stores data about gender. CITY table has the following fields:

- **CITY_ID:** CITY Id, the data type is NUMBER, this field is a primary key on the table and it is a sequence that increases field's value when a new city is added.

- **STATE:** the data type is TEXT, this field stores the state name that has the city, and it must be not NULL.
- **CITY:** the data type is TEXT, this field stores the name of city, and it must be not NULL.

A.1.4 CLASSIFICATION Database

a. GENERATED TABLES DUE TO FRAGMENTATION PROCESS

- **TIME1 TABLE:** This table contains many fields that collected from TRANSFORM TEMP STORAGE database table, but TIME1 table has data about the tests that performed before 1970.
- **TIME2 TABLE:** This table contains many fields that collected from TRANSFORM TEMP STORAGE database table, but TIME2 table has date about the tests that performed between years 1970 and 1980.
- **TIME3 TABLE:** This table contains many fields that collected from TRANSFORM TEMP STORAGE database table, but TIME3 table has date about the tests that performed after 1980 until now.
- **TYPE1 table:** This table contains many fields that collected from TRANSFORM TEMP STORAGE database table, but TESTO_IMG field has NULL value in all records in this table.
- **TYPE2 table:** This table contains many fields that collected from TRANSFORM TEMP STORAGE database table, but TESTO_VIEDO field has NULL value in all records in this table.
- **TYPE3 table:** This table contains many fields that collected from TRANSFORM TEMP STORAGE database table, but all records in this table including TESTO_IMG and TESTO_VIEDO fields have NOT NULL value.
- **TYPE4 table:** This table contains many fields that collected from TRANSFORM TEMP STORAGE database table without any change on it.

b. **Fields of each of: TIME1, TIME2, TIME3, TYPE1, TYPE2, TYPE3 and TYPE4 tables**

- **TESTO_RESULT:** Test Operation Result, the data type is NUMBER, the data type is TEXT, this field views patient's case in detail, and it must be not NULL.
- **TESTO_DATE:** Test Operation Date, the data type is TEXT, this field store date of test performed, and it must be not NULL.
- **TESTO_IMG:** Test Operation Image, the data type is TEXT, this field store URL of image that belongs to patient's test, like X-ray image.
- **TESTO_VIDEO:** Test Operation Video, the data type is TEXT, this field store URL of video that belongs to patient's test, like ultrasound video.
- **DISEASE_ID:** Disease Id, the data type is NUMBER, the data type is NUMBER, this field is a foreign key belong to DISEASE table that specify the patient's disease, and it must be not NULL.
- **MEDICINE_ID:** Medicine Id, the data type is NUMBER, this field is a foreign key belong to MEDICINES table that specify the medicine that treat disease, and it must be not NULL.
- **GENDER_ID:** Gender Id, the data type is NUMBER, this field is a foreign key belong to GENDER table that specify the patient's sex, and it must be not NULL.
- **P_AGE:** Patient AGE, the data type is TEXT, this field store patient's age, and it must be not NULL.
- **CITY_ID:** City Id, the data type is NUMBER, this field is a foreign key belong to CITY table that specify the patient's STATE and CITY, and it must be not NULL.

c. **DISEASES TABLE**

This table stores data about Diseases. This table has the following fields:

- **DISEASE_ID:** Disease Id, the data type is NUMBER, this field is a primary key on the table and it is a sequence that increases field's value when a new disease is added.
- **DISEASE_TITLE:** Disease Title, the data type is TEXT, this field stores disease's title, and it must be not NULL.

- **DISEASE_TYPE:** Disease Type, the data type is TEXT, this field indicates disease's type, and it must be not NULL.

d. MEDICINES TABLE

This table stores data about Medicines each medicine belong to disease.

MEDICINES table has the following fields:

- **MEDICINE_ID:** Medicine Id, the data type is NUMBER, this field is a primary key on the table and it is a sequence that increases field value when a new medicine is added.
- **MEDICINE_DESC:** Medicine description, the data type is TEXT, this field stores description about medicine, and it must be not NULL.
- **DISEAS_ID:** Disease Id, this field is a foreign key belong to DISEASE table that specify the patient's disease, and it must be not NULL.

e. GENDER TABLE

This table stores data about gender. GENDER table has the following fields:

- **GENDER_ID:** Gender Id, the data type is NUMBER, this field is a primary key on the table and it is a sequence that increases field's value when a new data is added.
- **GENDER_DESC:** Gender description, the data type is TEXT, this field stores description about gender (such as MALE, FEMAIL), and it must be not NULL.

f. CITY TABLE

This table stores data about gender. CITY table has the following fields:

- **CITY_ID:** CITY Id, the data type is NUMBER, this field is a primary key on the table and it is a sequence that increases field value when a new city is added.
- **STATE:** the data type is TEXT, this field stores the state name that has the city, and it must be not NULL.

- **CITY:** the data type is TEXT, this field stores the name of city, and it must be not NULL.

A.1.5 LOAD Database

The LOAD database is the final storage that consists of fact and dimension tables (look up tables), which are produced after load process was done on CLASSIFICATION database's tables. Load database includes the same tables of CLASSIFICATION database in addition to (optional) tables from CLINIC database.

A.2 Details of the Class Diagrams

A.2.1 Extract-Data Class

a. Class attributes

- **port:** String variable that stores database port.
- **host:** String variable that stores database host name.
- **user:** String variable that stores database user account.
- **password:** String variable that stores database user account password.
- **db:** String variable that stores database name.
- **url:** String variable that concatenates database url from the previous variables.
- **conn:** Object variable that stores a connection (session) with a specific database.

b. Class methods:

- **jdbcConnect():** This method can initialize the database connection.
- **extractData(sql : String):** This method can extract data form multiple data sources and stores extracted data into temp storage.

- **jdbcConClose():** This method can terminate the database connection.

A.2.2 Transform-Data Class

a. Class attributes:

- **port:** String variable that stores database port.
- **host:** String variable that stores database host name.
- **user:** String variable that stores database user account.
- **password:** String variable that stores database user account password.
- **db:** String variable that stores database's name.
- **url:** String variable that concatenates database url from the previous variables.
- **conn:** Object variable that stores a connection (session) with a specific database.

b. Class methods

- **jdbcConnect():** This method can initialize the database connection.
- **jdbcMultipleRowQuery(sqlQuery : String):** This method returns multiple rows from database according to the sqlQuery parameter that is passed SQL statement.
- **getAge(dob : String):** This method calculates age from data of birth and returns the age.
- **transformData(sqlInsert : String):** This method can transform data into specific form and store transformed data into temp storage.
- **jdbcConClose():** This method can terminate the database connection.

A.2.3 Classification-Data Class

a. Class attributes:

- **port:** String variable that stores database port.
- **host:** String variable that stores database host name.
- **user:** String variable that stores database user account.
- **password:** String variable that stores database user account password.
- **db:** String variable that stores database's name.
- **url:** String variable that concatenates database url from the previous variables.
- **conn:** Object variable that stores a connection (session) with a specific database.

b. Class methods:

- **jdbcConnect():** This method can initialize the database connection.
- **jdbcMultipleRowQuery(sqlQuery : String) :** This method returns multiple rows from database according to the sqlQuery parameter that is passed SQL statement.
- **getYear(d : String):** This method returns the year from date to use it in classification process.
- **classificationData(sqlInsert : String):** This method can classify data into related groups and create new table with classified data.
- **jdbcConClose():** This method can terminate the database connection.

A.2.4 Load-Data Class

a. Class attributes:

- **port:** String variable that stores database port.

- **host:** String variable that stores database host name.
- **user:** String variable that stores database user account.
- **password:** String variable that stores database user account password.
- **db:** String variable that stores database name.
- **url:** String variable that concatenates database url from the previous variables.
- **conn:** Object variable that stores a connection (session) with a specific database.

b. Class methods:

- **jdbcConnect():** This method can initialize the database connection.
- **loadData(sql : String) :** This method can load classified tables (fact tables) and look up tables (dimensional tables) into database.
- **jdbcConClose():** This method can terminate the database connection.

Table A.1: The basic structure of a WSDL document supposed to be followed to implement ETL Web services

```

<?xml version="1.0" encoding="UTF-8"?>
<definitions name="myService"
targetNamespace= "http://etl.cas.uum.edu.my/wsdl/myService"
xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:tns=" http://etl.cas.uum.edu.my/wsdl/myService "
xmlns:ns="http://xml.uum.edu.my/schema/myService"
xmlns:plnk="http://docs.oasis-open.org/wsbpel/2.0/plnktype">
<types>
</types>
<message name="myMessageRequest">
</message>
<message name="myMessageReply">
</message>
<portType name="myPortType">
<wsdl:operation name="myOperation">
<wsdl:input name="input1" message="tns:myMessageRequest"/>
<wsdl:output name="output1" message="tns:myMessageReply"/>
</wsdl:operation>
</portType>
<binding name="myBinding" type="tns:myPortType">
</binding>
<service name="myService">
<wsdl:port name="myPort" binding="tns:myBinding">
</wsdl:port>
</service>
</definitions>

```

Table A.2: A simple XML file that could be followed as an example the ETL framework

```

<?xml version="1.0"?>
<ETL>
<name>classified-fragmentation</name>
</ETL>

```

Table A.3: A simple XML file for the purpose of standardizing the ETL framework

```

<?xml version="1.0"?>
<ETL>
<name>classified-fragmentation</name>
<datetime>2010</datetime>
</ETL>

```

Table A.4: etl.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://xml.uum.org/schema/etl"
xmlns:tns="http://xml.uum.org/schema/etl"
<xsd:element name="etl">
<xsd:complexType>
<xsd:sequence>
<xsd:element name="name" type="xsd:string"/>
<xsd:element name="datetime" type="xsd:date"/>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
</xsd:schema>

```

Table A.5: etl.xml

```

<?xml version="1.0" encoding="UTF-8"?>
<etl
xmlns="http://xml.uum.org/schema/etl"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://xml.uum.org/schema/etl/etl.xsd">
<name>classified-fragmentation</name>
<datetime>2010</datetime>
</etl>

```

[illegible]

Figure A.1: JSP file as the presentation layer of the Extract component


```

File Edit View Insert Format Window Help (or) Window: 200x100
Classification.jsp
<!--
Classification.jsp
-->
<html>
<head>
<title>Classification.jsp</title>
</head>
<body>
<table border="1">
<tr>
<td>Classification.jsp</td>
</tr>
</table>
</body>
</html>
</pre>

```

Figure A 3 (a): JSP file as the presentation layer of the Classified-fragmentation component

[illegible]

Figure A.3 (b): JSP file as the presentation layer of the Classified-fragmentation component


```

5
6 package awad.phd.prototype;
7
8 import javax.jws.WebMethod;
9 import javax.jws.WebService;
10 import javax.ejb.Stateless;
11 import javax.jws.WebParam;
12
13
14
15
16 @WebService()
17 @Stateless()
18 public class TransformWS {
19
20
21
22     @WebMethod(operationName = "transform")
23     public String TransformOperation(@WebParam(name = "input") String input) {
24         String transformParameter = input;
25
26         return transformParameter.toUpperCase();
27     }
28 }
29
30
31
32

```

Figure A.6: A Web service in a distributed architecture that wraps the business logic of the Transformation Component

```

6 package awad.phd.prototype;
7
8 import javax.jws.WebMethod;
9 import javax.jws.WebService;
10 import javax.ejb.Stateless;
11 import javax.jws.WebParam;
12
13
14
15 @WebService()
16 @Stateless()
17 public class ClassifyWS {
18
19
20
21
22     @WebMethod(operationName = "classify")
23     public String ClassifyOperation(@WebParam(name = "input") String input) {
24         String classifyParameter = input;
25
26         return classifyParameter.toLowerCase();
27     }
28 }
29
30
31
32

```

Figure A.7: A Web service in a distributed architecture that wraps the business logic of the Classified-Fragmentation Component

```

6  package awad.phd.prototype;
7
8  import javax.jws.WebMethod;
9  import javax.jws.WebService;
10 import javax.ejb.Stateless;
11 import javax.jws.WebParam;
12
13
14
15
16
17 @WebService()
18 @Stateless()
19 public class LoadWS {
20
21
22
23     @WebMethod(operationName = "LoadOperation")
24     public String LoadOperation(@WebParam(name = "loadParameter")
25     String loadParameter) {
26
27         return "LoadOperation: " + loadParameter + " + ";
28
29     }
30
31
32 }

```

Figure A.8: A Web service in a distributed architecture that wraps the business logic of the Loading Component

A.3 EBNF definitions for the meta-model of the new ETL Framework

```
ETL ::= Extract_Service ,
      Transform_Service,
      Load_Service,
      [Extended_Service], Orchestration_Point,
      ETL_Service_Bus;

Extract_Service ::= Extraction_Queries, (Distributed_Component_Container |
      User_Distribution_Settings);

Transform_Service ::= Transformation_Queries,
      (Distributed_Component_Container |
      User_Distribution_Settings);

Load_Service ::= Loading_Queries, (Distributed_Component_Container |
      User_Distribution_Settings);

Extended_Service ::= [Query], (Distributed_Component_Container |
```

```

User_Distribution_Setti
ngs);

Orchestration_Point ::= Orchestration_Specifica
tions,{Orchestration_Sp
ecifications},
XSD_WSDL,
{XSD_WSDL},{Distributed
_Component_Container |
User_Distribution_Setti
ngs);

ETL_Service_Bus ::= {ETL_Service_Provider_B
us |
ETL_Service_Requester_B
us}, ETL_MessageRoute,
Message_Model ,
{Message_Model},
Instructions_Flow,
Communication_Protoco,
{Communication_Protocol
}, Orchestration_Point;

Extraction_Queries ::= extract_from_dataSource
,
update_to_DataStagingAr
ea;

Transformation_Queries ::= extract_from_DataStagin
gArea,

```

```

update_to_DataStagingArea;

Loading_Queries ::= extract_from_DataStagingArea,
                    update_to_finalDwRepository,
                    delete_temp_DSA_data;

extract_from_dataSource ::= select_clause
                             from_clause
                             [where_clause]
                             [groupby_clause]
                             [having_clause]
                             [orderby_clause];

update_to_DataStagingArea ::= update_clause
                             [where_clause];

extract_from_DataStagingArea ::= select_clause
                                 from_clause
                                 [where_clause]
                                 [having_clause]
                                 [orderby_clause];

update_to_finalDwRepository ::= update_clause
                               [where_clause];

delete_temp_DSA_data ::= delete_clause
                        [where_clause];

```

```

ETL_Service_Provider_Bus ::= (Extract_Service |
                               Transform_Service |
                               Load_Service |
                               [Extended_Service] |
                               {Extended_Service});

ETL_Service_Requester_Bus ::= ETL_Client_Web_Servic,
                               {ETL_Client_Web_Service
                               };

ETL_MessageRouter          ::= [Service_Engine],
                               {ETL_Client_Request ,
                               Input_Extraction,
                               Response_Redirect};

Communication_Protocol     ::= (SOAP | HTTP),
                               (Request_Response |
                               Publish_Subscribe |
                               Synchronous_Asynchronou
                               s);

Instructions_Flow          ::= Mediation_Flows__Interfa
                               ces;

select_clause              ::= SELECT [DISTINCT]
                               select_expression {,
                               select_expression}*;

update_clause              ::= UPDATE

```

		abstract_schema_name
		[[AS]
		identification_variable
] SET update_item {,
		update_item}*;
delete_clause	::=	DELETE FROM
		abstract_schema_name
		[[AS]
		identification_variabl]
		;
from_clause	::=	FROM
		dataSource_schema_table
		{,{dataSource_schema_ta
		ble
		collection_member_decla
		ration}}*;
dataSource_schema_table	::=	range_variable_declarat
		ion { join}*;
range_variable_declaration	::=	abstract_schema_name
		[AS]
		identification_variable
		;
join	::=	join_spec
		join_association_path_e
		xpression [AS]

```

                                identification_variable
                                ;

association_path_expression ::= collection_valued_path_
                                expression |
                                single_valued_associati
                                on_path_expression;

join_spec                      ::= [LEFT] [OUTER | INNER]
                                JOIN;

join_association_path_express join_collection_valued_
ion                             ::= path_expression |
                                join_single_valued_asso
                                ciation_path_expression
                                ;

join_collection_valued_path_e identification_variable
xpression                      ::= .collection_valued_asso
                                ciation_field;

join_single_valued_associatio identification_variable
n_path_expression             ::= .single_valued_associat
                                ion_field;

collection_member_declaration::= IN
                                (collection_valued_path
                                _expression) [AS]
                                identification_variable
                                ;

```

```

single_valued_path_expression ::= state_field_path_expression |
                                single_valued_association_path_expression;

state_field_path_expression ::= {identification_variable |
                                single_valued_association_path_expression}.state_field;

single_valued_association_path_expression ::= identification_variable
                                                .{single_valued_association_field.*
                                                single_valued_association_field;

collection_valued_path_expression ::= identification_variable
                                        .{single_valued_association_field.*
                                        collection_valued_association_field;

state_field ::= {embedded_class_state_field.*simple_state_field;

update_item ::= [identification_variable.] {state_field |

```

		single_valued_associati on_field} = new_value;
new_value	::=	simple_arithmetic_expre ssion string_primary datetime_primary boolean_primary enum_primary simple_entity_expressio n NULL;
select_expression	::=	single_valued_path_expr ession aggregate_expression identification_variable OBJECT(identification_v ariable) constructor_expression;
constructor_expression	::=	NEW constructor_name(constr uctor_item {,constructor_item}*);
constructor_item	::=	single_valued_path_expr ession aggregate_expression;

aggregate_expression	::=	{AVG MAX MIN SUM} ([DISTINCT] state_field_path_expres sion) COUNT ([DISTINCT] identification_variable state_field_path_expres sion single_valued_associati on_path_expression);
where_clause	::=	WHERE conditional_expression;
groupby_clause	::=	GROUP BY groupby_item {, groupby_item}*;
groupby_item	::=	single_valued_path_expr ession;
having_clause	::=	HAVING conditional__expression;
orderby_clause	::=	ORDER BY orderby_item {, orderby_item}*;
orderby_item	::=	state_field_path_expres sion [ASC DESC]

conditional_expression	::=	conditional_term conditional_expression OR conditional_term;
conditional_term	::=	conditional_factor conditional_term AND conditional_factor;
conditional_factor	::=	[NOT] conditional_primary
conditional_primary	::=	simple_cond_expression (conditional_expression) ;
simple_cond_expression	::=	between_expression like_expression in_expression null_comparison_express ion empty_collection_compar ison_expression;
between_expression	::=	arithmetic_expression [NOT] BETWEEN arithmetic_expressionAN D arithmetic_expression string_expression [NOT]

```

BETWEEN
string_expression AND
string_expression |
datetime_expression
[NOT] BETWEEN
datetime_expression AND
datetime_expression;

in_expression ::= state_field_path_expres
                 sion [NOT] IN (in_item
                              {, in_item}*
                              | subquery);

in_item ::= literal |
          input_parameter;

like_expression ::= string_expression [NOT]
                   LIKE pattern_value
                   [ESCAPE
                   escape_character];

null_comparison_expression ::= {single_valued_path_exp
                                ression |
                                input_parameter} IS
                                [NOT] NULL;

```

A.4 Screenshots for the Design of Database Tables Used in the Case Studies

A.4.1 First Case Study (PEC)

Name	Type	Length	Decimals	Allow Null
	int	11	0	✓
dependent_age	varchar	40	0	✓
dependent_gender	varchar	40	0	✓

Figure A.9: employee_dependents Table

Name	Type	Length	Decimals	Allow Null
emp_id	int	11	0	✓
emp_hand_phone	int	40	0	✓
emp_phone	int	40	0	✓
emp_address	varchar	40	0	✓
emp_dependents_id	int	40	0	✓

Figure A.10: employee_incharge_details Table

Name	Type	Length	Decimals	Allow Null
elec_id	int	11	0	✓
turbine_id	int	11	0	✓
month_id	int	11	0	✓
period_id	int	11	0	✓

Figure A.11: elec_brkout Table

Name	Type	Length	Decimals	Allow Null
month_id	int	11	0	✓
month_desc	varchar	20	0	✓

Figure A.12: months Table

Name	Type	Length	Decimals	Allow Null
period_id	int	11	0	✓
start_time	longtext	0	0	✓
end_time	longtext	0	0	✓

Figure A.13: period Table ("time" data is stored as "longtext" for programming simplicity)

Name	Type	Length	Decimals	Allow Null
trans_type_id	int	11	0	✓
trans_type_descnption	varchar	40	0	✓

Figure A.14: transaction_types Table

Name	Type	Length	Decimals	Allow Null
trans_id	int	11	0	✓
trans_month	int	11	0	✓
trans_year	int	11	0	✓
trans_period_id	int	11	0	✓
trans_type_id	int	11	0	✓
emp_incharge_id	int	11	0	✓

Figure A.15: turbine_transaction_history Table

Name	Type	Length	Decimals	Allow Null
turbine_id	int	11	0	✓
turbine_desc	longtext	0	0	✓

Figure A.16: turbines Table

A.4.2 Second Case Study (LUCT)

Name	Type	Length	Decimals	Allow Null
attendance_id	int	11	0	✓
attendance_desc	varchar	20	0	✓

Figure A.17: attendance Table

Name	Type	Length	Decimals	Allow Null
final_id	int	11	0	✓
final_desc	varchar	20	0	✓

Figure A.18: finalexam Table

Name	Type	Length	Decimals	Allow Null
gender_id	int	11	0	✓
gender_desc	varchar	20	0	✓

Figure A.19: gender Table

Name	Type	Length	Decimals	Allow Null
mark_id	int	11	0	✓
mark_desc	varchar	20	0	✓

Figure A.20: marks Table

Name	Type	Length	Decimals	Allow Null
id	int	11	0	✓
gender_id	int	11	0	✓
mark_id	int	11	0	✓
status_id	int	11	0	✓
final_id	int	11	0	✓
attendance_id	int	11	0	✓
semester_details	varchar	40		✓

Figure A.21: student_performance Table

A.4.3 Third Case Study (PIT)

Name	Type	Length	Decimals	Allow Null
id	int	11	0	✓
performance_desc	varchar	20	0	✓
manager_comments	varchar	40	0	✓

Figure A.21: student_performance Table

Name	Type	Length	Decimals	Allow Null
tr_id	int	11	0	✓
tr_dep_name	varchar	40	0	✓
tr_dep_age	varchar	40	0	✓

Figure A.21: student_performance Table

Name	Type	Length	Decimals	Allow Null
trainer_id	int	11	0	✓
trainer_name	varchar	40	0	✓
course_trained	varchar	250	0	✓
trainer_age	varchar	40	0	✓
trainer_highest_degree	varchar	40	0	✓
trainer_industrial_certificate	varchar	40	0	✓
trainer_dependents_id	varchar	40	0	✓
trainer_hand_phone	varchar	40	0	✓
trainer_phone	varchar	40	0	✓
trainer_address	varchar	40	0	✓

Figure A.21: student_performance Table

Name	Type	Length	Decimals	Allow Null
id	int	11	0	✓
trainer_id	int	11	0	✓
performance_id	int	11	0	✓

Figure A.21: student_performance Table

A.5 Fragmented Vs. Un-Fragmented Performance Testeing

JMeter tool is provided by the Apache Software foundation. It is used for measuring the speed deference with and without the classified-fragmentation component. Apache JMeter is a java desktop application designed to test software functions and performance, especially for web applications. It tests both static and dynamic resources like files, databases, Java Servlets, Java Server Pages, and Web-Services. JMeter is able to simulate a heavy load on a server, a database, or an object.

A.5.1 Creating Test Plan

A basic Test Plane is created to test statistical report execution speed before and after involving the classified-fragmentation component of the SOA-based ETL prototype. To construct JMeter Test Plan, a Thread Group element is added by right clicking on the Test Plan to get Add menu, and then select Add → Thread Group as shown in Figure A.22.

Then, the default properties are customized as follows:

- Name: Is set to “Thread Group”, since this property provides more description about Thread Group.
- Number of Threads: “1”, “2” and “3” are set to this field periodically for every test. This property provides JMeter with the number of users who are expected to simulate (the number of users who send concurrent request to execute statistical reports). The software provides virtual users to instead of asking persons to execute concurrently.

- Ramp-up Period: Is set to “1”. This property feeds JMeter how long to delay between starting each user, For example, when you enter a Ramp-Up Period is 2 seconds; JMeter will finish starting all of users by the end of the 2 seconds.
- Loop Count: This property tells JMeter how many times to repeat test. When a loop count value is 2, then JMeter will run test twice. To run test Plan repeatedly, select the Forever. In this testing, it is set to 1.

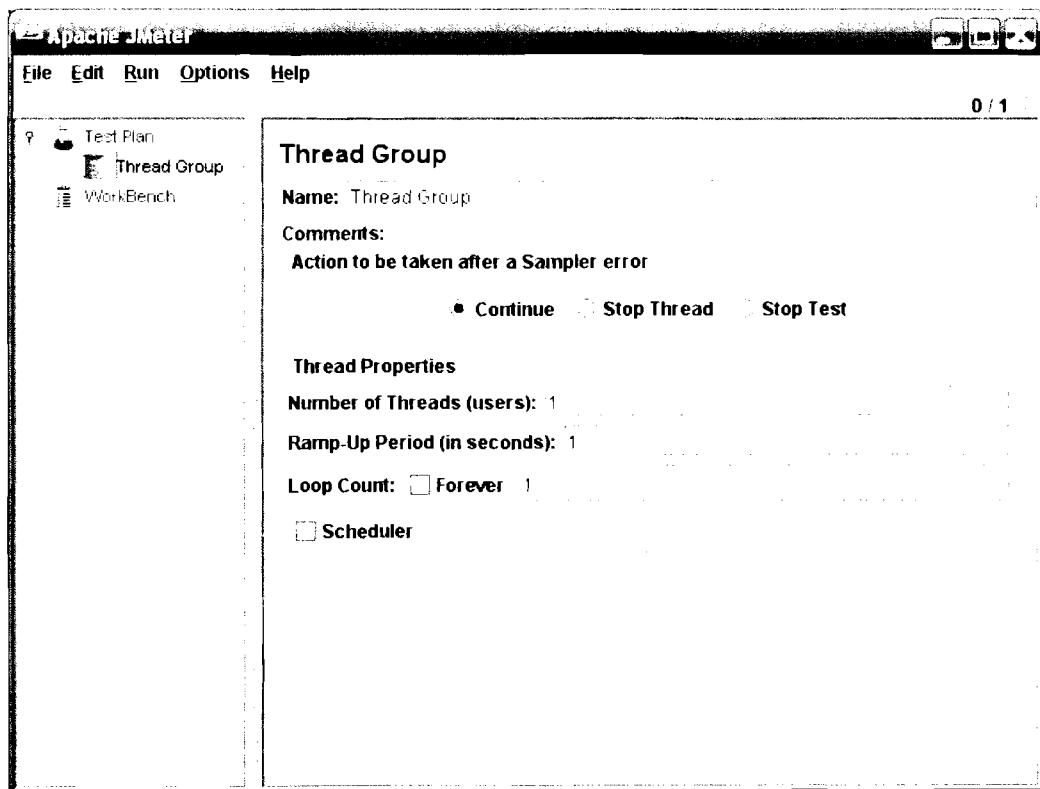


Figure A.22: Thread Group

A.5.2 Adding Default HTTP Request Properties

Settings are specified for requests. HTTP is selected because the statistical reports are hosted on a web server that needs HTTP protocol, which is Apache Tomcat Jakarta Web Server. This is done by right clicking on Thread Group element to get the Add menu and then select Add → Config Element → HTTP Request Defaults, as shown in Figure A.23.

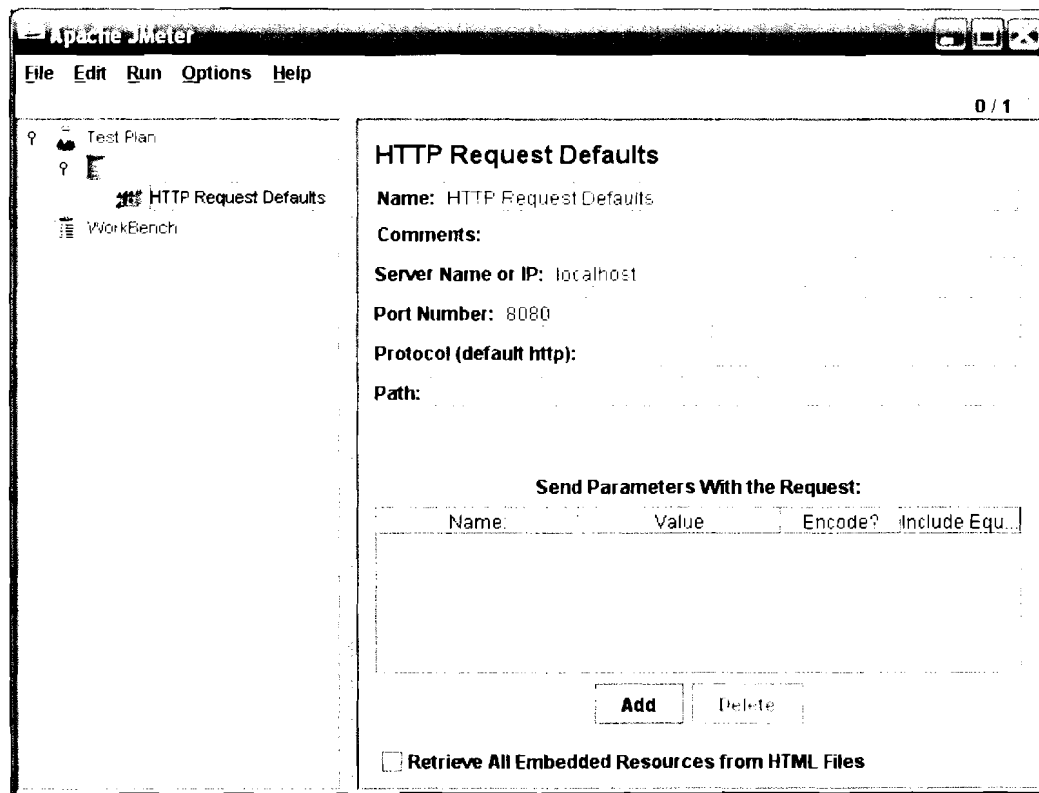


Figure A.23: Default HTTP Request Properties

After that, some of properties are customized to suit the required testing:

- Server Name or IP: This property point to the web server that host the web page of the two statistical reports (generated by fragmented and

unfragmented data) of this testing. For example, when the deployment is done at local host as this testing case, “localhost” is provided.

- Port Number: the default port value is 80, but in this testing the port number of Apache Tomcate Server is 8080, so, the port is set to “8080”.

A.5.3 Adding HTTP Requests

HTTP Request element is used to specify which web page will be test regarding its performance. This is done by right clicking on Thread Group to get Add menu then select Add→ Sampler→ HTTP Request. In this testing case, the name of the report web page is “index.jsp” and the path is identified to indicate the web page location on localhost web server, as shown in Figure A.24.

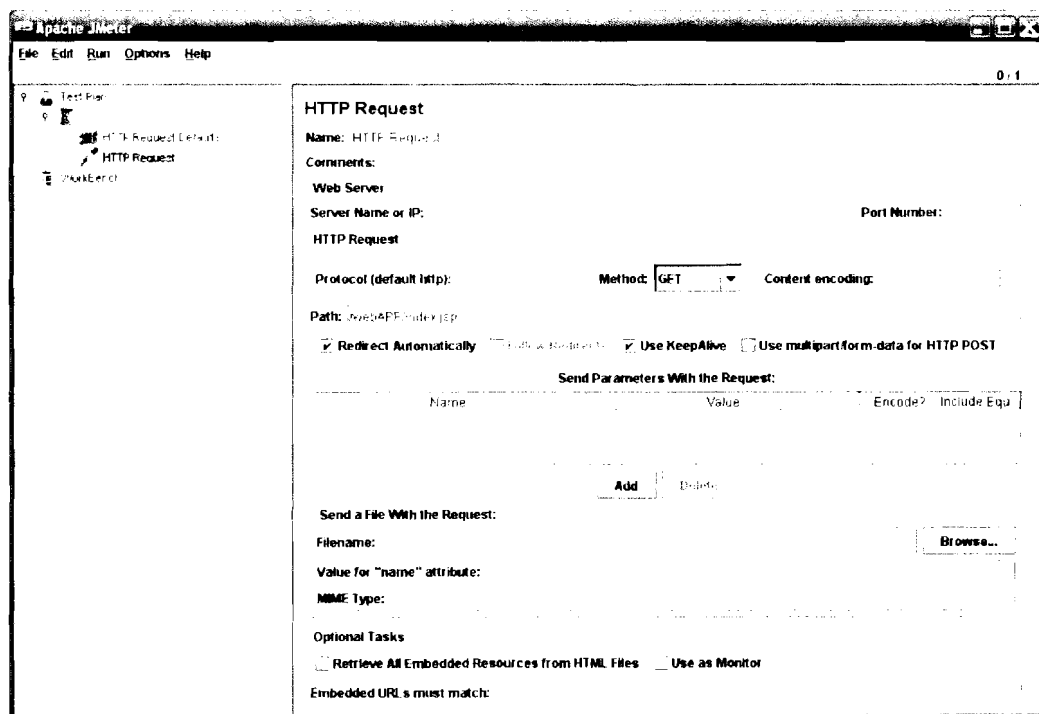


Figure A.24: HTTP Requests

A.5.4 Adding a Listener

This element stores all the HTTP requests results and view storing results. This is done by right clicking on Thread Group to get Add menu, then select Add → Listener → Generate results.

The testing tool results are influenced by a number of factors, such as fragmented data, non-fragmented data, number of records, the server or personal computer performance used to execute the testing and number of users. One report is generated twice (once using fragmented data and another using un-fragmented data) is tested. The report is shown in Figure A.25.

Disease/City	Alor Star	Baling	Changlun	Dong Dang	Dusun Kapas	Gua Tinggi	Jitra	Langkawi	Merbok	Tawar	Total
influenza	886	834	864	876	801	819	820	897	810	803	8410
heart disease	795	793	840	850	860	815	840	840	795	845	8273
headache	768	840	823	818	825	891	815	833	872	860	8345
hypertension	891	802	894	886	825	853	803	831	803	905	8493
disease1	1162	1053	1246	1124	1102	1162	1170	1145	1195	1126	11485
disease2	820	752	752	799	771	767	830	720	813	729	7753
disease3	736	750	790	805	792	812	785	782	775	747	7774
disease4	755	678	743	848	745	727	781	796	797	772	7642
disease5	731	764	708	852	781	813	778	751	811	724	7713
disease6	783	764	769	775	761	759	778	711	804	789	7693
Total	8327	8030	8429	8633	8263	8418	8400	8306	8475	8300	83581

Figure A.25: Report Generated Twice, Once Using Fragmented Data and the Other Using Un-Fragmented Data

A.5.5 Test Results

Both reports of fragmented and unfragmented data are re-generated 30 times using a volume of data 100,000 – 1,000,000 records as shown in Table A.6, and to make it

simpler for the reader to compare the results of fragmented and unfragmented data, only data from 100,000 to 300,000 is shown in Table 6.1 of section 6.7.2. These records (100,000 to 300,000) are shown in the first 9 rows of Table A.6. Concurrent virtual users who accessed the report concurrently are 1-3.

As a sample of showing the output of the JMeter regarding the time of execution, Figure A.26 shows a screenshot for testing result of 1,000,000 records with one user that matches record 28 of Table A.6. The left hand menu shows that the two reports to be tested are non-fragmented and fragmented web pages, while the “lable” and “Sample Time(ms)” fields in the middle of the same Figure show the results of time needed to execute both reports.

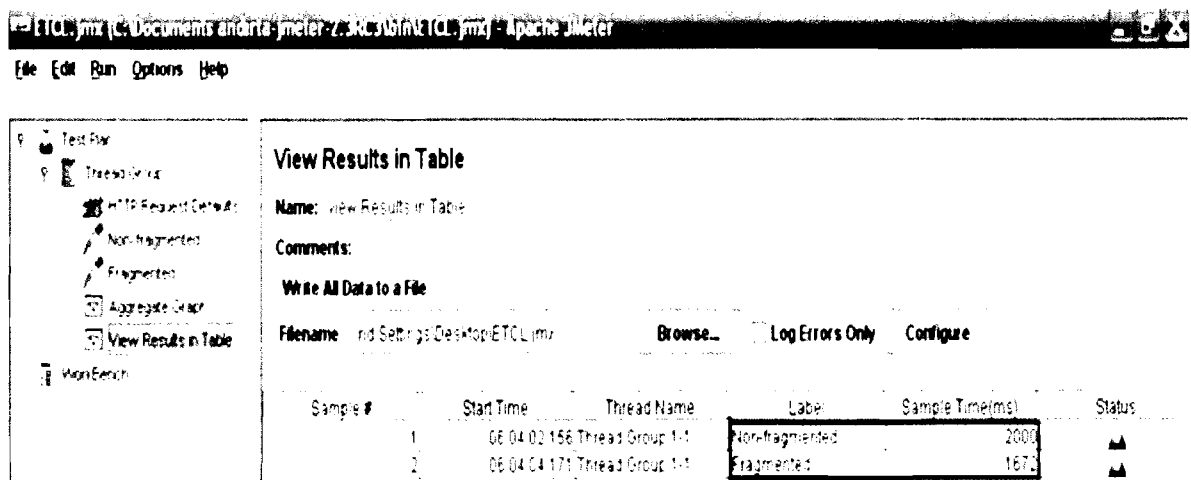


Figure A.26: Screenshot for Testing Result of 1,000,000 Records with One User

*Table A.6: Time Deference between Fragmented and Un-Fragmented Data for
Report Re-generation of 100,000 – 1,000,000 Records*

No.	Records	Users	Fragmented data (ms)	Non-fragmented data (ms)
1	100,000	1	188	297
2	100,000	2	192	207
3	100,000	3	255	364
4	200,000	1	359	515
5	200,000	2	369	520
6	200,000	3	369	643
7	300,000	1	516	688
8	300,000	2	531	719
9	300,000	3	625	815
10	400,000	1	703	891
11	400,000	2	713	958
12	400,000	3	851	1138
13	500,000	1	875	1094
14	500,000	2	901	1104
15	500,000	3	1060	1297
16	600,000	1	1063	1312
17	600,000	2	1109	1338
18	600,000	3	1307	1807
19	700,000	1	1234	1485
20	700,000	2	1239	1510
21	700,000	3	1278	1940
22	800,000	1	1391	1766
23	800,000	2	1500	1901
24	800,000	3	1531	2356
25	900,000	1	1500	1828
26	900,000	2	1526	1848
27	900,000	3	1797	2296
28	1,000,000	1	1672	2000
29	1,000,000	2	1703	2086
30	1,000,000	3	2034	2390

JMeter tool does not deal with many reporting softwares available in the market. Therefore, a special programming project is developed by the researcher for the purpose of creating the fragmented and unfragmented based reports as two web pages. These programming codes are hosted on Tomcate Jakarta server that is compatible with JMeter as both of them are produced by the same vendor “Apache Software foundation”. Table A.7 and Table A.8 Show the source code developed specially for creating the statistical report shown in Figure A.25. Table A.7 shows a reusable Java class that deals with the data warehouse repository and its stored procedure, while Table A.8 shows the JSP code used as a presentation layer for showing and formatting the report.

Table A.7: A reusable Java class that deals with the data warehouse repository and its stored procedure

```
package uum.awad.phd;

import java.io.*;
import java.sql.*;
import java.util.*;
import com.fdsapi.ResultSetConverter;
import com.jamonapi.utils.*;

public class BusinessBean {

    Connection connection = null;
    Statement statement;
    public int lenght2 = 1;
    public int length1 = 1;

    public BusinessBean() //throws Exception
    {

        try {
```

```

        String userName = "awad";
        String password = "";

        String dataSourceName = "phd_testing";
        String dbURL =
            "jdbc:odbc:" + dataSourceName;

        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

        connection =
            DriverManager.getConnection(dbURL,
                userName, password);
        statement =
            connection.createStatement();

    } catch (Exception e) {
    }

}

/* this method returns a 2-dimentional array
   contains all of the resultset items */
public String[][] getResultset() throws SQLException
{

    CallableStatement st =
        connection.prepareCall("{call
            statisticalReport(?,?)}",
            ResultSet.TYPE_SCROLL_SENSITIVE,
            ResultSet.CONCUR_READ_ONLY);
    st.setInt(1, 5000002);
    st.setInt(2, 4);
    st.execute();
    ResultSet rs = (ResultSet) st.executeQuery();
    ResultSetConverter rsC = new
        ResultSetConverter(rs);
    Object[][] rsArray = rsC.getResultSet();
    String[][] resultList = new
        String[rsArray.length][rsArray[0].length];
    for (int i = 0; i < rsArray.length; i++) {
        for (int j = 0; j < rsArray[0].length; j++)

```

```

        resultList[i][j] =
            rsArray[i][j].toString();
    }
}

return resultList;
}
/* Returns distinct values of the column in the
resultset which we wanna add it as the left
column in the cross tab report*/

public List getLeftColumn(String[][] s, int
leftColumnOrder) {
    ArrayList leftColumnArrayList = new ArrayList();
    for (int n = 0; n < s.length; n++) {
        if
        (leftColumnArrayList.contains(s[n][leftColumnOrd
er - 1])) {continue } else {

        leftColumnArrayList.add(s[n][leftColumnOrder -
1]);
        }
    }
    return leftColumnArrayList;
}

/*Returns distinct values of the column in the
resultset which we wanna add it as the titles
row in the cross tab report*/
public List getTitlesRow(String[][] s, int
titlesOrder) {
    ArrayList titlesArrayList = new ArrayList();
    for (int n = 0; n < s.length; n++) {
        if
        (titlesArrayList.contains(s[n][titlesOrder -
1])) {
            continue;
        } else {
            titlesArrayList.add(s[n][titlesOrder -
1]);
        }
    }
    return titlesArrayList;
}

```

```

/*gets the calculated results in a 2 dimensional array
of Longs*/
    public String[][] getCalculatedResults(String[][]
resultList, List leftColumnArrayList, List
titlesArrayList, int wantedFieldOrder, int
leftColumnOrder, int titlesOrder) {
        String[][] calcRs = new
String[leftColumnArrayList.size()][titlesArrayList.size(
)];
        Object[] leftColumnArrayList1 =
leftColumnArrayList.toArray();
        Object[] titlesArrayList1 =
titlesArrayList.toArray();
        int i = 0;

        for (i = 0; i < resultList.length; i++) {
            for (int j = 0; j < resultList[0].length;
j++) {
                for (int h = 0; h <
titlesArrayList1.length; h++) {
                    for (int v = 0; v <
leftColumnArrayList1.length; v++) {
                        if
(resultList[i][leftColumnOrder -
1].equals(leftColumnArrayList1[v].toString()) &&
resultList[i][titlesOrder -
1].equals(titlesArrayList1[h].toString())) {
                            calcRs[v][h] =
resultList[i][wantedFieldOrder - 1];
                        }
                    }
                }
            }

            return calcRs;
        }

        // close statements and terminate database
        connection

```

```

        protected void finalize() {
            // attempt to close database connection
            try {
                statement.close();
                connection.close();
            } // process SQLException on close operation
            catch (SQLException sqlException) {
                sqlException.printStackTrace();
            }
        }
    }
}

```

Table A.8: The JSP code used as a presentation layer for designing and formatting the report

```

<jsp:useBean id = "bB" scope = "page"
    class = "uum.awad.phd.BusinessBean" />

<%@ page contentType="text/html; charset=utf-8" language
= "java" %> < %@page import = "java.util.*" %>
<%@ page  import = "uum.awad.phd.*" %>

<html xmlns = "http://www.w3.org/1999/xhtml" >
    <head>
        < title > resultset values</
        title

        >
        <meta http-equiv="Content-Type" content = "text/html;
        charset=UTF-8" /> < /head> < body>

        <% // start scriptlet

```

```

        String[][] varList = bB.getResultset();
        List left = bB.getLeftColumn(varList, 1);
        List titles = bB.getTitlesRow(varList, 2);
        String[][] calcRs = bB.getCalculatedResults(varList,
        left, titles, 5, 1, 2);
        Object[] left1 = left.toArray();
        %>

        <center><Table border="1" cellspacing = "0"
        cellpadding = "0" id = "Table1" style = "border-
        collapse: collapse; border-style: solid; border-width:
        1px" bordercolorlight = "#C0C0C0" bordercolordark =
        "#000000" >

        <%-- -- -- -- ---titles row ------%>
        <tr bgcolor="#4E4E4E" >
            <td> < center > <font
            color = "#FFFFFF" > <b>
            Disease</b></font></center></td>
        <%
            Iterator listIterator2 = titles.iterator();

            while ( listIterator2.hasNext() ) {
                String next2=(String)listIterator2.next();

                %>
                <td><font
                    color = "#FFFFFF" > <b> <%= next2 %> < /b> <
                /font> < /td> <%

                }
                %>
                <td><center><font color="#FFFFFF" > <b>
                Total</b>

                ></font></center></td>
            </tr>
            <%-- -----end of titles row------%>

        <%
        long vTotal=0;
        for

```

```

        (int k=0;
        k<calcRs.length ; k++ ){

            if(k%2 == 0) {
                out.print("<tr bgcolor=\"#F5F3F5\"><td>" +
                left1[k] + "</td>");
            } else {
                out.print("<tr><td>" + left1[k] + "</td>");

            }
            for (int l = 0; l < calcRs[k].length; l++) {
                %>
                <td><center><%
                if(
                    calcRs[k][l].equalsIgnoreCase("0")) {
                        out.print("");
                    } else {
                        out.print(calcRs[k][l]);
                    }
                %> < /center> < /td> < %
                vTotal += Long.parseLong(calcRs[k][l]);
            }
            %>
            <td><center><%
            if(
                vTotal == 0) {
                    out.print("");
                } else {
                    out.print(vTotal);
                }
            %> < /center> < /td> < %    vTotal = 0;
            %>
        </tr>

        <% }

        long hTotal = 0;
        long[] lastRow = new long[calcRs[0].length];
        for
            (int a=0;
            a<calcRs
                [0].length;a++){
        for(int b=0; b < calcRs.length; b++) {
            hTotal += Long.parseLong(calcRs[b][a]);
        }
    
```

```

        lastRow[a] = hTotal;
        hTotal = 0;

    }
    %
    >
<tr bgcolor="#4E4E4E" >
    <td> < center > <font
        color = "#FFFFFF" > <b>
        Total</
        b></font></center></td>
<%
long crossTotal=0;
    for
        (int c=0;
        c<lastRow.length ; c++ ){

    if(lastRow[c]==0) {
        out.print("<td><center>" + "" +
"</center></td>");
    } else {
        out.print("<td><center><font
color=#FFFFFF><b>" + lastRow[c] +
"</b></font></center></td>");
    }
        crossTotal += lastRow[c];
    }
    %>
    <td><center><font color="#FFFFFF" > <b> <%= crossTotal
%> < /b> < /font> < /center> < /td> < /tr> < /Table> <
/center> < /body> < /html>

```

Appendix B

Source Code of the Prototype

Table B.1: ExtractData.java

```
package data.etl.dw;

import java.sql.*;
import java.util.*;
import java.io.*;

public class ExtractData {

    private String host;
    private String port;
    private String user;
    private String password;
    private String db;
    private String url;
    private Connection conn;

    public ExtractData(String config) throws IOException {
        Properties myproperties = new Properties();

        FileInputStream in = new FileInputStream(config);
        myproperties.load(in);
        in.close();

        this.host = myproperties.getProperty("host");
        this.port = myproperties.getProperty("port");
        this.user = myproperties.getProperty("user");
        this.password = myproperties.getProperty("password");
        this.db = myproperties.getProperty("db");

        this.url = "jdbc:mysql://" + this.host + ":" + this.port + "/" + this.db;
    }

    public void jdbcConnect() {
        try {
            Class.forName("com.mysql.jdbc.Driver");
            this.conn = DriverManager.getConnection(this.url, this.user, this.password);
```

```

        } catch (SQLException ex) {
        } catch (ClassNotFoundException ex) {
        } catch (java.lang.Exception ex) {
        }
    }

    public void jdbcConClose() {
        try {
            this.conn.close();
        } catch (Exception ex) {
        }
    }

    public void extractData(String sql) {
        try {
            Statement extractStmt = this.conn.createStatement();
            extractStmt.executeUpdate(sql);
            extractStmt.close();
        } catch (SQLException ex) {
        }
    }
}

```

Table B.2: TransformData.java

```

package data.etl.dw;

import java.sql.*;
import java.util.*;
import java.io.*;

public class TransformData {

    private String host;
    private String port;
    private String user;
    private String password;
    private String db;
    private String url;
    private Connection conn;

```

```

public TransformData(String config) throws IOException {
    Properties myproperties = new Properties();

    FileInputStream in = new FileInputStream(config);
    myproperties.load(in);
    in.close();

    this.host = myproperties.getProperty("host");
    this.port = myproperties.getProperty("port");
    this.user = myproperties.getProperty("user");
    this.password = myproperties.getProperty("password");
    this.db = myproperties.getProperty("db");

    this.url = "jdbc:mysql://" + this.host + ":" + this.port + "/" + this.db;
}

public void jdbcConnect() {
    try {
        Class.forName("com.mysql.jdbc.Driver");
        this.conn = DriverManager.getConnection(this.url, this.user, this.password);

    } catch (SQLException ex) {
    } catch (ClassNotFoundException ex) {
    } catch (java.lang.Exception ex) {
    }
}

public void jdbcConClose() {
    try {
        this.conn.close();
    } catch (Exception ex) {
    }
}

public void transformData(String sqlInsert) throws SQLException {

    Statement insertStmt = this.conn.createStatement();
    insertStmt.executeUpdate(sqlInsert);
    insertStmt.close();
}

public Vector jdbcMultipleRowQuery(String sqlQuery) {
    Vector v = new Vector();
    try {
        Statement queryStmt = this.conn.createStatement();
        ResultSet rs = queryStmt.executeQuery(sqlQuery);
        int colCount = rs.getMetaData().getColumnCount();
        String s;

```

```

        while (rs.next()) {
            Vector vec = new Vector();
            for (int i = 0; i < colCount; i++) {
                s = rs.getString(i + 1);
                vec.addElement(s);
            }
            v.addElement(vec);
        }
        rs.close();
        queryStmt.close();

    } catch (SQLException ex) {
    }
    return v;
}

//Transform DOB to Age
public int getAge(String dob) {
    int day, month, year, ageYears, ageMonths, ageDays;

    String[] split = dob.split("-");

    day = Integer.parseInt(split[2]);
    month = Integer.parseInt(split[1]);
    year = Integer.parseInt(split[0]);

    Calendar cd = Calendar.getInstance();
    Calendar bd = new GregorianCalendar(year, month, day);
    ageYears = cd.get(Calendar.YEAR) - bd.get(Calendar.YEAR);
    if (cd.before(new GregorianCalendar(cd.get(Calendar.YEAR), month, day))) {
        ageYears--;
        ageMonths = (12 - (bd.get(Calendar.MONTH) + 1)) +
(bd.get(Calendar.MONTH));
        if (day > cd.get(Calendar.DAY_OF_MONTH)) {
            ageDays = day - cd.get(Calendar.DAY_OF_MONTH);
        } else if (day < cd.get(Calendar.DAY_OF_MONTH)) {
            ageDays = cd.get(Calendar.DAY_OF_MONTH) - day;
        } else {
            ageDays = 0;
        }
    } else if (cd.after(bd)) {
        ageMonths = (12 - (bd.get(Calendar.MONTH) + 1)) - 1;
        if (day > cd.get(Calendar.DAY_OF_MONTH)) {
            ageDays = day - cd.get(Calendar.DAY_OF_MONTH) - day;
        } else if (day < cd.get(Calendar.DAY_OF_MONTH)) {
            ageDays = cd.get(Calendar.DAY_OF_MONTH) - day;
        } else {
            ageDays = 0;
        }
    }
}

```

```

    } else {
        ageYears = cd.get(Calendar.YEAR) - bd.get(Calendar.YEAR);
        ageMonths = 0;
        ageDays = 0;
    }
    if (ageMonths > 6) {
        ageYears = ageYears + 1;
    }

    return ageYears;
}
}

```

Table B.3: ClassificationData.java

```

package data.etl.dw;

import java.sql.*;
import java.util.*;
import java.io.*;

public class ClassificationData {

    private String host;
    private String port;
    private String user;
    private String password;
    private String db;
    private String url;
    private Connection conn;

    public ClassificationData(String config) throws IOException {
        Properties myproperties = new Properties();

        FileInputStream in = new FileInputStream(config);
        myproperties.load(in);
        in.close();

        this.host = myproperties.getProperty("host");
        this.port = myproperties.getProperty("port");
        this.user = myproperties.getProperty("user");
        this.password = myproperties.getProperty("password");
        this.db = myproperties.getProperty("db");
    }
}

```

```

        this.url = "jdbc:mysql://" + this.host + ":" + this.port + "/" + this.db;
    }

    public void jdbcConnect() {
        try {
            Class.forName("com.mysql.jdbc.Driver");
            this.conn = DriverManager.getConnection(this.url, this.user, this.password);

        } catch (SQLException ex) {
        } catch (ClassNotFoundException ex) {
        } catch (java.lang.Exception ex) {
        }
    }

    public void jdbcConClose() {
        try {
            this.conn.close();
        } catch (Exception ex) {
        }
    }

    public void classificaionData(String sqlInsert) throws SQLException {

        Statement insertStmt = this.conn.createStatement();
        insertStmt.executeUpdate(sqlInsert);
        insertStmt.close();
    }

    public Vector jdbcMultipleRowQuery(String sqlQuery) {
        Vector v = new Vector();
        try {
            Statement queryStmt = this.conn.createStatement();
            ResultSet rs = queryStmt.executeQuery(sqlQuery);
            int colCount = rs.getMetaData().getColumnCount();
            String s;
            while (rs.next()) {
                Vector vec = new Vector();
                for (int i = 0; i < colCount; i++) {
                    s = rs.getString(i + 1);
                    vec.addElement(s);
                }
                v.addElement(vec);
            }
            rs.close();
            queryStmt.close();

        } catch (SQLException ex) {

```

```

    }
    return v;
}

public int getYear(String d) {
    int year;
    String[] split = d.split("-");

    year = Integer.parseInt(split[0]);
    return year;
}
}

```

Table B.4: LoadData.java

```

package data.etl.dw;

import java.sql.*;
import java.util.*;
import java.io.*;

public class LoadData {

    private String host;
    private String port;
    private String user;
    private String password;
    private String db;
    private String url;
    private Connection conn;

    public LoadData(String config) throws IOException {
        Properties myproperties = new Properties();

        FileInputStream in = new FileInputStream(config);
        myproperties.load(in);
        in.close();

        this.host = myproperties.getProperty("host");
        this.port = myproperties.getProperty("port");
        this.user = myproperties.getProperty("user");
        this.password = myproperties.getProperty("password");
        this.db = myproperties.getProperty("db");
    }
}

```

```

        this.url = "jdbc:mysql://" + this.host + ":" + this.port + "/" + this.db;
    }

    public void jdbcConnect() {
        try {
            Class.forName("com.mysql.jdbc.Driver");
            this.conn = DriverManager.getConnection(this.url, this.user, this.password);

        } catch (SQLException ex) {
        } catch (ClassNotFoundException ex) {
        } catch (java.lang.Exception ex) {
        }
    }

    public void jdbcConClose() {
        try {
            this.conn.close();
        } catch (Exception ex) {
        }
    }

    public void loadData(String sql) {
        try {
            Statement extractStmt = this.conn.createStatement();
            extractStmt.executeUpdate(sql);
            extractStmt.close();
        } catch (SQLException ex) {
        }
    }
}

```

Table B.5: WSDL Code for Extract Web Service

```

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!-- Generated by JAX-WS RI at http://jax-ws.dev.java.net. RI's version is JAX-
WS RI 2.1.3.1-hudson-749-SNAPSHOT. -->
<definitions targetNamespace="http://prototype.phd.awad/"
name="ExtractWSService" xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:tns="http://prototype.phd.awad/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"

```

```

xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-
utility-1.0.xsd">
  <ns1:Policy wsu:Id="ExtractWSPortBinding_ExtractOperation_WSAT_Policy"
xmlns:ns1="http://www.w3.org/ns/ws-policy">
    <ns1:ExactlyOne>
      <ns1:All>
        <ns2:ATAlwaysCapability
xmlns:ns2="http://schemas.xmlsoap.org/ws/2004/10/wsat"/>
        <ns3:ATAssertion ns1:Optional="true" ns4:Optional="true"
xmlns:ns4="http://schemas.xmlsoap.org/ws/2002/12/policy"
xmlns:ns3="http://schemas.xmlsoap.org/ws/2004/10/wsat"/>
      </ns1:All>
    </ns1:ExactlyOne>
  </ns1:Policy>
  <types>
    <xsd:schema>
      <xsd:import namespace="http://prototype.phd.awad/"
schemaLocation="ExtractWSService_schema1.xsd"/>
    </xsd:schema>
  </types>
  <message name="ExtractOperation">
    <part name="parameters" element="tns:ExtractOperation"/>
  </message>
  <message name="ExtractOperationResponse">
    <part name="parameters" element="tns:ExtractOperationResponse"/>
  </message>
  <portType name="ExtractWS">
    <operation name="ExtractOperation">
      <input message="tns:ExtractOperation"/>
      <output message="tns:ExtractOperationResponse"/>
    </operation>
  </portType>
  <binding name="ExtractWSPortBinding" type="tns:ExtractWS">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http"
style="document"/>
    <operation name="ExtractOperation">
      <ns5:PolicyReference
URI="#ExtractWSPortBinding_ExtractOperation_WSAT_Policy"
xmlns:ns5="http://www.w3.org/ns/ws-policy"/>
      <soap:operation soapAction=""/>
      <input>
        <soap:body use="literal"/>
      </input>
      <output>
        <soap:body use="literal"/>
      </output>
    </operation>
  </binding>
</service name="ExtractWSService">

```

```

    <port name="ExtractWSPort" binding="tns:ExtractWSPortBinding">
      <soap:address location="REPLACE_WITH_ACTUAL_URL"/>
    </port>
  </service>
</definitions>

```

Table B.6. XML Schema for Extract Web Service

```

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xs:schema version="1.0" targetNamespace="http://prototype.phd.awad/"
  xmlns:tns="http://prototype.phd.awad/"
  xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="ExtractOperation" type="tns:ExtractOperation"/>

  <xs:element name="ExtractOperationResponse"
    type="tns:ExtractOperationResponse"/>

  <xs:complexType name="ExtractOperation">
    <xs:sequence>
      <xs:element name="extractParameter" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="ExtractOperationResponse">
    <xs:sequence>
      <xs:element name="return" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>

```

Table B.7: WSDL Code for Transform Web Service

```

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!-- Generated by JAX-WS RI at http://jax-ws.dev.java.net. RI's version is JAX-
WS RI 2.1.3.1-hudson-749-SNAPSHOT. -->

```

```

<definitions targetNamespace="http://prototype.phd.awad/"
name="TransformWSService" xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:tns="http://prototype.phd.awad/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
wssecurity-utility-1.0.xsd">
  <ns1:Policy
wsu:Id="TransformWSPortBinding_TransformOperation_WSAT_Policy"
xmlns:ns1="http://www.w3.org/ns/ws-policy">
    <ns1:ExactlyOne>
      <ns1:All>
        <ns2:ATAIwaysCapability
xmlns:ns2="http://schemas.xmlsoap.org/ws/2004/10/wsat"/>
        <ns3:ATAssertion ns1:Optional="true" ns4:Optional="true"
xmlns:ns4="http://schemas.xmlsoap.org/ws/2002/12/policy"
xmlns:ns3="http://schemas.xmlsoap.org/ws/2004/10/wsat"/>
      </ns1:All>
    </ns1:ExactlyOne>
  </ns1:Policy>
  <types>
    <xsd:schema>
      <xsd:import namespace="http://prototype.phd.awad/"
schemaLocation="TransformWSService_schema1.xsd"/>
    </xsd:schema>
  </types>
  <message name="TransformOperation">
    <part name="parameters" element="tns:TransformOperation"/>
  </message>
  <message name="TransformOperationResponse">
    <part name="parameters" element="tns:TransformOperationResponse"/>
  </message>
  <portType name="TransformWS">
    <operation name="TransformOperation">
      <input message="tns:TransformOperation"/>
      <output message="tns:TransformOperationResponse"/>
    </operation>
  </portType>
  <binding name="TransformWSPortBinding" type="tns:TransformWS">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http"
style="document"/>
    <operation name="TransformOperation">
      <ns5:PolicyReference
URI="#TransformWSPortBinding_TransformOperation_WSAT_Policy"
xmlns:ns5="http://www.w3.org/ns/ws-policy"/>
      <soap:operation soapAction=""/>
      <input>
        <soap:body use="literal"/>
      </input>
    </operation>
  </binding>

```

```

        <output>
          <soap:body use="literal"/>
        </output>
      </operation>
    </binding>
    <service name="TransformWSService">
      <port name="TransformWSPort" binding="tns:TransformWSPortBinding">
        <soap:address location="REPLACE_WITH_ACTUAL_URL"/>
      </port>
    </service>
  </definitions>

```

Table B.8: XML Schema for Transform Web Service

```

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xs:schema version="1.0" targetNamespace="http://prototype.phd.awad/"
  xmlns:tns="http://prototype.phd.awad/"
  xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="TransformOperation" type="tns:TransformOperation"/>

  <xs:element name="TransformOperationResponse"
    type="tns:TransformOperationResponse"/>

  <xs:complexType name="TransformOperation">
    <xs:sequence>
      <xs:element name="transformParameter" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="TransformOperationResponse">
    <xs:sequence>
      <xs:element name="return" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>

```

Table B.9: WSDL Code for Classify Web Service

```

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!-- Generated by JAX-WS RI at http://jax-ws.dev.java.net. RI's version is JAX-WS
RI 2.1.3.1-hudson-749-SNAPSHOT. -->
<definitions targetNamespace="http://prototype.phd.awad/"
name="ClassifyWSService" xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:tns="http://prototype.phd.awad/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:wsu="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
  <ns1:Policy wsu:Id="ClassifyWSPortBinding_ClassifyOperation_WSAT_Policy"
xmlns:ns1="http://www.w3.org/ns/ws-policy">
    <ns1:ExactlyOne>
      <ns1:All>
        <ns2:ATAlwaysCapability
xmlns:ns2="http://schemas.xmlsoap.org/ws/2004/10/wsat"/>
        <ns3:ATAssertion ns1:Optional="true" ns4:Optional="true"
xmlns:ns4="http://schemas.xmlsoap.org/ws/2002/12/policy"
xmlns:ns3="http://schemas.xmlsoap.org/ws/2004/10/wsat"/>
      </ns1:All>
    </ns1:ExactlyOne>
  </ns1:Policy>
  <types>
    <xsd:schema>
      <xsd:import namespace="http://prototype.phd.awad/"
schemaLocation="ClassifyWSService_schema1.xsd"/>
    </xsd:schema>
  </types>
  <message name="ClassifyOperation">
    <part name="parameters" element="tns:ClassifyOperation"/>
  </message>
  <message name="ClassifyOperationResponse">
    <part name="parameters" element="tns:ClassifyOperationResponse"/>
  </message>
  <portType name="ClassifyWS">
    <operation name="ClassifyOperation">
      <input message="tns:ClassifyOperation"/>
      <output message="tns:ClassifyOperationResponse"/>
    </operation>
  </portType>
  <binding name="ClassifyWSPortBinding" type="tns:ClassifyWS">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http"
style="document"/>
    <operation name="ClassifyOperation">
      <ns5:PolicyReference
URI="#ClassifyWSPortBinding_ClassifyOperation_WSAT_Policy"
xmlns:ns5="http://www.w3.org/ns/ws-policy"/>
      <soap:operation soapAction=""/>
      <input>
        <soap:body use="literal"/>
      </input>
    </operation>
  </binding>

```

```

    </input>
    <output>
      <soap:body use="literal"/>
    </output>
  </operation>
</binding>
<service name="ClassifyWSService">
  <port name="ClassifyWSPort" binding="tns:ClassifyWSPortBinding">
    <soap:address location="REPLACE_WITH_ACTUAL_URL"/>
  </port>
</service>
</definitions>

```

Table B.10: XML Schema for Classify Web Service

```

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xs:schema version="1.0" targetNamespace="http://prototype.phd.awad/"
  xmlns:tns="http://prototype.phd.awad/"
  xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="ClassifyOperation" type="tns:ClassifyOperation"/>

  <xs:element name="ClassifyOperationResponse"
    type="tns:ClassifyOperationResponse"/>

  <xs:complexType name="ClassifyOperation">
    <xs:sequence>
      <xs:element name="classifyParameter" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="ClassifyOperationResponse">
    <xs:sequence>
      <xs:element name="return" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>

```

Table B.11: WSDL Code for Load Web Service

```

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!-- Generated by JAX-WS RI at http://jax-ws.dev.java.net. RI's version is JAX-WS
RI 2.1.3.1-hudson-749-SNAPSHOT. -->
<definitions targetNamespace="http://prototype.phd.awad/"
name="LoadWSService" xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:tns="http://prototype.phd.awad/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:wsu="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
  <ns1:Policy wsu:Id="LoadWSPortBinding_LoadOperation_WSAT_Policy"
xmlns:ns1="http://www.w3.org/ns/ws-policy">
    <ns1:ExactlyOne>
      <ns1:All>
        <ns2:ATAlwaysCapability
xmlns:ns2="http://schemas.xmlsoap.org/ws/2004/10/wsdl/">
          <ns3:ATAssertion ns1:Optional="true" ns4:Optional="true"
xmlns:ns4="http://schemas.xmlsoap.org/ws/2002/12/policy"
xmlns:ns3="http://schemas.xmlsoap.org/ws/2004/10/wsdl/">
            </ns3:ATAssertion>
          </ns2:ATAlwaysCapability>
        </ns1:All>
      </ns1:ExactlyOne>
    </ns1:Policy>
  </types>
  <xsd:schema>
    <xsd:import namespace="http://prototype.phd.awad/"
schemaLocation="LoadWSService_schema1.xsd"/>
  </xsd:schema>
</types>
<message name="LoadOperation">
  <part name="parameters" element="tns:LoadOperation"/>
</message>
<message name="LoadOperationResponse">
  <part name="parameters" element="tns:LoadOperationResponse"/>
</message>
<portType name="LoadWS">
  <operation name="LoadOperation">
    <input message="tns:LoadOperation"/>
    <output message="tns:LoadOperationResponse"/>
  </operation>
</portType>
<binding name="LoadWSPortBinding" type="tns:LoadWS">
  <soap:binding transport="http://schemas.xmlsoap.org/soap/http"
style="document"/>
  <operation name="LoadOperation">
    <ns5:PolicyReference
URI="#LoadWSPortBinding_LoadOperation_WSAT_Policy"
xmlns:ns5="http://www.w3.org/ns/ws-policy"/>

```

```

    <soap:operation soapAction=""/>
    <input>
      <soap:body use="literal"/>
    </input>
    <output>
      <soap:body use="literal"/>
    </output>
  </operation>
</binding>
<service name="LoadWSService">
  <port name="LoadWSPort" binding="tns:LoadWSPortBinding">
    <soap:address location="REPLACE_WITH_ACTUAL_URL"/>
  </port>
</service>
</definitions>

```

Table B.12: XML Schema for Load Web Service

```

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xs:schema version="1.0" targetNamespace="http://prototype.phd.awad/"
  xmlns:tns="http://prototype.phd.awad/"
  xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="LoadOperation" type="tns:LoadOperation"/>

  <xs:element name="LoadOperationResponse"
    type="tns:LoadOperationResponse"/>

  <xs:complexType name="LoadOperation">
    <xs:sequence>
      <xs:element name="loadParameter" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="LoadOperationResponse">
    <xs:sequence>
      <xs:element name="return" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>

```

Table B.13: BPEL XML Code

```

<?xml version="1.0" encoding="UTF-8"?>
<process
  name="ETL_Requestor"
  targetNamespace="http://enterprise.netbeans.org/bpel/ETL_Requestor/ETL_Requestor"
  xmlns="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:sxt="http://www.sun.com/wsbpel/2.0/process/executable/SUNExtension/Trace"
  xmlns:sxed="http://www.sun.com/wsbpel/2.0/process/executable/SUNExtension/Editor"
  xmlns:sxat="http://www.sun.com/wsbpel/2.0/process/executable/SUNExtension/Attachment"

  xmlns:sxeh="http://www.sun.com/wsbpel/2.0/process/executable/SUNExtension/ErrorHandlerling"
  xmlns:tns="http://enterprise.netbeans.org/bpel/ETL_Requestor/ETL_Requestor"
  xmlns:sxed2="http://www.sun.com/wsbpel/2.0/process/executable/SUNExtension/Editor2"
  xmlns:ns0="http://xml.netbeans.org/schema/ETL_Requestor">
  <import namespace="http://j2ee.netbeans.org/wsd/ETL_Requestor/ETL_Requestor"
location="ETL_Requestor.wsd" importType="http://schemas.xmlsoap.org/wsd/" />
  <import namespace="http://enterprise.netbeans.org/bpel/ExtractWSServiceWrapper"
location="ExtractWSServiceWrapper.wsd" importType="http://schemas.xmlsoap.org/wsd/" />
  <import namespace="http://prototype.phd.awad/"
location="ExtractEJB/wsd/ExtractWSService.wsd"
importType="http://schemas.xmlsoap.org/wsd/" />
  <import namespace="http://enterprise.netbeans.org/bpel/TransformWSServiceWrapper"
location="TransformWSServiceWrapper.wsd"
importType="http://schemas.xmlsoap.org/wsd/" />
  <import namespace="http://prototype.phd.awad/"
location="TransformEJB/wsd/TransformWSService.wsd"
importType="http://schemas.xmlsoap.org/wsd/" />
  <import namespace="http://enterprise.netbeans.org/bpel/ClassifyWSServiceWrapper"
location="ClassifyWSServiceWrapper.wsd" importType="http://schemas.xmlsoap.org/wsd/" />
  <import namespace="http://prototype.phd.awad/"
location="ClassifyEJB/wsd/ClassifyWSService.wsd"
importType="http://schemas.xmlsoap.org/wsd/" />
  <import namespace="http://enterprise.netbeans.org/bpel/LoadWSServiceWrapper"
location="LoadWSServiceWrapper.wsd" importType="http://schemas.xmlsoap.org/wsd/" />
  <import namespace="http://prototype.phd.awad/"
location="LoadEJB/wsd/LoadWSService.wsd"
importType="http://schemas.xmlsoap.org/wsd/" />
  <partnerLinks>
    <partnerLink name="Extract_PI."
  xmlns:tns="http://enterprise.netbeans.org/bpel/ExtractWSServiceWrapper"

```

```

partnerLinkType="tns:ExtractWSLinkType" partnerRole="ExtractWSRole"/>
  <partnerLink name="Transform_PL"
xmlns:tns="http://enterprise.netbeans.org/bpel/TransformWSServiceWrapper"
partnerLinkType="tns:TransformWSLinkType" partnerRole="TransformWSRole"/>
    <partnerLink name="Classify_PL"
xmlns:tns="http://enterprise.netbeans.org/bpel/ClassifyWSServiceWrapper"
partnerLinkType="tns:ClassifyWSLinkType" partnerRole="ClassifyWSRole"/>
        <partnerLink name="Load_PL"
xmlns:tns="http://enterprise.netbeans.org/bpel/LoadWSServiceWrapper"
partnerLinkType="tns:LoadWSLinkType" partnerRole="LoadWSRole"/>
            <partnerLink name="ETLAdministrators_PL"
xmlns:tns="http://j2ee.netbeans.org/wsd/ETL_Requestor/ETL_Requestor"
partnerLinkType="tns:ETL" myRole="ETL_RequestorPortTypeRole"/>
        </partnerLinks>
    <variables>
        <variable name="LoadOperationOut" xmlns:tns="http://prototype.phd.awad/"
messageType="tns:LoadOperationResponse"/>
        <variable name="LoadOperationIn" xmlns:tns="http://prototype.phd.awad/"
messageType="tns:LoadOperation"/>
        <variable name="ClassifyOperationOut" xmlns:tns="http://prototype.phd.awad/"
messageType="tns:ClassifyOperationResponse"/>
        <variable name="ClassifyOperationIn" xmlns:tns="http://prototype.phd.awad/"
messageType="tns:ClassifyOperation"/>
        <variable name="TransformOperationOut" xmlns:tns="http://prototype.phd.awad/"
messageType="tns:TransformOperationResponse"/>
        <variable name="TransformOperationIn" xmlns:tns="http://prototype.phd.awad/"
messageType="tns:TransformOperation"/>
        <variable name="ETL_RequestorOperationOut"
xmlns:tns="http://j2ee.netbeans.org/wsd/ETL_Requestor/ETL_Requestor"
messageType="tns:ETL_RequestorOperationResponse"/>
        <variable name="ExtractOperationOut" xmlns:tns="http://prototype.phd.awad/"
messageType="tns:ExtractOperationResponse"/>
        <variable name="ExtractOperationIn" xmlns:tns="http://prototype.phd.awad/"
messageType="tns:ExtractOperation"/>
        <variable name="ETL_RequestorOperationIn"
xmlns:tns="http://j2ee.netbeans.org/wsd/ETL_Requestor/ETL_Requestor"
messageType="tns:ETL_RequestorOperationRequest"/>
    </variables>
    <sequence>
        <receive name="ReceiveAdministratorRequest" createInstance="yes"
partnerLink="ETLAdministrators_PL" operation="ETL_RequestorOperation"
xmlns:tns="http://j2ee.netbeans.org/wsd/ETL_Requestor/ETL_Requestor"
portType="tns:ETL_RequestorPortType" variable="ETL_RequestorOperationIn"/>
        <if name="If1">
            <condition>
                ($ETL_RequestorOperationOut.responsePart/ns0:operationReturn="extract")
            </condition>
            <sequence name="SequenceExtract">
                <assign name="AssignExtract">

```

```

        <copy>
            <from>$ETL_RequestorOperationOut.responsePart/ns0:operationReturn</from>
            <to>$ExtractOperationIn.parameters/extractParameter</to>
        </copy>
    </assign>
    <invoke name="InvokeExtract" partnerLink="Extract_PL"
operation="ExtractOperation" xmlns:tns="http://prototype.phd.awad/"
portType="tns:ExtractWS" inputVariable="ExtractOperationIn"
outputVariable="ExtractOperationOut"/>
    <assign name="AssignExtract2">
        <copy>
            <from>$ExtractOperationOut.parameters/return</from>
            <to>$ETL_RequestorOperationIn.requestPart/ns0:operationParameter</to>
        </copy>
    </assign>
    <reply name="ReplyExtract" partnerLink="ETLAdministrators_PL"
operation="ETL_RequestorOperation"
xmlns:tns="http://j2ee.netbeans.org/wsdl/ETL_Requestor/ETL_Requestor"
portType="tns:ETL_RequestorPortType" variable="ETL_RequestorOperationOut"/>
    </sequence>
</elseif>
    <condition>
        ($ETL_RequestorOperationOut.responsePart/ns0:operationReturn="transform")
    </condition>
    <sequence name="SequenceTransform">
        <assign name="AssignTransform">
            <copy>
                <from>$ETL_RequestorOperationOut.responsePart/ns0:operationReturn</from>
                <to>$TransformOperationIn.parameters/transformParameter</to>
            </copy>
        </assign>
        <invoke name="InvokeTransform" partnerLink="Transform_PL"
operation="TransformOperation" xmlns:tns="http://prototype.phd.awad/"
portType="tns:TransformWS" inputVariable="TransformOperationIn"
outputVariable="TransformOperationOut"/>
        <assign name="AssignTransform2">
            <copy>
                <from>$TransformOperationOut.parameters/return</from>
                <to>$ETL_RequestorOperationIn.requestPart/ns0:operationParameter</to>
            </copy>
        </assign>
        <reply name="ReplyTransform" partnerLink="ETLAdministrators_PL"
operation="ETL_RequestorOperation"
xmlns:tns="http://j2ee.netbeans.org/wsdl/ETL_Requestor/ETL_Requestor"
portType="tns:ETL_RequestorPortType" variable="ETL_RequestorOperationOut"/>
    </sequence>
</elseif>
</elseif>

```

```

        <condition>
        ($ETL_RequestorOperationOut.responsePart/ns0:operationReturn="classify")
        </condition>
        <sequence name="SequenceClassify">
            <assign name="AssignClassify">
                <copy>

<from>$ETL_RequestorOperationOut.responsePart/ns0:operationReturn</from>
            <to>$ClassifyOperationIn.parameters/classifyParameter</to>
            </copy>
        </assign>
        <invoke name="InvokeClassify" partnerLink="Classify_PL"
operation="ClassifyOperation" xmlns:tns="http://prototype.phd.awad/"
portType="tns:ClassifyWS" inputVariable="ClassifyOperationIn"
outputVariable="ClassifyOperationOut"/>
            <assign name="AssignClassify2">
                <copy>
                    <from>$ClassifyOperationOut.parameters/return</from>
                    <to>$ETL_RequestorOperationIn.requestPart/ns0:operationParameter</to>
                </copy>
            </assign>
            <reply name="ReplyClassify" partnerLink="ETLAdministrators_PL"
operation="ETL_RequestorOperation"
xmlns:tns="http://j2ee.netbeans.org/wsdl/ETL_Requestor/ETL_Requestor"
portType="tns:ETL_RequestorPortType" variable="ETL_RequestorOperationOut"/>
        </sequence>
    </elseif>
    <elseif>
        <condition>
        ($ETL_RequestorOperationOut.responsePart/ns0:operationReturn="load")
        </condition>
        <sequence name="SequenceLoad">
            <assign name="AssignLoad">
                <copy>

<from>$ETL_RequestorOperationOut.responsePart/ns0:operationReturn</from>
            <to>$LoadOperationIn.parameters/loadParameter</to>
            </copy>
        </assign>
        <invoke name="InvokeLoad" partnerLink="Load_PL" operation="LoadOperation"
xmlns:tns="http://prototype.phd.awad/" portType="tns:LoadWS"
inputVariable="LoadOperationIn" outputVariable="LoadOperationOut"/>
            <assign name="AssignLoad2">
                <copy>
                    <from>$LoadOperationOut.parameters/return</from>
                    <to>$ETL_RequestorOperationIn.requestPart/ns0:operationParameter</to>
                </copy>
            </assign>
            <reply name="ReplyLoad" partnerLink="ETLAdministrators_PL"

```

```

operation="ETL_RequestorOperation"
xmlns:tns="http://j2ee.netbeans.org/wsd/ETL_Requestor/ETL_Requestor"
portType="tns:ETL_RequestorPortType" variable="ETL_RequestorOperationOut"/>
    </sequence>
  </elseif>
</if>
</sequence>
</process>

```

Table B.14: WSDL Code of the BPEL

```

<?xml version="1.0" encoding="UTF-8"?>
<definitions name="ETL"
targetNamespace="http://j2ee.netbeans.org/wsd/ETL_Requestor/ETL_Requestor"
  xmlns="http://schemas.xmlsoap.org/wsd/"
  xmlns:wsdl="http://schemas.xmlsoap.org/wsd/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:tns="http://j2ee.netbeans.org/wsd/ETL_Requestor/ETL_Requestor"
  xmlns:ns="http://xml.netbeans.org/schema/ETL_Requestor"
  xmlns:plnk="http://docs.oasis-open.org/wsbpel/2.0/plnktype"
  xmlns:tns1="http://j2ee.netbeans.org/wsd/ETL_Requestor/ETL_Requestor">
  <types>
    <xsd:schema targetNamespace="#TARGET_NAMESPACE">
      <xsd:import
namespace="http://xml.netbeans.org/schema/ETL_Requestor"
schemaLocation="ETL_Requestor.xsd"/>
    </xsd:schema>
  </types>
  <message name="ETL_RequestorOperationRequest">
    <part name="requestPart" element="ns:processApplicElement"/>
  </message>
  <message name="ETL_RequestorOperationResponse">
    <part name="responsePart" element="ns:processApplicRespElement"/>
  </message>
  <portType name="ETL_RequestorPortType">
    <operation name="ETL_RequestorOperation">
      <input name="input1"
message="tns1:ETL_RequestorOperationRequest"/>
      <output name="output1"
message="tns1:ETL_RequestorOperationResponse"/>
    </operation>

```

```

    </portType>
    <plnk:partnerLinkType name="ETL">
      <!-- A partner link type is automatically generated when a new port type is
      added. Partner link types are used by BPEL processes.
      In a BPEL process, a partner link represents the interaction between the
      BPEL process and a partner service. Each partner link is associated with a
      partner link type.
      A partner link type characterizes the conversational relationship between
      two services. The partner link type can have one or two roles.-->
      <plnk:role name="ETL_RequestorPortTypeRole"
      portType="tns1:ETL_RequestorPortType"/>
    </plnk:partnerLinkType>
  </definitions>

```

Table B.15L: XML Schema of the BPEL

```

<?xml version="1.0" encoding="UTF-8"?>

<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://xml.netbeans.org/schema/ETL_Requestor"
  xmlns:tns="http://xml.netbeans.org/schema/ETL_Requestor"
  elementFormDefault="qualified">
  <xsd:complexType name="processApplicType">
    <xsd:sequence>
      <xsd:element name="operationParameter" nillable="true"
type="xsd:string"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="prossApplicRespType">
    <xsd:sequence>
      <xsd:element name="operationReturn" nillable="true"
type="xsd:string"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:element name="processApplicElement" type="tns:processApplicType"/>
  <xsd:element name="processApplicRespElement"
type="tns:prossApplicRespType"/>
</xsd:schema>

```

Table B.16: Code of the composite application

```

<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<casa xmlns="http://java.sun.com/xml/ns/casa" xmlns:ns1="ETL_CompositeApp"

```

```

xmlns:ns2="http://j2ee.netbeans.org/wsdl/ETL_Requestor/ETL_Requestor"
xmlns:ns3="http://prototype.phd.awad/"
xmlns:ns4="http://enterprise.netbeans.org/bpel/ETL_Requestor/ETL_Requestor"
xmlns:xlink="http://www.w3.org/2000/xlink">
  <endpoints>
    <endpoint endpoint-name="AdministratorPort" interface-
name="ns2:ETL_RequestorPortType" name="endpoint1" service-
name="ns1:casaService1"/>
    <endpoint endpoint-name="ExtractPort" interface-name="ns3:ExtractWS"
name="endpoint2" service-name="ns1:casaService2"/>
    <endpoint display-name="ETLAdministrators_PL" endpoint-
name="ETL_RequestorPortTypeRole_myRole" file-path="ETL_Requestor.bpel"
interface-name="ns2:ETL_RequestorPortType" name="endpoint3" process-
name="ETL_Requestor" service-name="ns4:ETLAdministrators_PL"/>
    <endpoint display-name="Extract_PL" endpoint-
name="ExtractWSRole_partnerRole" file-path="ETL_Requestor.bpel" interface-
name="ns3:ExtractWS" name="endpoint4" process-name="ETL_Requestor"
service-name="ns4:Extract_PL"/>
    <endpoint display-name="Transform_PL" endpoint-
name="TransformWSRole_partnerRole" file-path="ETL_Requestor.bpel"
interface-name="ns3:TransformWS" name="endpoint5" process-
name="ETL_Requestor" service-name="ns4:Transform_PL"/>
    <endpoint display-name="Classify_PL" endpoint-
name="ClassifyWSRole_partnerRole" file-path="ETL_Requestor.bpel" interface-
name="ns3:ClassifyWS" name="endpoint6" process-name="ETL_Requestor"
service-name="ns4:Classify_PL"/>
    <endpoint display-name="Load_PL" endpoint-
name="LoadWSRole_partnerRole" file-path="ETL_Requestor.bpel" interface-
name="ns3:LoadWS" name="endpoint7" process-name="ETL_Requestor" service-
name="ns4:Load_PL"/>
    <endpoint name="endpoint8" endpoint-name="TransformPort" interface-
name="ns3:TransformWS" service-name="ns1:casaService3"/>
    <endpoint name="endpoint9" endpoint-name="ClassifyPort" interface-
name="ns3:ClassifyWS" service-name="ns1:casaService4"/>
    <endpoint name="endpoint10" endpoint-name="LoadPort" interface-
name="ns3:LoadWS" service-name="ns1:casaService5"/>
  </endpoints>
  <service-units>
    <service-engine-service-unit artifacts-zip="ETL_Requestor.jar" component-
name="sun-bpel-engine" defined="true" description="Represents this Service Unit"
internal="true" name="ETL_CompositeApp-ETL_Requestor" unit-
name="ETL_Requestor" unknown="false" x="35" y="99">
      <provides endpoint="endpoint3"/>
      <consumes endpoint="endpoint4"/>
      <consumes endpoint="endpoint5"/>
      <consumes endpoint="endpoint6"/>
      <consumes endpoint="endpoint7"/>
    </service-engine-service-unit>
    <binding-component-service-unit artifacts-zip="sun-http-binding.jar"

```

```

component-name="sun-http-binding" description="Represents this Service Unit"
name="ETL_CompositeApp-sun-http-binding" unit-name="sun-http-binding">
  <ports>
    <port bindingType="soap" x="67" y="39">
      <link
xlink:href="../../jbiasa/ETL_CompositeApp.wsdl#xpointer(/definitions/service[@nam
e='casaService1']/port[@name='AdministratorPort'])" xlink:type="simple"/>
      <consumes endpoint="endpoint1"/>
      <provides endpoint="endpoint1"/>
    </port>
    <port bindingType="soap" x="67" y="118">
      <link
xlink:href="../../jbiasa/ETL_CompositeApp.wsdl#xpointer(/definitions/service[@nam
e='casaService2']/port[@name='ExtractPort'])" xlink:type="simple"/>
      <consumes endpoint="endpoint2"/>
      <provides endpoint="endpoint2"/>
    </port>
    <port x="67" y="197" bindingType="soap">
      <link
xlink:href="../../jbiasa/ETL_CompositeApp.wsdl#xpointer(/definitions/service[@nam
e='&apos;casaService3&apos;']/port[@name='&apos;TransformPort&apos;'])"
xlink:type="simple"/>
      <consumes endpoint="endpoint8"/>
      <provides endpoint="endpoint8"/>
    </port>
    <port x="67" y="276" bindingType="soap">
      <link
xlink:href="../../jbiasa/ETL_CompositeApp.wsdl#xpointer(/definitions/service[@nam
e='&apos;casaService4&apos;']/port[@name='&apos;ClassifyPort&apos;'])"
xlink:type="simple"/>
      <consumes endpoint="endpoint9"/>
      <provides endpoint="endpoint9"/>
    </port>
    <port x="67" y="355" bindingType="soap">
      <link
xlink:href="../../jbiasa/ETL_CompositeApp.wsdl#xpointer(/definitions/service[@nam
e='&apos;casaService5&apos;']/port[@name='&apos;LoadPort&apos;'])"
xlink:type="simple"/>
      <consumes endpoint="endpoint10"/>
      <provides endpoint="endpoint10"/>
    </port>
  </ports>
</binding-component-service-unit>
</service-units>
<connections>
  <connection consumer="endpoint4" provider="endpoint2" state="new"/>
  <connection consumer="endpoint1" provider="endpoint3" state="new"/>
  <connection state="new" consumer="endpoint5" provider="endpoint8"/>
  <connection state="new" consumer="endpoint6" provider="endpoint9"/>

```

```

        <connection state="new" consumer="endpoint7" provider="endpoint10"/>
    </connections>
    <porttypes>
        <link
xlink:href=" ../jbiasa/ETL_CompositeApp.wsdl#xpointer(/definitions/portType[@na
me='dummyCasaPortType'])" xlink:type="simple"/>
        <link
xlink:href=" ../jbiServiceUnits/ETL_Requestor/ETL_Requestor.wsdl#xpointer(/defi
nitions/portType[@name='ETL_RequestorPortType'])" xlink:type="simple"/>
        <link xlink:href=" ../jbiServiceUnits/META-
INF/ETL_Requestor/src/_references/_projects/ClassifyEJB/src/conf/wsdl/Classify
WSService.wsdl#xpointer(/definitions/portType[@name='ClassifyWS'])"
xlink:type="simple"/>
        <link xlink:href=" ../jbiServiceUnits/META-
INF/ETL_Requestor/src/_references/_projects/LoadEJB/src/conf/wsdl/LoadWSSer
vice.wsdl#xpointer(/definitions/portType[@name='LoadWS'])"
xlink:type="simple"/>
        <link xlink:href=" ../jbiServiceUnits/META-
INF/ETL_Requestor/src/_references/_projects/TransformEJB/src/conf/wsdl/Transfo
rmWSService.wsdl#xpointer(/definitions/portType[@name='TransformWS'])"
xlink:type="simple"/>
        <link xlink:href=" ../jbiServiceUnits/META-
INF/ETL_Requestor/src/_references/_projects/ExtractEJB/src/conf/wsdl/ExtractWS
Service.wsdl#xpointer(/definitions/portType[@name='ExtractWS'])"
xlink:type="simple"/>
        <link
xlink:href=" ../jbiServiceUnits/ETL_CompositeApp.wsdl#xpointer(/definitions/port
Type[@name='dummyCasaPortType'])" xlink:type="simple"/>
    </porttypes>
    <bindings>
        <link
xlink:href=" ../jbiasa/ETL_CompositeApp.wsdl#xpointer(/definitions/binding[@na
me='casaBinding1'])" xlink:type="simple"/>
        <link
xlink:href=" ../jbiasa/ETL_CompositeApp.wsdl#xpointer(/definitions/binding[@na
me='casaBinding2'])" xlink:type="simple"/>
        <link xlink:href=" ../jbiServiceUnits/META-
INF/ETL_Requestor/src/_references/_projects/ClassifyEJB/src/conf/wsdl/Classify
WSService.wsdl#xpointer(/definitions/binding[@name='ClassifyWSportBinding'])"
xlink:type="simple"/>
        <link xlink:href=" ../jbiServiceUnits/META-
INF/ETL_Requestor/src/_references/_projects/LoadEJB/src/conf/wsdl/LoadWSSer
vice.wsdl#xpointer(/definitions/binding[@name='LoadWSportBinding'])"
xlink:type="simple"/>
        <link xlink:href=" ../jbiServiceUnits/META-
INF/ETL_Requestor/src/_references/_projects/TransformEJB/src/conf/wsdl/Transfo
rmWSService.wsdl#xpointer(/definitions/binding[@name='TransformWSportBindi
ng'])" xlink:type="simple"/>
        <link xlink:href=" ../jbiServiceUnits/META-

```

```

INF/ETL_Requestor/src/_references/_projects/ExtractEJB/src/conf/wsd/ExtractWS
Service.wsd#xpointer(/definitions/binding[@name='ExtractWSPortBinding'])"
xlink:type="simple"/>
    <link
xlink:href=" ../jbiServiceUnits/ETL_CompositeApp.wsd#xpointer(/definitions/bindi
ng[@name='casaBinding1'])" xlink:type="simple"/>
    <link
xlink:href=" ../jbiServiceUnits/ETL_CompositeApp.wsd#xpointer(/definitions/bindi
ng[@name='casaBinding2'])" xlink:type="simple"/>
    </bindings>
    <services>
    <link
xlink:href=" ../jbiasa/ETL_CompositeApp.wsd#xpointer(/definitions/service[@nam
e='casaService1'])" xlink:type="simple"/>
    <link
xlink:href=" ../jbiasa/ETL_CompositeApp.wsd#xpointer(/definitions/service[@nam
e='casaService2'])" xlink:type="simple"/>
    <link xlink:href=" ../jbiServiceUnits/META-
INF/ETL_Requestor/src/_references/_projects/ClassifyEJB/src/conf/wsd/Classify
WSService.wsd#xpointer(/definitions/service[@name='ClassifyWSService'])"
xlink:type="simple"/>
    <link xlink:href=" ../jbiServiceUnits/META-
INF/ETL_Requestor/src/_references/_projects/LoadEJB/src/conf/wsd/LoadWSSer
vice.wsd#xpointer(/definitions/service[@name='LoadWSService'])"
xlink:type="simple"/>
    <link xlink:href=" ../jbiServiceUnits/META-
INF/ETL_Requestor/src/_references/_projects/TransformEJB/src/conf/wsd/Transfo
rmWSService.wsd#xpointer(/definitions/service[@name='TransformWSService'])"
xlink:type="simple"/>
    <link xlink:href=" ../jbiServiceUnits/META-
INF/ETL_Requestor/src/_references/_projects/ExtractEJB/src/conf/wsd/ExtractWS
Service.wsd#xpointer(/definitions/service[@name='ExtractWSService'])"
xlink:type="simple"/>
    <link
xlink:href=" ../jbiServiceUnits/ETL_CompositeApp.wsd#xpointer(/definitions/servi
ce[@name='casaService1'])" xlink:type="simple"/>
    <link
xlink:href=" ../jbiServiceUnits/ETL_CompositeApp.wsd#xpointer(/definitions/servi
ce[@name='casaService2'])" xlink:type="simple"/>
    </services>
    <regions>
    <region name="WSDL Endpoints" width="150"/>
    <region name="JBI Modules" width="904"/>
    <region name="External Modules" width="200"/>
    </regions>
</casa>

```

Appendix C

Questionnaires as Deliverables of Structured Interviews Done with Industry Experts

- *First Questionnaire:* is the deliverable of a structured interview done with a *DW specialist* from Singapore branch of **Sybase** international company. The interview is done by phone on 26 May, 2010, and then, the questionnaire is filled and sent back by email on 31 May, 2010.
- *Second Questionnaire:* is the deliverable of a structured interview done with a *DW administrator* from Malaysia branch of **SAS** international company. The interview is done on 1 June, 2010 at the company location.
- *Third Questionnaire:* is the deliverable of a structured interview done with an *IT consultant* from Malaysia branch of **SAS** international company. The interview is done on 1 June, 2010 at the company location.
- *Fourth Questionnaire:* is the deliverable of a structured interview done with a *DW administrator* from R&D department of **TM** (Telekom Malaysia), Malaysian Company. The interview is done on 15 June, 2010 at R&D Cyberjaya branch of the company.
- *Fifth Questionnaire:* is the deliverable of a structured interview done with a *software developer* from **EIGC** Company in Dubai, UAE. The interview is done by phone on 10 May, 2010, and then, the questionnaire is filled and sent back by email on 30 May, 2010.
- *Sixth Questionnaire:* is the deliverable of a structured interview done with a *DW manager* from **PIT** Palestinian company. The interview is done by phone on 23 April, 2010, and then, the questionnaire is filled and sent back by email on 27 April, 2010.

Expert Opinion Interview and Questionnaire

Name: Amit Vidyasagar
Email: amitv@sybase.com
Hand-Phone: +65 8233 4746
Telephone: +65 6571 3000
Organization Name: Sybase Singapore Pte. Ltd.
Organization Website: www.sybase.com
Organization address: 438B Alexandra Road,
#04-01/04 Alexandra Technopark,
Singapore 119968

Section 1: Information about you and your organization.


1. Which best describes your current position?
 - ☐ DW Analyst
 - ☐ DW Manager
 - ☐ DW Specialist
(architect/engineer/developer)
 - ☐ Project Leader
 - ☐ IT Consultant
 - ☐ DW Administrator
 - ☐ Software/Web developer
 - ☐ System Architect/Analyst
2. How long have you been in your current position?
 - ☐ Less than 1 year
 - ☐ 3 to 6 years
 - ☐ 1 to 3 years
 - ☐ More than 6 years
3. What is the current status of DW activities in your organization?
 - ☐ Develop and implement DW projects for organizations
 - ☐ Consulting DW projects
 - ☐ Do general Web applications
 - ☐ Just started doing DW projects
 - ☐ Planning to enter DW field in the future
 - ☐ Develop and enhance DW tools
4. How long have DW, SOA, or Web services in your organization been in existence?
 - ☐ Less than 1 year
 - ☐ 3 to 6 years
 - ☐ Not available.
 - ☐ 1 to 3 years
 - ☐ More than 6 years
5. How long have you worked for this organization?
 - ☐ Less than 2 years
 - ☐ 5 to 10 years
 - ☐ 2 to 5 years
 - ☐ More than 10 years

6. What is the percentage of including web services or SOA in your organization projects?

- ☐ Not used at all
- ☐ Less than 10%
- ☐ From 10% to 40%
- ☐ From 40% to 70%
- ☐ From 70% to 90%
- ☐ From 90% to 100%

Section 2: Problems of the current ETL framework

For each of the following statement, please circle the number that indicates the extent to which you agree or disagree.

		Strongly Disagree  Strongly Agree				
PR1	If DW extracts its data from many different locations, and each location administrator needs to push the data from his side. Furthermore, you use a licensed ETL tool for each location. Therefore, the budget of the project will be higher because of the number of licenses needed.	1	2	3	4	
PR2	Transformation and Loading features are redundant at the Extraction location because only Extraction is needed at data source locations. In addition, those two features would be necessary only in the location of the data warehouse destination. That leads to redundancy problem i.e. we pay a license for all ETL features and use some of them in some data location.	1	2		4	5
PR3	In the current ETL framework, including all the features as tightly-coupled components in one software; decreases the ease of use, because it is easier for an ETL user to learn how to use only the needed functionality rather than learning all ETL functionalities.	1	2		4	5
PR4	Managing many ETL tools in many data source locations for the same project to extract data from many different sources; needs extra administration, communication and maintenance effort, because the administrators are often different persons from one location to another and they could use different ETL tools and do different configurations to those tools.	1	2	3		5
PR5	Sometimes due to the complexity, long learning curve of the available ETL tools, and difficulty to achieve some extensibility in terms of additional functionalities; some organizations prefer to turn to in-house development to perform ETL tasks, which increases the project effort.	1	2	3	4	
PR6	In the current ETL framework, it is not easy to extend ETL components by adding other components; to suite additional special business needs.	1	2	3		5
PR7	Not all ETL tools are compatible with all server environments, and each vendor's tool runs on specific environment including specific operating system and other requirements. That is considered sometimes an impediment for some customers.	1	2	3		5

PR8	In the current ETL framework, there are many impediments to include ETL as a part of a complete portal that manages the DW project, because of lack of standards in the current ETL framework to communicate with other components of the portal.	1	2	3	5

Section 3: Solutions for the problems after restructuring the current ETL framework based on SOA

For each of the following statement, please circle the number that indicates the extent to which you agree or disagree.

Strongly Disagree ← Strongly Agree

SO1	Using the enhanced ETL framework solves the problem of licenses needed for each data source location, because ETL components can be deployed once on application server(s). Then, the same deployed version can be used remotely by all parties of the DW project.	1	2	3	5
SO2	Using the enhanced ETL framework solves the problem of component redundancy, since only one common component is available in the deployed version. Then, the needed component can be invoked remotely by the administrator and it is the same component for other administrators in other locations.	1	2	3	5
SO3	Using the enhanced ETL framework solves the problem of difficult usability, because ETL administrator needs to learn how to use only the component he needs, and not the whole ETL tool.	1	2	4	5
SO4	Using the enhanced ETL framework solves the problem of extra administration, communication and maintenance effort, because all the administrators share the same component which can be configured once, and each administrator can follow the common manual to use the component for his special needs.	1	2	3	5
SO5	Using the enhanced ETL framework solves the problem in which many organizations prefer to turn to in-house development to perform ETL tasks, because the new framework is extendable and any extra component to suit special business needs can be merged easily to the framework rather than rebuilding the ETL from the scratch.	1	2	3	5
SO6	Using the enhanced ETL framework solves the problem of lack of extensibility, because the components of the enhanced framework are distributable, inter-operable, and loosely-coupled to each others. So, the framework can be extended without complications.	1	2	4	5
SO7	Using the enhanced ETL framework solves the problem of environment	1	2	3	5

	compatibility, because all the compatibility requirements can be configured once on the shared application server which the components will be deployed to.				
SO8	Using the enhanced ETL framework solves the problem of including ETL as a part of a web portal because the ETL components are built based on SOA which guarantees the portability of the components, and it follows web services standards for communication among components.	1	2	3	5

Section 4: Advantages of the restructured ETL framework over the traditional one

For each of the following statement, please circle the number that indicates the extent to which you agree or disagree.

		Strongly Disagree ← → Strongly Agree			
AD1	<i>Interoperability:</i> The interaction between clients (ETL administrators) and loosely-coupled ETL services has widespread interoperability. It is required that clients and services can communicate and understand each other no matter what platform they run on. This target is met because clients and ETL services have a standard way of communication among each other. In addition, interoperability provides consistency across platforms, systems, and languages for the ETL framework components.	1	2	3	5
AD2	<i>Flexibility and extensibility:</i> Loosely-coupled services in ETL framework are typically more flexible than tightly-coupled ETL framework. In the traditional framework, the ETL components are tightly-coupled, share semantics, libraries, and often share state. This makes it difficult to evolve the application to keep up with changing business requirements. The loosely-coupled, document-based, and asynchronous natures of loosely-coupled services allow applications to be flexible, and easy to evolve with requirement changes. Therefore, the new ETL framework can have extra components without complexities. Based on that, it can be extended in the future by adding any extra component to suit enterprises' new business requirements.	1	2	3	5
AD3	<i>Reusability:</i> In case that the new ETL framework is followed, ETL developers in data warehouse industry can reuse the code of an existing ETL component developed by any other ETL provider; to meet new ETL business requirements. Reusing functionality that already exists outside or inside an enterprise instead of developing code that reproduces those functions; can result in a big savings in application development cost and time.	1	2	3	5
AD4	<i>Scalability:</i> Because ETL framework services are loosely-coupled, applications that follow the framework standards can scale easily, at least easier than applications that follow more tightly-coupled frameworks. That is because there are few dependencies between the requesting	1	2	3	5

	application and the services it uses. That enables the service to handle more requests because it does not need to handle lots of communication overhead like tightly-coupled components.				
AD5	<i>Cost Efficiency:</i> An ETL framework based on SOA, results in cheaper solutions because the integration of clients and services does not require heavy analysis and unique code of customized solutions. Furthermore, because services in the enhanced framework are loosely-coupled, applications that use these services are cheaper to maintain and easier to extend than customized solutions. Furthermore, administration, communication and maintenance effort will be less, because all the administrators share the same component which can be configured once, and each administrator can follow the common manual to use the component for his special needs. In addition, the solution of redundancy and licenses problem supports cheaper solutions.	1	2	3	5
AD6	<i>Ease of use:</i> Using the enhanced ETL framework solves the problem of usability difficulty, because the ETL administrator (user) needs to learn how to use only the component he needs, and not the whole ETL tool.	1	2	3	5
AD7	<i>Compatibility:</i> Using the enhanced ETL framework solves the problem of compatibility, because all the compatibility requirements can be configured once on the shared application server which the components are deployed to.	1	2	3	5
AD8	<i>Portability:</i> Using the enhanced ETL framework solves the problem of including ETL as a part of a web portal because the ETL components are built based on SOA which guarantees the portability of the components, and it follows web services standards for communication among components.	1	2	3	5

Kindly give your advices and comments if any:

Thank You

EXPERT OPINION INTERVIEW AND QUESTIONNAIRE

Name: Farhan Bin Sami
 Email: farhanbinisamail@sa.com
 Handphone: 03-2273-6288
 Telephone: 03-2273-6288
 Organization Name: SAS Institute Sdn. Bhd.
 Organization Website: www.sas.com
 Organization address: SAS INSTITUTE SDN. BHD. (153715-7)
Suite 2B-6-1, Level 6, Block 2B, Plaza Sentral,
Jalan Stesen Sentral 5, Kuala Lumpur Sentral
50470 Kuala Lumpur.
Tel: 03-2273 6288 Fax: 03-2273 7971

Section 1: Information about you and your organization.

1. Which best describes your current position?

<input type="radio"/> DW Analyst	<input type="radio"/> IT Consultant
<input type="radio"/> DW Manager	<input checked="" type="radio"/> DW Administrator
<input type="radio"/> DW Specialist (architect/engineer/developer)	<input type="radio"/> Software/Web developer
<input type="radio"/> Project Leader	<input type="radio"/> System Architect/Analyst
2. How long have you been in your current position?

<input checked="" type="radio"/> Less than 1 year	<input type="radio"/> 1 to 3 years
<input type="radio"/> 3 to 6 years	<input type="radio"/> More than 6 years
3. What is the current status of DW activities in your organization?

<input checked="" type="radio"/> Develop and implement DW projects for organizations	<input type="radio"/> Just started doing DW projects
<input checked="" type="radio"/> Consulting DW projects	<input checked="" type="radio"/> Planning to enter DW field in the future
<input type="radio"/> Do general Web applications	<input type="radio"/> Develop and enhance DW tools
4. How long have DW, SOA, or Web services in your organization been in existence?

<input type="radio"/> Less than 1 year	<input type="radio"/> 1 to 3 years
<input type="radio"/> 3 to 6 years	<input checked="" type="radio"/> More than 6 years
<input type="radio"/> Not available.	
5. How long have you worked for this organization?

<input type="radio"/> Less than 2 years	<input type="radio"/> 2 to 5 years
<input type="radio"/> 5 to 10 years	<input checked="" type="radio"/> More than 10 years
6. What is the percentage of including web services or SOA in your organization projects?

<input type="radio"/> Not used at all	<input checked="" type="radio"/> From 40% to 70%
<input type="radio"/> Less than 10%	<input type="radio"/> From 70% to 90%
<input type="radio"/> From 10% to 40%	<input type="radio"/> From 90% to 100%

Section 2: Problems of the current ETL framework

For each of the following statement, please circle the number that indicates the extent to which you agree or disagree.

		Strongly Disagree	1	2	3	4	5	Strongly Agree
PR1	If DW extracts its data from many different locations, and each location administrator needs to push the data from his side. Furthermore, you use a licensed ETL tool for each location. Therefore, the budget of the project will be higher because of the number of licenses needed.	1	2	3	4	5		
PR2	Transformation and Loading features are redundant at the Extraction location because only Extraction is needed at data source locations. In addition, those two features would be necessary only in the location of the data warehouse destination. That leads to redundancy problem i.e. we pay a license for all ETL features and use some of them in some data location.	1	2	3	4	5		
PR3	In the current ETL framework, including all the features as tightly-coupled components in one software, decreases the ease of use, because it is easier for an ETL user to learn how to use only the needed functionality rather than learning all ETL functionalities.	1	2	3	4	5		
PR4	Managing many ETL tools in many data source locations for the same project to extract data from many different sources; needs extra administration, communication and maintenance effort, because the administrators are often different persons from one location to another and they could use different ETL tools and do different configurations to those tools.	1	2	3	4	5		
PR5	Sometimes due to the complexity, long learning curve of the available ETL tools, and difficulty to achieve some extensibility in terms of additional functionalities; some organizations prefer to turn to in-house development to perform ETL tasks, which increases the project effort.	1	2	3	4	5		
PR6	In the current ETL framework, it is not easy to extend ETL components by adding other components; to suite additional special business needs.	1	2	3	4	5		
PR7	Not all ETL tools are compatible with all server environments, and each vendor's tool runs on specific environment including specific operating system and other requirements. That is considered sometimes an impediment for some customers.	1	2	3	4	5		
PR8	In the current ETL framework, there are many impediments to include ETL as a part of a complete portal that manages the DW project, because of lack of standards in the current ETL framework to communicate with other components of the portal.	1	2	3	4	5		

Section 3: Solutions for the problems after restructuring the current ETL framework based on SOA

For each of the following statements, please circle the number that indicates the extent to which you agree or disagree.

Strongly Disagree ← Strongly Agree

		1	2	3	4	5
SO1	Using the enhanced ETL framework solves the problem of licenses needed for each data source location, because ETL components can be deployed once on application servers. Then, the same deployed version can be used remotely by all parties of the DW project.				✓	
SO2	Using the enhanced ETL framework solves the problem of component redundancy, since only one common component is available in the deployed version. Then, the needed component can be invoked remotely by the administrator and it is the same component for other administrators in other locations.			✓	✓	
SO3	Using the enhanced ETL framework solves the problem of difficult usability, because ETL administrator needs to learn how to use only the component he needs, and not the whole ETL tool.			✓		
SO4	Using the enhanced ETL framework solves the problem of extra administration, communication and maintenance effort, because all the administrators share the same component which can be configured once, and each administrator can follow the common manual to use the component for his special needs.				✓	
SO5	Using the enhanced ETL framework solves the problem in which many organizations prefer to turn to in-house development to perform ETL tasks, because the new framework is extendable and any extra component to suit special business needs can be merged easily to the framework rather than rebuilding the ETL from the scratch.				✓	
SO6	Using the enhanced ETL framework solves the problem of lack of extensibility, because the components of the enhanced framework are distributable, inter-operable, and loosely-coupled to each others. So, the framework can be extended without complications.				✓	
SO7	Using the enhanced ETL framework solves the problem of environment compatibility, because all the compatibility requirements can be configured once on the shared application server which the components will be deployed to.				✓	
SO8	Using the enhanced ETL framework solves the problem of including ETL as a part of a web portal because the ETL components are built based on SOA which guarantees the portability of the components, and it follows web services standards for communication among components.			✓		

Section 4: Advantages of the restructured ETL framework over the traditional one

For each of the following statements, please circle the number that indicates the extent to which you agree or disagree.

		Strongly Disagree	←	→	Strongly Agree
AD1	<i>Interoperability:</i> The interaction between clients (ETL administrators) and loosely-coupled ETL services has widespread interoperability. It is required that clients and services can communicate and understand each other no matter what platform they run on. This target is met because clients and ETL services have a standard way of communication among each other. In addition, interoperability provides consistency across platforms, systems, and languages for the ETL framework components.	1	2	3	4 5
AD2	<i>Flexibility and extensibility:</i> Loosely-coupled services in ETL framework are typically more flexible than tightly-coupled ETL framework. In the traditional framework, the ETL components are tightly-coupled, share semantics, libraries, and often share state. This makes it difficult to evolve the application to keep up with changing business requirements. The loosely-coupled, document-based, and asynchronous natures of loosely-coupled services allow applications to be flexible, and easy to evolve with requirement changes. Therefore, the new ETL framework can have extra components without complexities. Based on that, it can be extended in the future by adding any extra component to suit enterprises' new business requirements.	1	2	3	4 5
AD3	<i>Reusability:</i> In case that the new ETL framework is followed, ETL developers in data warehouse industry can reuse the code of an existing ETL component developed by any other ETL provider, to meet new ETL business requirements. Reusing functionality that already exists outside or inside an enterprise instead of developing code that reproduces those functions, can result in a big savings in application development cost and time.	1	2	3	4 5
AD4	<i>Scalability:</i> Because ETL framework services are loosely-coupled, applications that follow the framework standards can scale easily, at least easier than applications that follow more tightly-coupled frameworks. That is because there are few dependencies between the requesting application and the services it uses. That enables the service to handle more requests because it does not need to handle lots of communication overhead like tightly-coupled components.	1	2	3	4 5
AD5	<i>Cost Efficiency:</i> An ETL framework based on SOA, results in cheaper solutions because the integration of clients and services does not require heavy analysis and unique code of customized solutions. Furthermore, because services in the enhanced framework are loosely-coupled, applications that use these services are cheaper to maintain and easier to extend than customized solutions. Furthermore, administration, communication and maintenance effort will be less, because all the administrators share the same component which can be configured once, and each administrator can follow the common manual to use the	1	2	3	4 5

Kindly give your advices and comments if any:

311

EXPERT OPINION INTERVIEW AND QUESTIONNAIRE

Name: CHONG SECK WAH
 Email: seckwah.chong@sas.com
 Business Unit:
 Telephone: 03-22736286
 Organization Name: SAS INSTITUTE SDN BHD
 Organization Website: ~~sdn bhd~~ www.sas.com
 Organization Address:

SAS INSTITUTE SDN BHD
 Suite 2B-6-1, Level 6, Block 2B, Plaza Sentral
 Jalan Stesen Sentral 5, Kuala Lumpur Sentral
 50470 Kuala Lumpur,
 Tel: 03-2273 6288 Fax: 03-2273 7971

Section 1: Information about you and your organization.

1. Which best describes your current position?

<input type="radio"/> DW Analyst	<input checked="" type="radio"/> IT Consultant
<input type="radio"/> DW Manager	<input type="radio"/> DW Administrator
<input type="radio"/> DW Specialist (architect/engineer/developer)	<input type="radio"/> Software Web developer
<input type="radio"/> Project Leader	<input type="radio"/> System Architect Analyst
2. How long have you been in your current position?

<input checked="" type="radio"/> Less than 1 year	<input type="radio"/> 1 to 3 years
<input type="radio"/> 3 to 6 years	<input type="radio"/> More than 6 years
3. What is the current status of DW activities in your organization?

<input checked="" type="radio"/> Develop and implement DW projects for organizations	<input type="radio"/> Just started doing DW projects
<input type="radio"/> Consulting DW projects	<input type="radio"/> Planning to enter DW field in the future
<input type="radio"/> Do general Web applications	<input type="radio"/> Develop and enhance DW tools
4. How long have DW, SOA, or Web services in your organization been in existence?

<input type="radio"/> Less than 1 year	<input type="radio"/> 1 to 3 years
<input type="radio"/> 3 to 6 years	<input checked="" type="radio"/> More than 6 years
<input checked="" type="radio"/> Not available	
5. How long have you worked for this organization?

<input checked="" type="radio"/> Less than 2 years	<input type="radio"/> 2 to 5 years
<input type="radio"/> 5 to 10 years	<input type="radio"/> More than 10 years
6. What is the percentage of including web services or SOA in your organization projects?

<input type="radio"/> Not used at all	<input checked="" type="radio"/> From 40% to 70%
<input type="radio"/> Less than 10%	<input type="radio"/> From 70% to 90%
<input type="radio"/> From 10% to 40%	<input type="radio"/> From 90% to 100%

Section 2: Problems of the current ETL framework

For each of the following statement, please check the number that indicates the extent to which you agree or disagree.

		Strongly Disagree ← → Strongly Agree				
		1	2	3	4	5
PR1	If DW extracts its data from many different locations, and each location administrator needs to push the data from his side. Furthermore, you use a licensed ETL tool for each location. Therefore, the budget of the project will be higher because of the number of licenses needed.	1	2	3	4	5
PR2	Transformation and Loading features are redundant at the Extraction location because only Extraction is needed at data source locations. In addition, those two features would be necessary only in the location of the data warehouse destination. That leads to redundancy problem i.e. we pay a license for all ETL features and use some of them in some data location.	1	2	3	4	5
PR3	In the current ETL framework, including all the features as tightly-coupled components in one software, decreases the ease of use, because it is easier for an ETL user to learn how to use only the needed functionality rather than learning all ETL functionalities.	1	2	3	4	5
PR4	Managing many ETL tools in many data source locations for the same project to extract data from many different sources; needs extra administration, communication and maintenance effort, because the administrators are often different persons from one location to another and they could use different ETL tools and do different configurations to those tools.	1	2	3	4	5
PR5	Sometimes due to the complexity, long learning curve of the available ETL tools, and difficulty to achieve some extensibility in terms of additional functionalities; some organizations prefer to turn to in-house development to perform ETL tasks, which increases the project effort.	1	2	3	4	5
PR6	In the current ETL framework, it is not easy to extend ETL components by adding other components; to suite additional special business needs.	1	2	3	4	5
PR7	Not all ETL tools are compatible with all server environments, and each vendor's tool runs on specific environment including specific operating system and other requirements. That is considered sometimes an impediment for some customers.	1	2	3	4	5
PR8	In the current ETL framework, there are many impediments to include ETL as a part of a complete portal that manages the DW project, because of lack of standards in the current ETL framework to communicate with other components of the portal.	1	2	3	4	5

Section 3: Solutions for the problems after restructuring the current ETL framework based on SOA

For each of the following statements, please circle the number that indicates the extent to which you agree or disagree.

Strongly Disagree ← → Strongly Agree

		1	2	3	4	5
S01	Using the enhanced ETL framework solves the problem of licenses needed for each data source location, because ETL components can be deployed once on application servers. Then, the same deployed version can be used remotely by all parties of the DW project.				(4)	
S02	Using the enhanced ETL framework solves the problem of component redundancy, since only one common component is available in the deployed version. Then, the needed component can be invoked remotely by the administrator and it is the same component for other administrators in other locations.			(3)		
S03	Using the enhanced ETL framework solves the problem of difficult usability, because ETL administrator needs to learn how to use only the component he needs, and not the whole ETL tool.			(3)		
S04	Using the enhanced ETL framework solves the problem of extra administration, communication and maintenance effort, because all the administrators share the same component which can be configured once, and each administrator can follow the common manual to use the component for his special needs.				(4)	
S05	Using the enhanced ETL framework solves the problem in which many organizations prefer to turn to in-house development to perform ETL tasks, because the new framework is extendable and any extra component to suit special business needs can be merged easily to the framework rather than rebuilding the ETL from the scratch.				(4)	
S06	Using the enhanced ETL framework solves the problem of lack of extensibility, because the components of the enhanced framework are distributable, inter-operable, and loosely-coupled to each others. So, the framework can be extended without complications.				(4)	
S07	Using the enhanced ETL framework solves the problem of environment compatibility, because all the compatibility requirements can be configured once on the shared application server which the components will be deployed to.				(4)	
S08	Using the enhanced ETL framework solves the problem of including ETL as a part of a web portal because the ETL components are built based on SOA which guarantees the portability of the components, and it follows web services standards for communication among components.			(3)		

Section 4: Advantages of the restructured ETL framework over the traditional one

For each of the following statements, please circle the number that indicates the extent to which you agree or disagree.

		Strongly Disagree ← → Strongly Agree				
		1	2	3	4	5
AD1	<i>Interoperability:</i> The interaction between clients (ETL administrators) and loosely-coupled ETL services has widespread interoperability. It is required that clients and services can communicate and understand each other no matter what platform they run on. This target is met because clients and ETL services have a standard way of communication among each other. In addition, interoperability provides consistency across platforms, systems, and languages for the ETL framework components.			(3)		
AD2	<i>Flexibility and extensibility:</i> Loosely-coupled services in ETL framework are typically more flexible than tightly-coupled ETL framework. In the traditional framework, the ETL components are tightly-coupled, share semantics, libraries, and often share state. This makes it difficult to evolve the application to keep up with changing business requirements. The loosely-coupled, document-based, and asynchronous natures of loosely-coupled services allow applications to be flexible, and easy to evolve with requirement changes. Therefore, the new ETL framework can have extra components without complexities. Based on that, it can be extended in the future by adding any extra component to suit enterprises' new business requirements.			(3)		
AD3	<i>Reusability:</i> In case that the new ETL framework is followed, ETL developers in data warehouse industry can reuse the code of an existing ETL component developed by any other ETL provider, to meet new ETL business requirements. Reusing functionality that already exists outside or inside an enterprise instead of developing code that reproduces those functions, can result in a big savings in application development cost and time.				(4)	
AD4	<i>Scalability:</i> Because ETL framework services are loosely-coupled, applications that follow the framework standards can scale easily, at least easier than applications that follow more tightly-coupled frameworks. That is because there are few dependencies between the requesting application and the services it uses. That enables the service to handle more requests because it does not need to handle lots of communication overhead like tightly-coupled components.			(3)		
AD5	<i>Cost Efficiency:</i> An ETL framework based on SOA, results in cheaper solutions because the integration of clients and services does not require heavy analysis and unique code of customized solutions. Furthermore, because services in the enhanced framework are loosely-coupled, applications that use these services are cheaper to maintain and easier to extend than customized solutions. Furthermore, administration, communication and maintenance effort will be less, because all the administrators share the same component which can be configured once, and each administrator can follow the common manual to use the				(4)	

	component for his special needs. In addition, the solution of redundancy and licenses problem supports cheaper solutions.				
AD6	<i>Ease of use</i> : Using the enhanced ETL framework solves the problem of usability difficulty, because the ETL administrator (user) needs to learn how to use only the component he needs, and not the whole ETL tool.	1	2	3	4 5
AD7	<i>Compatibility</i> : Using the enhanced ETL framework solves the problem of compatibility, because all the compatibility requirements can be configured once on the shared application server which the components are deployed to.	1	2	3	4 5
AD8	<i>Portability</i> : Using the enhanced ETL framework solves the problem of including ETL as a part of a web portal because the ETL components are built based on SOA which guarantees the portability of the components, and it follows web services standards for communication among components.	1	2	3	4 5

Kindly give your advices and comments if any:

Thank You

EXPERT OPINION INTERVIEW AND QUESTIONNAIRE

Name: Muhammad Zaini Zulkefli
Email: zaini@tmrnd.com.my
Hand-Phone: 019.4980767
Telephone:
Organization Name: TM R&D, Cyberjaya
Organization Website:
Organization address:

Section 1: Information about you and your organization.

1. Which best describes your current position?
 - ☐ DW Analyst
 - ☐ DW Manager
 - ☐ DW Specialist (architect/engineer/developer)
 - ☐ Project Leader
 - ☐ IT Consultant
 - ☒ DW Administrator
 - ☐ Software/Web developer
 - ☐ System Architect/Analyst
2. How long have you been in your current position?
 - ☐ Less than 1 year
 - ☒ 3 to 6 years
 - ☐ 1 to 3 years
 - ☐ More than 6 years
3. What is the current status of DW activities in your organization?
 - ☒ Develop and implement DW projects for organizations
 - ☐ Consulting DW projects
 - ☐ Do general Web applications
 - ☐ Just started doing DW projects
 - ☐ Planning to enter DW field in the future
 - ☐ Develop and enhance DW tools
4. How long have DW, SOA, or Web services in your organization been in existence?
 - ☐ Less than 1 year
 - ☐ 3 to 6 years
 - ☐ Not available.
 - ☐ 1 to 3 years
 - ☒ More than 6 years
5. How long have you worked for this organization?
 - ☐ Less than 2 years
 - ☐ 5 to 10 years
 - ☒ 2 to 5 years
 - ☐ More than 10 years
6. What is the percentage of including web services or SOA in your organization projects?
 - ☐ Not used at all
 - ☐ Less than 10%
 - ☐ From 10% to 40%
 - ☒ From 40% to 70%
 - ☐ From 70% to 90%
 - ☐ From 90% to 100%

Section 2: Problems of the current ETL framework

For each of the following statement, please circle the number that indicates the extent to which you agree or disagree

		Strongly Disagree ← → Strongly Agree				
PR1	If DW extracts its data from many different locations, and each location administrator needs to push the data from his side. Furthermore, you use a licensed ETL tool for each location. Therefore, the budget of the project will be higher because of the number of licenses needed.	1	2	3	4	5
PR2	Transformation and Loading features are redundant at the Extraction location because only Extraction is needed at data source locations. In addition, those two features would be necessary only in the location of the data warehouse destination. That leads to redundancy problem i.e. we pay a license for all ETL features and use some of them in some data location.	1	2	3	4	5
PR3	In the current ETL framework, including all the features as tightly-coupled components in one software; decreases the ease of use, because it is easier for an ETL user to learn how to use only the needed functionality rather than learning all ETL functionalities.	1	2	3	4	5
PR4	Managing many ETL tools in many data source locations for the same project to extract data from many different sources; needs extra administration, communication and maintenance effort, because the administrators are often different persons from one location to another and they could use different ETL tools and do different configurations to those tools.	1	2	3	4	5
PR5	Sometimes due to the complexity, long learning curve of the available ETL tools, and difficulty to achieve some extensibility in terms of additional functionalities; some organizations prefer to turn to in-house development to perform ETL tasks, which increases the project effort.	1	2	3	4	5
PR6	In the current ETL framework, it is not easy to extend ETL components by adding other components; to suite additional special business needs.	1	2	3	4	5
PR7	Not all ETL tools are compatible with all server environments, and each vendor's tool runs on specific environment including specific operating system and other requirements. That is considered sometimes an impediment for some customers.	1	2	3	4	5
PR8	In the current ETL framework, there are many impediments to include ETL as a part of a complete portal that manages the DW project, because of lack of standards in the current ETL framework to communicate with other components of the portal.	1	2	3	4	5

Section 3: Solutions for the problems after restructuring the current ETL framework based on SOA

For each of the following statement, please circle the number that indicates the extent to which you agree or disagree

Strongly Disagree ← Strongly Agree

SO1	Using the enhanced ETL framework solves the problem of licenses needed for each data source location, because ETL components can be deployed once on application server(s). Then, the same deployed version can be used remotely by all parties of the DW project.	1	2	3	4	5
SO2	Using the enhanced ETL framework solves the problem of component redundancy, since only one common component is available in the deployed version. Then, the needed component can be invoked remotely by the administrator and it is the same component for other administrators in other locations.	1	2	3	4	5
SO3	Using the enhanced ETL framework solves the problem of difficult usability, because ETL administrator needs to learn how to use only the component he needs, and not the whole ETL tool.	1	2	3	4	5
SO4	Using the enhanced ETL framework solves the problem of extra administration, communication and maintenance effort, because all the administrators share the same component which can be configured once, and each administrator can follow the common manual to use the component for his special needs.	1	2	3	4	5
SO5	Using the enhanced ETL framework solves the problem in which many organizations prefer to turn to in-house development to perform ETL tasks, because the new framework is extendable and any extra component to suit special business needs can be merged easily to the framework rather than rebuilding the ETL from the scratch.	1	2	3	4	5
SO6	Using the enhanced ETL framework solves the problem of lack of extensibility, because the components of the enhanced framework are distributable, inter-operable, and loosely-coupled to each others. So, the framework can be extended without complications.	1	2	3	4	5
SO7	Using the enhanced ETL framework solves the problem of environment compatibility, because all the compatibility requirements can be configured once on the shared application server which the components will be deployed to.	1	2	3	4	5
SO8	Using the enhanced ETL framework solves the problem of including ETL as a part of a web portal because the ETL components are built based on SOA which guarantees the portability of the components, and it follows web services standards for communication among components.	1	2	3	4	5

Section 4: Advantages of the restructured ETL framework over the traditional one

For each of the following statement, please circle the number that indicates the extent to which you agree or disagree

		Strongly Disagree	←	→	Strongly Agree
AD1	<i>Interoperability:</i> The interaction between clients (ETL administrators) and loosely-coupled ETL services has widespread interoperability. It is required that clients and services can communicate and understand each other no matter what platform they run on. This target is met because clients and ETL services have a standard way of communication among each other. In addition, interoperability provides consistency across platforms, systems, and languages for the ETL framework components.	1	2	3	4 5
AD2	<i>Flexibility and extensibility:</i> Loosely-coupled services in ETL framework are typically more flexible than tightly-coupled ETL framework. In the traditional framework, the ETL components are tightly-coupled, share semantics, libraries, and often share state. This makes it difficult to evolve the application to keep up with changing business requirements. The loosely-coupled, document-based, and asynchronous natures of loosely-coupled services allow applications to be flexible, and easy to evolve with requirement changes. Therefore, the new ETL framework can have extra components without complexities. Based on that, it can be extended in the future by adding any extra component to suit enterprises' new business requirements.	1	2	3	4 5
AD3	<i>Reusability:</i> In case that the new ETL framework is followed, ETL developers in data warehouse industry can reuse the code of an existing ETL component developed by any other ETL provider, to meet new ETL business requirements. Reusing functionality that already exists outside or inside an enterprise instead of developing code that reproduces those functions; can result in a big savings in application development cost and time.	1	2	3	4 5
AD4	<i>Scalability:</i> Because ETL framework services are loosely-coupled, applications that follow the framework standards can scale easily, at least easier than applications that follow more tightly-coupled frameworks. That is because there are few dependencies between the requesting application and the services it uses. That enables the service to handle more requests because it does not need to handle lots of communication overhead like tightly-coupled components.	1	2	3	4 5
AD5	<i>Cost Efficiency:</i> An ETL framework based on SOA, results in cheaper solutions because the integration of clients and services does not require heavy analysis and unique code of customized solutions. Furthermore, because services in the enhanced framework are loosely-coupled, applications that use these services are cheaper to maintain and easier to extend than customized solutions. Furthermore, administration, communication and maintenance effort will be less, because all the administrators share the same component which can be configured once, and each administrator can follow the common manual to use the	1	2	3	4 5

	component for his special needs. In addition, the solution of redundancy and licenses problem supports cheaper solutions.					
AD6	<i>Ease of use:</i> Using the enhanced ETL framework solves the problem of usability difficulty, because the ETL administrator (user) needs to learn how to use only the component he needs, and not the whole ETL tool.	1	2	3	4	5
AD7	<i>Compatibility:</i> Using the enhanced ETL framework solves the problem of compatibility, because all the compatibility requirements can be configured once on the shared application server which the components are deployed to.	1	2	3	4	5
AD8	<i>Portability:</i> Using the enhanced ETL framework solves the problem of including ETL as a part of a web portal because the ETL components are built based on SOA which guarantees the portability of the components, and it follows web services standards for communication among components.	1	2	3	4	5

Kindly give your advices and comments if any:

Thank You

EXPERT OPINION INTERVIEW AND QUESTIONNAIRE

Name : OSAMA M.S HAMMOUDA

Address : 102, Al Khayma, Al Khayma, Dubai, U.A.E.

Mobile No. : 00971502544801

Home No. : 00971043382456, Fax : 234

Company Name : Emirates Industrial Gases Company (EIGC)

Company Website : www.eigc.ae

Company Address : Al Itihad Road, Deira, Dubai, Uae

Section 1: Information about you and your organization.

1. Which best describes your current position?
DW Analyst
DW Manager
DW Specialist (architect/engineer/developer)
Project Leader
IT Consultant
DW Administrator
Software/Web developer
System Architect/Analyst
2. How long have you been in your current position?
Less than 1 year
3 to 6 years
1 to 3 years
More than 6 years
3. What is the current status of DW activities in your organization?
Develop and implement DW projects for organizations
Consulting DW projects
Do general Web applications
Just started doing DW projects
Planning to enter DW field in the future
Develop and enhance DW tools
4. How long have DW, SOA, or Web services in your organization been in existence?
Less than 1 year
3 to 6 years
Not available
1 to 3 years
More than 6 years
5. How long have you worked for this organization?
Less than 2 years
5 to 10 years
2 to 5 years
More than 10 years
6. What is the percentage of including web services or SOA in your organization projects?
Not used at all
Less than 10%
From 10% to 40%
From 40% to 70%
From 70% to 90%
From 90% to 100%

Section 2: Problems of the current ETL framework

For each of the following statement, please circle the number that indicates the extent to which you agree or disagree.

		Strongly Disagree	1	2	3	4	5 Strongly Agree
PR1	If DW extracts its data from many different locations, and each location administrator needs to push the data from his side. Furthermore, you use a licensed ETL tool for each location. Therefore, the budget of the project will be higher because of the number of licenses needed.	1	2	3	4	5	
PR2	Transformation and Loading features are redundant at the Extraction location because only Extraction is needed at data source locations. In addition, those two features would be necessary only in the location of the data warehouse destination. That leads to redundancy problem i.e. we pay a license for all ETL features and use some of them in some data location.	1	2	3	4	5	
PR3	In the current ETL framework, including all the features as tightly-coupled components in one software, decreases the ease of use, because it is easier for an ETL user to learn how to use only the needed functionality rather than learning all ETL functionalities.	1	2	3	4	5	
PR4	Managing many ETL tools in many data source locations for the same project to extract data from many different sources; needs extra administration, communication and maintenance effort, because the administrators are often different persons from one location to another and they could use different ETL tools and do different configurations to those tools.	1	2	3	4	5	
PR5	Sometimes due to the complexity, long learning curve of the available ETL tools, and difficulty to achieve some extensibility in terms of additional functionalities, some organizations prefer to turn to in-house development to perform ETL tasks, which increases the project effort.	1	2	3	4	5	
PR6	In the current ETL framework, it is not easy to extend ETL components by adding other components; to suite additional special business needs.	1	2	3	4	5	
PR7	Not all ETL tools are compatible with all server environments, and each vendor's tool runs on specific environment including specific operating system and other requirements. That is considered sometimes an impediment for some customers.	1	2	3	4	5	
PR8	In the current ETL framework, there are many impediments to include ETL as a part of a complete portal that manages the DW project, because of lack of standards in the current ETL framework to communicate with other components of the portal.	1	2	3	4	5	

Section 3: Solutions for the problems after restructuring the current ETL framework based on SOA

For each of the following statements, please circle the number to indicate the extent to which you agree or disagree.

Strongly Disagree ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 Strongly Agree

SO1 Using the enhanced ETL framework solves the problem of licenses needed for each data source location, since ETL components can be deployed once on application servers so that the same deployed version can be used remotely by all parties on the Web project. 1 2 3 4 5

SO2 Using the enhanced ETL framework solves the problem of component redundancy, since only one common component is available in the deployed version. Then the needed component can be invoked remotely by the administrator and it is the same component for other administrators in other locations. 1 2 3 4 5

SO3 Using the enhanced ETL framework solves the problem of difficult usability, because ETL administrator needs to learn how to use only the component he needs and not the whole ETL tool. 1 2 3 4 5

SO4 Using the enhanced ETL framework solves the problem of extra administration, communication and maintenance effort, because all the administrators share the same component which can be configured once, and each administrator can follow the common manual to use the component for his special needs. 1 2 3 4 5

SO5 Using the enhanced ETL framework solves the problem in which many organizations prefer to turn to in-house development to perform ETL tasks, because the new framework is extensible, and any extra component to suit special business needs can be merged easily to the framework rather than rebuilding the ETL from the scratch. 1 2 3 4 5

SO6 Using the enhanced ETL framework solves the problem of lack of extensibility, because the components of the enhanced framework are distributable, inter-operable, and loosely coupled to each others. So, the framework can be extended without complications. 1 2 3 4 5

SO7 Using the enhanced ETL framework solves the problem of environment compatibility, because all the compatibility requirements can be configured once on the shared application server which the components will be deployed to. 1 2 3 4 5

SO8 Using the enhanced ETL framework solves the problem of including ETL as a part of a web portal because the ETL components are built based on SOA which guarantees the portability of the components, and it follows web services standards for communication among components. 1 2 3 4 5

Section 4: Advantages of the restructured ETL framework over the traditional one

For each of the following statement, please circle the number that indicates the extent to which you agree or disagree.

Strongly Disagree ← → Strongly Agree

- | | | |
|-----|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------|
| AD1 | <p><i>Interoperability:</i> The interaction between clients (ETL administrators) and loosely-coupled ETL services has widespread interoperability. It is required that clients and services can communicate and understand each other no matter what platform they run on. This target is met because clients and ETL services have a standard way of communication among each other. In addition, interoperability provides consistency across platforms, systems, and languages for the ETL framework components.</p> | <p>1 2 3 4 5</p> <p style="margin-left: 100px;">(3)</p> |
| AD2 | <p><i>Flexibility and extensibility:</i> Loosely coupled services in ETL framework are typically more flexible than tightly-coupled ETL framework. In the traditional framework, the ETL components are tightly-coupled, share semantics, libraries, and often share state. This makes it difficult to evolve the application to keep up with changing business requirements. The loosely-coupled, document-based, and asynchronous natures of loosely-coupled services allow applications to be flexible, and easy to evolve with requirement changes. Therefore, the new ETL framework can have extra components without complexities. Based on that, it can be extended in the future by adding any extra component to suit enterprises' new business requirements.</p> | <p>1 2 3 4 5</p> <p style="margin-left: 100px;">(3)</p> |
| AD3 | <p><i>Reusability:</i> In case that the new ETL framework is followed, ETL developers in data warehouse industry can reuse the code of an existing ETL component developed by any other ETL provider, to meet new ETL business requirements. Reusing functionality that already exists outside or inside an enterprise instead of developing code that reproduces those functions, can result in a big savings in application development cost and time.</p> | <p>1 2 3 4 5</p> <p style="margin-left: 100px;">(4)</p> |
| AD4 | <p><i>Scalability:</i> Because ETL framework services are loosely-coupled, applications that follow the framework standards can scale easily, at least easier than applications that follow more tightly-coupled frameworks. That is because there are few dependencies between the requesting application and the services it uses. That enables the service to handle more requests because it does not need to handle lots of communication overhead like tightly-coupled components.</p> | <p>1 2 3 4 5</p> <p style="margin-left: 100px;">(4)</p> |
| AD5 | <p><i>Cost Efficiency:</i> An ETL framework based on SOA, results in cheaper solutions because the integration of client and services does not require heavy analysis and unique code of customized solutions. Furthermore, because services in the enhanced framework are loosely-coupled, applications that use these services are cheaper to maintain and easier to extend than customized solutions. Furthermore, administration, communication and maintenance effort will be less, because all the administrators share the same component which can be configured once, and each administrator can follow the common manual to use the</p> | <p>1 2 3 4 5</p> <p style="margin-left: 100px;">(4)</p> |

component for his special needs. In addition, the solution of redundancy and increases problem, improves cheaper solutions.

AD6: *Flexibility*. Using the enhanced E-IT framework solves the problem of flexibility difficulty, because the E-IT components can be needed to help how to use only the component he wants, and not the whole E-IT tool.

AD7: *Compatibility*. Using the enhanced E-IT framework solves the problem of compatibility, because all the compatibility requirements can be configured once on the shared application server which the components are deployed to.

AD8: *Portability*. Using the enhanced E-IT framework solves the problem of including E-IT as a part of a web portal because the E-IT components are built based on SOA which guarantees the portability of the components, and it follows web services standard for communication among components.

Kindly give your advices and comments if any:

Thank You

Expert Opinion Interview and Questionnaire

Name: Mohammad Skaik

Email: info@pit.ps

Telephone: +972 08-2861772

Hand-phone: +970 0599773819

Organization Name: Professionals Information Technology (PIT) Co. Ltd.

Organization website: <http://www.pit.ps/>

Organization address: 5 Fl.,Kadhem Bld.,Omar Al-Mokhtar St., Al-remal, Gaza,
Palestine.

Section 1: Information about you and your organization.

7. Which best describes your current position?
- ☐ DW Analyst
 - ☐ **DW Manager**
 - ☐ DW Specialist (architect/engineer/developer)
 - ☐ Project Leader
 - ☐ IT Consultant
 - ☐ DW Administrator
 - ☐ Software/Web developer
 - ☐ System Architect/Analyst
8. How long have you been in your current position?
- ☐ Less than 1 year
 - ☐ **3 to 6 years**
 - ☐ 1 to 3 years
 - ☐ More than 6 years
9. What is the current status of DW activities in your organization?
- ☐ Develop and implement DW projects for organizations
 - ☐ **Consulting DW projects**
 - ☐ Do general Web applications
 - ☐ Just started doing DW projects
 - ☐ Planning to enter DW field in the future
 - ☐ Develop and enhance DW tools
10. How long have DW, SOA, or Web services in your organization been in existence?
- ☐ Less than 1 year
 - ☐ **3 to 6 years**
 - ☐ Not available.
 - ☐ 1 to 3 years
 - ☐ More than 6 years
11. How long have you worked for this organization?
- ☐ Less than 2 years
 - ☐ **5 to 10 years**
 - ☐ 2 to 5 years
 - ☐ More than 10 years
12. What is the percentage of including web services or SOA in your organization projects?
- ☐ Not used at all
 - ☐ Less than 10%
 - ☐ From 10% to 40%
 - ☐ From 40% to 70%
 - ☐ **From 70% to 90%**
 - ☐ From 90% to 100%


Section 2: Problems of the current ETL framework

For each of the following statement, please circle the number that indicates the extent to which you agree or disagree.

		Strongly Disagree \longleftrightarrow Strongly Agree				
PR1	If DW extracts its data from many different locations, and each location administrator needs to push the data from his side. Furthermore, you use a licensed ETL tool for each location. Therefore, the budget of the project will be higher because of the number of licenses needed.	1	2	3	4	5
PR2	Transformation and Loading features are redundant at the Extraction location because only Extraction is needed at data source locations. In addition, those two features would be necessary only in the location of the data warehouse destination. That leads to redundancy problem i.e. we pay a license for all ETL features and use some of them in some data location.	1	2	3	4	5
PR3	In the current ETL framework, including all the features as tightly-coupled components in one software; decreases the ease of use, because it is easier for an ETL user to learn how to use only the needed functionality rather than learning all ETL functionalities.	1	2	3	4	5
PR4	Managing many ETL tools in many data source locations for the same project to extract data from many different sources; needs extra administration, communication and maintenance effort, because the administrators are often different persons from one location to another and they could use different ETL tools and do different configurations to those tools.	1	2	3	4	5
PR5	Sometimes due to the complexity, long learning curve of the available ETL tools, and difficulty to achieve some extensibility in terms of additional functionalities; some organizations prefer to turn to in-house development to perform ETL tasks, which increases the project effort.	1	2	3	4	5
PR6	In the current ETL framework, it is not easy to extend ETL components by adding other components; to suite additional special business needs.	1	2	3	4	5
PR7	Not all ETL tools are compatible with all server environments, and each vendor's tool runs on specific environment including specific operating system and other requirements. That is considered sometimes an impediment for some customers.	1	2	3	4	5
PR8	In the current ETL framework, there are many impediments to include ETL as a part of a complete portal that manages the DW project, because of lack of standards in the current ETL framework to communicate with other components of the portal.	1	2	3	4	5

Section 3: Solutions for the problems after restructuring the current ETL framework based on SOA

For each of the following statement, please circle the number that indicates the extent to which you agree or disagree.

Strongly Disagree  Strongly Agree

SO1	Using the enhanced ETL framework solves the problem of licenses needed for each data source location, because ETL components can be deployed once on application server(s). Then, the same deployed version can be used remotely by all parties of the DW project.	1	2	3	4	5
SO2	Using the enhanced ETL framework solves the problem of component redundancy, since only one common component is available in the deployed version. Then, the needed component can be invoked remotely by the administrator and it is the same component for other administrators in other locations.	1	2	3	4	5
SO3	Using the enhanced ETL framework solves the problem of difficult usability, because ETL administrator needs to learn how to use only the component he needs, and not the whole ETL tool.	1	2	3	4	5
SO4	Using the enhanced ETL framework solves the problem of extra administration, communication and maintenance effort, because all the administrators share the same component which can be configured once, and each administrator can follow the common manual to use the component for his special needs.	1	2	3	4	5
SO5	Using the enhanced ETL framework solves the problem in which many organizations prefer to turn to in-house development to perform ETL tasks, because the new framework is extendable and any extra component to suit special business needs can be merged easily to the framework rather than rebuilding the ETL from the scratch.	1	2	3	4	5
SO6	Using the enhanced ETL framework solves the problem of lack of extensibility, because the components of the enhanced framework are distributable, inter-operable, and loosely-coupled to each others. So, the framework can be extended without complications.	1	2	3	4	5
SO7	Using the enhanced ETL framework solves the problem of environment compatibility, because all the compatibility requirements can be configured once on the shared application server which the components will be deployed to.	1	2	3	4	5
SO8	Using the enhanced ETL framework solves the problem of including ETL as a part of a web portal because the ETL components are built based on SOA which guarantees the portability of the components, and it follows web services standards for communication among components.	1	2	3	4	5

Section 4: Advantages of the restructured ETL framework over the traditional one

For each of the following statement, please circle the number that indicates the extent to which you agree or disagree.

		Strongly Disagree ← → Strongly Agree				
AD1	<i>Interoperability:</i> The interaction between clients (ETL administrators) and loosely-coupled ETL services has widespread interoperability. It is required that clients and services can communicate and understand each other no matter what platform they run on. This target is met because clients and ETL services have a standard way of communication among each other. In addition, interoperability provides consistency across platforms, systems, and languages for the ETL framework components.	1	2	3	4	5
AD2	<i>Flexibility and extensibility:</i> Loosely-coupled services in ETL framework are typically more flexible than tightly-coupled ETL framework. In the traditional framework, the ETL components are tightly-coupled, share semantics, libraries, and often share state. This makes it difficult to evolve the application to keep up with changing business requirements. The loosely-coupled, document-based, and asynchronous natures of loosely-coupled services allow applications to be flexible, and easy to evolve with requirement changes. Therefore, the new ETL framework can have extra components without complexities. Based on that, it can be extended in the future by adding any extra component to suit enterprises' new business requirements.	1	2	3	4	5
AD3	<i>Reusability:</i> In case that the new ETL framework is followed, ETL developers in data warehouse industry can reuse the code of an existing ETL component developed by any other ETL provider; to meet new ETL business requirements. Reusing functionality that already exists outside or inside an enterprise instead of developing code that reproduces those functions; can result in a big savings in application development cost and time.	1	2	3	4	5
AD4	<i>Scalability:</i> Because ETL framework services are loosely-coupled, applications that follow the framework standards can scale easily, at least easier than applications that follow more tightly-coupled frameworks. That is because there are few dependencies between the requesting application and the services it uses. That enables the service to handle more requests because it does not need to handle lots of communication overhead like tightly-coupled components.	1	2	3	4	5
AD5	<i>Cost Efficiency:</i> An ETL framework based on SOA, results in cheaper solutions because the integration of clients and services does not require heavy analysis and unique code of customized solutions. Furthermore, because services in the enhanced framework are loosely-coupled, applications that use these services are cheaper to maintain and easier to	1	2	3	4	5

	extend than customized solutions. Furthermore, administration, communication and maintenance effort will be less, because all the administrators share the same component which can be configured once, and each administrator can follow the common manual to use the component for his special needs. In addition, the solution of redundancy and licenses problem supports cheaper solutions.					
AD6	<i>Ease of use:</i> Using the enhanced ETL framework solves the problem of usability difficulty, because the ETL administrator (user) needs to learn how to use only the component he needs, and not the whole ETL tool.	1	2	3	4	5
AD7	<i>Compatibility:</i> Using the enhanced ETL framework solves the problem of compatibility, because all the compatibility requirements can be configured once on the shared application server which the components are deployed to.	1	2	3	4	5
AD8	<i>Portability:</i> Using the enhanced ETL framework solves the problem of including ETL as a part of a web portal because the ETL components are built based on SOA which guarantees the portability of the components, and it follows web services standards for communication among components.	1	2	3	4	5

Kindly give your advices and comments if any:

The idea seems promising and worth investigating. Good luck.

Thank You

Appendix D

Case Studies User Manual and Parts of the Source Code

D.1 Case Study Mnuual

This simple user manual is written to help case study respondents to follow certain steps in applying the SOA-based ETL prototype to execute ETL processes. The flowchart of Figure A.1 presents the main operations of SOA-based ETL prototype and the flow of the operations (processes). Operations can be executed separately, and some operations are dependent on other operations. For example, if there is a problem in executing the Extract operation, other ETL operations won't be executed because Extraction is the starting operation of the SOA-based ETL prototype. Table D.1 describes the necessary and optional operations.

Table D.1: Necessary and Optional Operations.

Operation	Status
Extract	Mandatory
Transform	Optional
Classification	Optional
Load	Mandatory

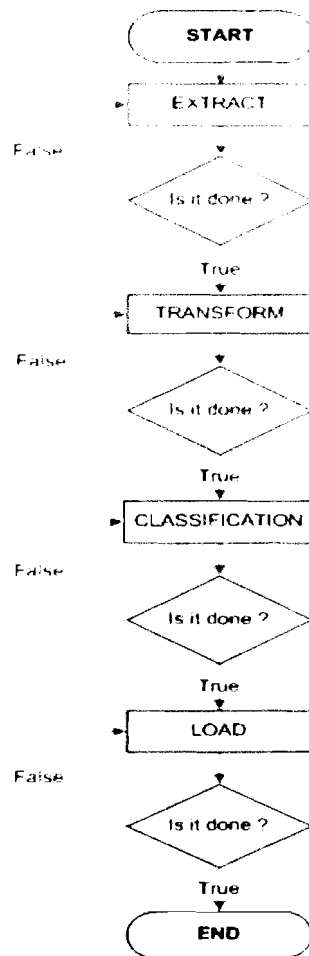


Figure D.1: SOA-based ETL Flowchart

D.1.1 Pre-installation requirements

To be able to use the SOA-based ETL prototype you have to install:

- Java SE Development Kit (JDK) 6.0 or higher.
- MySQL database 5.0 or higher in addition to the DBMS available at your organization.
- Apache Tomcat web Server 5.9 or higher.

- Any web browser.

D.1.2 ConFigure Database Connection

You must configure database connection through modifying the *config.txt* file that has the database connection variables, which is located in WEB-INF folder (see example in Figure D.2)

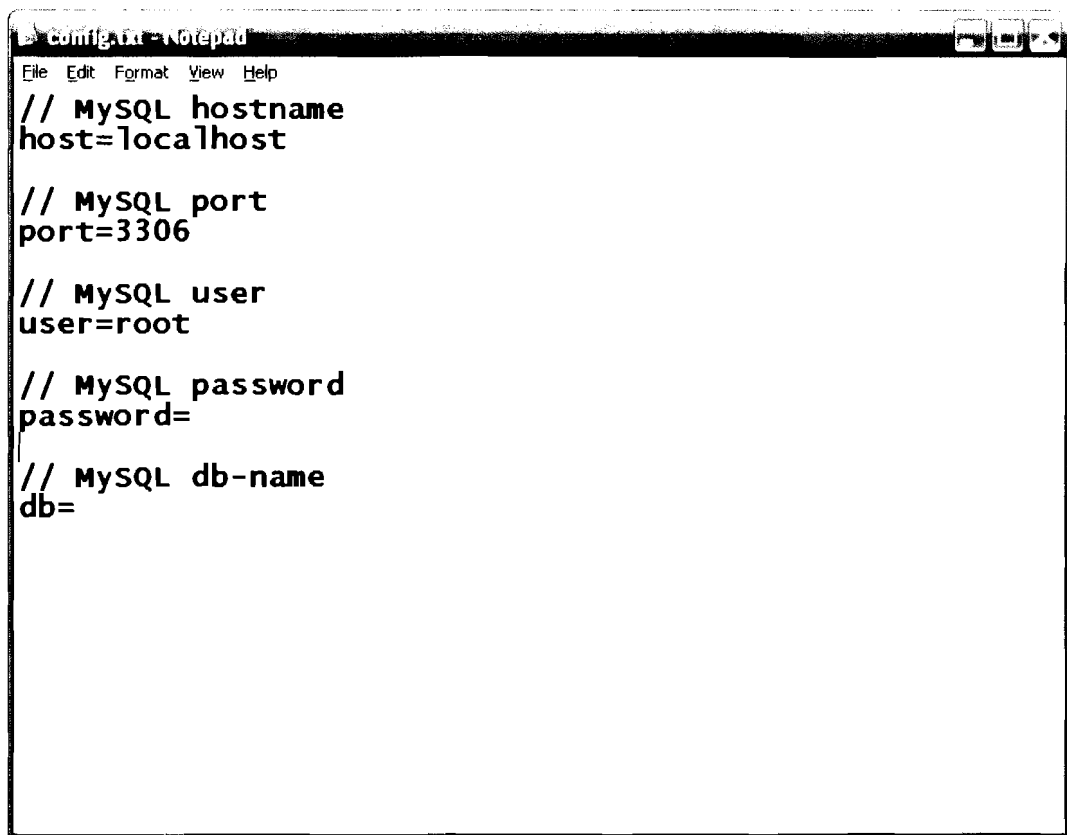


Figure D.2: Database connection variables

D.1.3 Extract Operation

For the *extract.jsp* file that is located in *jsp* folder of the Webapps main folder, the following steps are supposed to be done to configure the extract operation:

a. Create extract temp storage

Specify the database name that is stored the extracted data (see example: step 1 at Figure D.3).

b. Create fact Table

Build a query that will create the fact Table and extract data from the main data sources to the extract temp storage database (see example: step 2 at Figure D.3).

c. Create Dimension Tables

Build queries that will create the dimension (look up) tables and extract its data from the main data sources to the extractTempStorage database (see stage 3 at Figure D.3). This step is optional if the data of the dimension table is few records and you want to insert the data manually.

Note: If you would like to add a new extract method you can go to *ExtractData.java* file that is located in *\WEB-INF\classes*.

```

File Edit Search View Window Build Tools Database Window Help
extract.jsp

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<%
data.setJdbcExtractData(extract = new DataUtil().ExtractData(pageContext.getRequest().getRealPath() + WEB-INF/extract.jdbcConnect);

String sql;

    create temp table
    sql="create database extracttempstorage";
    extract.extractData(sql);

    extract data from temp storage and then store it in temp table
    sql = "create table extracttempstorage.temp as select t.TEST_ID,RESULT,t.TEST_DATE,t.TEST_IMG,t.
    " t.TEST_VIDEO,t.DISEASE_ID,t.MEDICINE_ID,p.Gender_ID,p.P_BIRTHDAY,p.CITY_ID "
    " from clinic.testoperations t,clinic.patients p where p.Patient_ID=t.Patient_ID ";
    extract.extractData(sql);

    dimension tables
    sql = "create table extracttempstorage.city as select * from clinic.city";
    extract.extractData(sql);
    sql = "create table extracttempstorage.disease as select * from clinic.disease";
    extract.extractData(sql);
    sql = "create table extracttempstorage.gender as select * from clinic.gender";
    extract.extractData(sql);
    sql = "create table extracttempstorage.medicines as select * from clinic.medicines";
    extract.extractData(sql);

extract.jdbcConnClose();
extract = null;
%>

<br><br><tr><td align=center><b><font size=14>Extract ...Done</font></td></tr>
<% response.setHeader("Refresh","3;URL=...index.html"); %>

```

Figure D.3: Extract Operation

D.1.4 Transform Operation

This is an optional component. You can skip it if your organization does not need it.

At *transform.jsp* file that is located in *jsp* folder, the following steps have to be done

to configure the transform operation:

a. *Create transform temp storage*

Specify the database name that is stored the tranformed data (see step 1 at Figure D.4).

b. *Create Fact Table*

Create empty fact table as a copy from Temp fact table that is located in the extractTempStorage database (see step 2 at Figure D.4).

c. *Alter Fact Table*

You can alter the empty fact table to change any field name or datatype to become compatible with the tranformed data (see step 3 at Figure D.4).

d. *Transform data*

Build a query to select extracted data from the extractTempStorage database, then transform the selected data such as calculate age from birthday by getAge() method. After that, insert transformed data into the Temp fact table that located in the transformTempStorage database (see step 4 at Figure D.4).

If you would like to build another transform method other than getAge() method, go to TransformData.java file that located in \WEB-INF\classes and add your new transform method, then use it.

e. Create Dimension Tables

Build queries that will create the dimension (look up) tables and copy its data from the extractTempStorage database to the transformTempStorage database (see stage 5 at Figure D.4).

Note: If you would like to add a new transform method you can go to TransformData.java file that is located in \WEB-INF\classes.

```

<!-- Import the classes, packages, and java.sql.*; for JDBC driver class -->
<%
data.etl.dwt.TransformData transform = new data.etl.dwt.TransformData(pageContext.getServiceContext().getServletPath(),
transform, true);

Vector v1,v2;
String sql,insert_sql;

//create temporary db
sql="create database TransformTempStorage";
transform.transformData(sql);

// create empty table into which tempstorage db
sql="create table TransformTempStorage.TEMP as select * from extractTempStorage.TEMP where 1=0";
transform.transformData(sql);

// alter the new table fields name and datatype
sql="alter table TransformTempStorage.TEMP change f_BIRTHDAY f_AGE varchar(100)";
transform.transformData(sql);

sql = "select t.TESTO_RESULT,t.TESTO_IMG,t.TESTO_VIDEO,t.DISEASE_ID,t.MEDICINE_ID,t.Gender_ID, " +
      "t.f_BIRTHDAY,t.CITY_ID,t.TESTO_DATE from extractTempStorage.TEMP t";

v1=transform.jdbcMultipleRowQuery(sql);
for(int i=0;i<v1.size();i++){
    v2 = (Vector)v1.elementAt(i);

    insert_sql="insert into TransformTempStorage.TEMP (TESTO_RESULT,TESTO_IMG,TESTO_VIDEO,DISEASE_ID,MEDICINE_ID," +
    " Gender_ID,f_AGE,CITY_ID,TESTO_DATE) VALUES (" +v2.get(0)+"'," +v2.get(1)+"'," +v2.get(2)+"'," +v2.get(3)+"'," +
    " "+v2.get(4)+"'," +v2.get(5)+"'," +transform.getAge((String)v2.get(6))+"'," +v2.get(7)+"'," +v2.get(8)+"")";

    transform.transformData(insert_sql);
}

// dimension Table
sql = "create table TransformTempStorage.city as select * from extractTempStorage.city";
transform.transformData(sql);
sql = "create table TransformTempStorage.disease as select * from extractTempStorage.disease";
transform.transformData(sql);
sql = "create table TransformTempStorage.gender as select * from extractTempStorage.gender";
transform.transformData(sql);
sql = "create table TransformTempStorage.medicines as select * from extractTempStorage.medicines";
transform.transformData(sql);

transform.jdbcConnClose();
transform = null;
%>

<!--<br>align="center"><br>font-size=12px>Transform ...Done! font-size=12px>
<!-- response.setHeader("Refresh","300;URL=index.html"); -->

```

Figure D.4: Transform Operation

D.1.5 Classification Operation

This is an optional component added to the prototype to prove its extensibility. You can skip this component if your organization does not need it. At *classification.jsp* file that is located in *jsp* folder, there are few steps to configure the classification operation:

a. *Create classification temp storage*

Specify the database name that stores the classified data (see step 1 at Figure D.5.1).

b. *Create Fact Table*

Create empty fact tables as a copy from Temp fact table that is located in the transformTempStorage database, you can create a number of fact tables according to the classification of data (see step 2 at Figure D.5.1).

c. *Classify data*

First of all, build a query to select transformed data from the transformTempStorage database (see step 3 at Figure D.5.2), and then set variables with the retrieved data to prepare it for classification (see step 5.1 at Figure D.5.2). In this classification operation we classify data according its datatype, such as which row has either image or video data, and which row does not have any of them. After that, insert classified data into three different fact tables (see step 3.2 at Figure D.5.2). Here, we classify data by another condition, it is the date of data, such as we classify data that was created before 1970 and

insert it into time1 fact table. The *getYear()* method returns the year from the date; to use year in classification operation (see step 3.3 at Figure D.5.2).

d. *Create Dimension Tables*

Build queries that will create the dimension (look up) tables and copy its data from the transformTempStorage database to the classification database (see stage 4 at Figure D.5.2).

Note: If you would like to add a new classification method you can go to ClassificationData.java file, that is located in \WEB-INF\classes.



```
data.etl.dw.ClassificationData classification = new data.etl.dw.ClassificationData(pageContext.getServletContext().
classification.jdbcConnect());

Vector v1,v2;
String sql,insert,sql;

// create temporary if
sql="create database classification";
classification.classificationData(sql);

// create empty fact tables from TransformTempStorage;
sql="create table classification.type1 as select * from TransformTempStorage.TEMP where 1=0";
classification.classificationData(sql);
sql="create table classification.type2 as select * from TransformTempStorage.TEMP where 1=2";
classification.classificationData(sql);
sql="create table classification.type3 as select * from TransformTempStorage.TEMP where 1=0";
classification.classificationData(sql);
sql="create table classification.type4 as select * from TransformTempStorage.TEMP where 1=0";
classification.classificationData(sql);

sql="create table classification.time1 as select * from TransformTempStorage.TEMP where 1=2";
classification.classificationData(sql);
sql="create table classification.time2 as select * from TransformTempStorage.TEMP where 1=2";
classification.classificationData(sql);
sql="create table classification.time3 as select * from TransformTempStorage.TEMP where 1=2";
classification.classificationData(sql);
```

Figure D.5.1: Classification Operation

[illegible]

Figure D.5.2: Classification Operation

D.1.6 Load Operation

At *load.jsp* file that is located in *jsp* folder, there are few steps to configure the classification operation:

a. Create load database

Specify the database name that stores the final form of data (see step 1 at Figure D.6).

b. Create Fact Tables

Build queries that will create the fact tables and loads its data from the classification (or any other ETL functionality) database to the load database (see step 2 at Figure D.6).

c. Create Dimension Tables

Build queries that will create the dimension (*look up*) tables and load its data from the classification database to the load database (see stage 3 at Figure D.6).

Note: If you would like to add a new extract method you can go to *LoadData.java* file, that is located in *\WEB-INF\classes*.

```

File Edit Search View Project Build Tools Database Window Help
load.jsp
String sql;

sql = "create table loadE.type1 as select * from classification.type1";
load.loadData(sql);

sql = "create table loadE.type2 as select * from classification.type2";
load.loadData(sql);

sql = "create table loadE.type3 as select * from classification.type3";
load.loadData(sql);

sql = "create table loadE.type4 as select * from classification.type4";
load.loadData(sql);

sql = "create table loadE.type5 as select * from classification.type5";
load.loadData(sql);

sql = "create table loadE.type6 as select * from classification.type6";
load.loadData(sql);

sql = "create table loadE.type7 as select * from classification.type7";
load.loadData(sql);

sql = "create table loadE.type8 as select * from classification.type8";
load.loadData(sql);

load.close();

sql = "create table loadE.city as select * from classification.city";
load.loadData(sql);

sql = "create table loadE.disease as select * from classification.disease";
load.loadData(sql);

sql = "create table loadE.gender as select * from classification.gender";
load.loadData(sql);

sql = "create table loadE.medicines as select * from classification.medicines";
load.loadData(sql);

load.jdbcConnClose();
load = null;
}

```

Figure D.6: Load Operation

D.2 Parts of the Case Studies' Source Code

Table D.2. Main Class for Extraction Processes

```
/**
Author: Mohammed Awad
Developed for: SOA-based ETL prototype evaluation (Case studies)
Date Created: 5th Dec., 2011
Copyrighted © 2011, 2012. All rights reserved
*/
package data.etl.dw;

import java.sql.*;

import java.util.*;

import java.io.*;

public class ExtractData {

    private String host, port, user, password, db, url;

    private Connection conn;

    public ExtractData(String config) throws IOException {

        Properties myproperties = new Properties();

        FileInputStream in = new FileInputStream(config);

        myproperties.load(in);

        in.close();

        this.host = myproperties.getProperty("host");

        this.port = myproperties.getProperty("port");

        this.user = myproperties.getProperty("user");

        this.password = myproperties.getProperty("password");
```

```
this.db = myproperties.getProperty("db");

this.url = "jdbc:odbc:" + this.host + ":" + this.port + "/" + this.db;

}

public void jdbcConnect() {

    try {

        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

        this.conn = DriverManager.getConnection(this.url, this.user, this.password);

    } catch (Exception ex) {

    }

}

public void jdbcConClose() {

    try {

        this.conn.close();

    } catch (Exception ex) {

    }

}

public void extractData(String sql) {

    try {

        Statement extractStmt = this.conn.createStatement();

        extractStmt.executeUpdate(sql);

        extractStmt.close();

    } catch (SQLException ex) {
```

```

    }
}
}

```

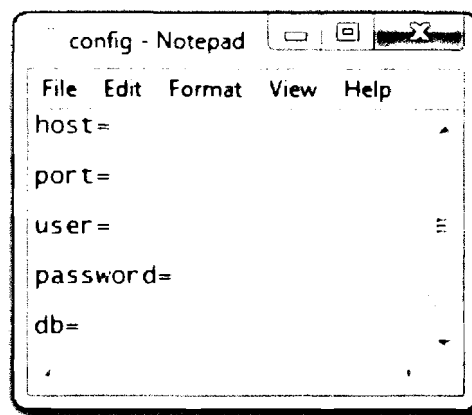


Figure D.7. Config File to hold the source database specifications

Table D.3: JSP Page that creates an object of the ExtractData class of Table D.2 for executing the Extraction process for PEC case study

```

<%@ page import="java.util.*,java.sql.*"%>

<%

data.etl.dw.ExtractData extract = new

data.etl.dw.ExtractData(pageContext.getServletContext().getRealPath("/WEB-

INF/config.txt"));

extract.jdbcConnect();

```

```

String sql;

//drop temporary DB

sql="drop database extracttempstorage";

extract.extractData(sql);

//create temporary DB

sql="create database extracttempstorage";

extract.extractData(sql);

//extract data from main source and then store it in TEMP Table

sql = "create Table extracttempstorage.temp as SELECT"+

"1","turbine_transaction_history.trans_month,"+

"turbine_transaction_history.trans_period_id"+

"FROM"+

"gpgc.turbine_transaction_history,"+

"gpgc.transaction_types"+

"WHERE"+

"turbine_transaction_history.trans_type_id = transaction_types.trans_type_id

AND"+

"transaction_types.trans_type_id = '2'";

extract.extractData(sql);

//dimension Tables

sql = "create Table extracttempstorage.period as select * from pgpc.period";

extract.extractData(sql);

```

```

sql = "create Table  extracttempstorage.months  as select * from pgpc.months";

extract.extractData(sql);

extract.jdbcConClose();

extract = null;

%>

<br><br><br><p align=center><b><font size=6>Extraction Done Successfully
...</font></b></p>

<%--  response.setHeader("Refresh","3;URL=../index.html");    --%>

```

Table D.4: Main Class for Transformation Processes for PEC case study

```

package data.etl.dw;

import java.sql.*;

import java.util.*;

import java.io.*;

public class TransformData {

    private String host;

    private String port;

    private String user;

    private String password;

    private String db;

```

```
private String url;

private Connection conn;

public TransformData(String config2) throws IOException {

    Properties myproperties = new Properties();

    FileInputStream in = new FileInputStream(config2);

    myproperties.load(in);

    in.close();

    this.host = myproperties.getProperty("host");

    this.port = myproperties.getProperty("port");

    this.user = myproperties.getProperty("user");

    this.password = myproperties.getProperty("password");

    this.db = myproperties.getProperty("db");

    this.url = "jdbc:odbc:" + this.host + ":" + this.port + "/" + this.db;

}

public void jdbcConnect() {

    try {

        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

        this.conn = DriverManager.getConnection(this.url, this.user, this.password);

    } catch (SQLException ex) {

    } catch (ClassNotFoundException ex) {

    }

}
```

```
    } catch (java.lang.Exception ex) {  
  
    }  
}
```

```
public void jdbcConClose() {  
  
    try {  
  
        this.conn.close();  
  
    } catch (Exception ex) {  
  
    }  
}
```

```
public void transformData(String sqlInsert) throws SQLException {  
  
    Statement insertStmt = this.conn.createStatement();  
  
    insertStmt.executeUpdate(sqlInsert);  
  
    insertStmt.close();  
}
```

```
public Vector jdbcMultipleRowQuery(String sqlQuery) {  
  
    Vector v = new Vector();  
  
    try {  
  
        Statement queryStmt = this.conn.createStatement();  
  
        ResultSet rs = queryStmt.executeQuery(sqlQuery);  

```

```

        int colCount = rs.getMetaData().getColumnCount();

        String s;

        while (rs.next()) {

            Vector vec = new Vector();

            for (int i = 0; i < colCount; i++) {

                s = rs.getString(i + 1);

                vec.addElement(s);

            }

            v.addElement(vec);

        }

        rs.close();

        queryStmt.close();

    } catch (SQLException ex) {

    }

    return v;

}

//Transform English

public String toArabic(String englishUnit) {

    String arabicUnit;

    if englishUnit.endsWith("AM")

    {

        arabicUnit = englishUnit.replace("صباحا", "AM")

```

```

    } else if englishUnit.endsWith("PM") {

        arabicUnit = englishUnit.replace("مساء", "AM")

    }

    return arabicUnit;

}

}

```

Table D.5: JSP Page that creates an object of the TransformData class of Table D.4 for transforming time units from English to Arabic for PEC case study

```

<%@ page import="java.util.*,java.sql.*"%>

<%

data.etl.dw.TransformData transform = new

data.etl.dw.TransformData(pageContext.getServletContext().getRealPath("/WEB-

INF/config.txt"));

transform.jdbcConnect();

Vector v1,v2;

String sql,insert_sql;

//drop temporary DB

sql="drop database if exists TransformTempStorage";

transform.transformData(sql);

//create temporary DB

```

```

sql="create database TransformTempStorage";

transform.transformData(sql);

// create empty Table from extracttempstorage DB

sql="create Table TransformTempStorage.TEMP as select * from
extracttempstorage.TEMP where l=2";

transform.transformData(sql);

sql = "select p.period_id, p.start_time, p.end_time from extracttempstorage.period
p";

v1=transform.jdbcMultipleRowQuery(sql);

    for(int i=0;i<v1.size();i++){

        v2 = (Vector)v1.elementAt(i);

insert_sql="insert into TransformTempStorage.period(period_id, start_time,
end_time) VALUES
('"+v2.get(0)+"','"+transform.toArabic(v2.get(1))+"','"+transform.toArabic(v2.get(2))
+"')";

transform.transformData(insert_sql);

    }

transform.jdbcConClose();

transform = null;

%>

<br><br><br><p align=center><b><font size=6>Transformation Done Successfully
...</font></b></p>

```

```
<%-- response.setHeader("Refresh","3;URL=./index.html"); --%>
```

Table D.6: JSP Page that creates an object of the ExtractData class of Table D.2 for executing the Extraction process for LUCT case study

```
<%@ page import="java.util.*,java.sql.*"%>

<%

data.etl.dw.ExtractData extract = new

data.etl.dw.ExtractData(pageContext.getServletContext().getRealPath("/WEB-

INF/config.txt"));

extract.jdbcConnect();

String sql;

//drop temporary DB

sql="drop database extracttempstorage";

extract.extractData(sql);

//create temporary DB

sql="create database extracttempstorage";

extract.extractData(sql);

//extract data from main source and then store it in TEMP Table

"sql = "create Table extracttempstorage.temp as SELECT "+

"std__student_performance.id, "+
```

```

"std_student_performance.gender_id, "+
"std_student_performance.mark_id, "+
"std_student_performance.status_id, "+
"std_student_performance.final_id"+
"FROM"+
"std_student_performance , "+
"std_student_status
"Inner Join std_marks ON " = " , "+
"std_gender , "+
"std_finalexam , "+
"std_attendance"+
"WHERE"+
"std_student_performance.gender_id = std_gender.gender_id AND"+
"std_student_performance.mark_id = std_marks.mark_id AND"+
"std_student_performance.status_id = std_student_status.status_id AND"+
"std_student_performance.final_id = std_finalexam.final_id AND"+
"std_student_performance.attendance_id = std_attendance.attendance_id";
extract.extractData(sql);

//dimension Tables

sql = "create Table  extracttempstorage.gender as select * from luct.std_gender";
extract.extractData(sql);

sql = "create Table  extracttempstorage.marks as select * from luct.std_marks";

```

```

extract.extractData(sql);

sql = "create Table  extracttempstorage.status as select * from luct.std_status";

extract.extractData(sql);

sql = "create Table  extracttempstorage.finalexam as select * from
luct.std_finalexam";

extract.extractData(sql);

sql = "create Table  extracttempstorage.attendance as select * from
luct.std_attendance";

extract.extractData(sql);

extract.jdbcConClose();

extract = null;

%>

<br><br><br><p align=center><b><font size=6>Extraction Done Successfully
...</font></b></p>

```

Table D.7: JSP Page that creates an object of the ExtractData class of Table D.2 for executing the Extraction process for PIT case study

```

<%@ page import="java.util.*,java.sql.*"%>

<%

```

```
data.etl.dw.ExtractData extract = new  
  
data.etl.dw.ExtractData(pageContext.getServletContext().getRealPath("/WEB-  
INF/config.txt"));  
  
extract.jdbcConnect();  
  
String sql;  
  
//drop temporary DB  
  
sql="drop database extracttempstorage";  
  
extract.extractData(sql);  
  
//create temporary DB  
  
sql="create database extracttempstorage";  
  
extract.extractData(sql);  
  
//extract data from main source and then store it in TEMP Table  
  
sql = "create Table extracttempstorage.temp as SELECT"+  
"tr_trainer_details.trainer_id, "+  
"tr_trainer_evaluation.performance_id"+  
"FROM"+  
"tr_performance_level , "+  
"tr_trainer_dependents , "+  
"tr_trainer_details , "+  
"tr_trainer_evaluation"+  
"WHERE"+  
"tr_trainer_evaluation.trainer_id = tr_trainer_details.trainer_id AND"+
```

```
"tr_performance_level.performance_id = "+ "tr_trainer_evaluation.performance_id";
extract.extractData(sql);"

//dimension Tables

sql = "create Table  extracttempstorage. trainer_details as select"+
"tr_trainer_details.trainer_id, "+
"tr_trainer_details.course_trained from pit.trainer_details";
extract.extractData(sql);

sql = "create Table  extracttempstorage.performance_level  as select"+
"tr_performance_level.performance_id, "+
"tr_performance_level.performance_desc from pit.tr_performance_level";
extract.extractData(sql);

extract.jdbcConClose();

extract = null;

%>

<br><br><br><p align=center><b><font size=6>Extraction Done Successfully
...</font></b></p>
```
