

Assessing and Analysing Data Quality in Service Oriented Architectures Developing a Data Quality Process

Plamen Petkov

A dissertation submitted in fulfilment of the requirements for the award of
Doctor of Philosophy (Ph.D.)



Dublin City University
School of Computing

Supervisor: *Dr. Markus Helfert*

July 2016

Declaration

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the award of Doctor of Philosophy is entirely my own work, that I have exercised reasonable care to ensure that the work is original, and does not to the best of my knowledge breach any law of copyright, and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work.

Signed: _____ (Candidate) ID No.: 11212753 Date: _____

Acknowledgements

Firstly, I would like to thank my supervisor Dr. Markus Helfert for the continuous support of my Ph.D. research, for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of this dissertation. I could not have imagined having a better advisor and mentor for my Ph.D. study.

I would like to express my special gratitude to my family and friends for their love and encouragement. I thank my mother, sister and father for affording me many opportunities throughout the years. Additionally, I would like to give my sincerest thanks to Chelsea Demastrie for her endless patience, commitment and support while writing this thesis.

BIG thanks also to all the members and past members of the Business Informatics Group(BIG), especially Karen, Giovanni, Lukasz, Martin, Rainny, Phelim and Owen. Your feedback and insights were most helpful and greatly appreciated.

Special thanks to my academic partners from University of Granada, Spain for providing me the opportunity to collaborate and practically evaluate my research with them. I thank Sandra Rodriguez-Valenzuela the stimulating discussions and enormous help.

Last but not least, I want to thank the Irish Research Council (IRC) and School of Computing at Dublin City University, who funded this research study and provided me an excellent working environment. It also provided me an exceptional opportunity to learn not only knowledge but also the Irish culture.

Table of Contents

LIST OF FIGURES.....	IV
LIST OF TABLES.....	VI
LIST OF ABBREVIATIONS	VII
ABSTRACT	VIII
CHAPTER 1 INTRODUCTION.....	1
1.1 SERVICE-ORIENTED APPROACH AS DESIGN STYLE	1
1.2 IMPORTANCE OF DATA QUALITY	2
1.3 RESEARCH PROBLEM AND OBJECTIVES	5
1.4 RESEARCH SCOPE AND REQUIREMENTS.....	7
1.5 THESIS ORGANIZATION	9
CHAPTER 2 LITERATURE REVIEW	12
2.1 OVERVIEW	12
2.2 FUNDAMENTAL CONCEPTS OF SERVICE ORIENTED ARCHITECTURES	13
2.2.1 <i>Drivers for SOA</i>	14
2.2.2 <i>SOA Design Patterns</i>	16
2.2.2.1 Key Principles affecting Data Quality	17
2.2.3 <i>Web Services Approach</i>	23
2.3 SERVICE ORIENTED RESEARCH.....	25
2.4 INFORMATION SYSTEMS AND DATA QUALITY	28
2.5 COMPARISON OF DATA QUALITY FRAMEWORKS	30
2.5.1 <i>Types of Information Systems and Application Contexts of DQ</i>	33
2.5.2 <i>Data Quality Dimensions</i>	39
2.5.2.1 Semiotic Theory	40
2.5.2.2 Quality of Service and Data Quality Dimensions.....	43
2.5.3 <i>Data Types</i>	45
2.6 SUMMARY OF DATA QUALITY METHODOLOGIES.....	48
2.7 RESEARCH THEMES	52
2.8 SUMMARY	54
CHAPTER 3 RESEARCH METHODOLOGY	56
3.1 INTRODUCTION	56
3.2 METHODOLOGICAL REQUIREMENTS	56

3.3	METHODOLOGY SELECTION	58
3.3.1	<i>Action Research</i>	58
3.3.2	<i>Grounded Theory</i>	59
3.3.3	<i>Method Engineering</i>	59
3.3.4	<i>Case Study Research</i>	60
3.3.5	<i>Design Science Research</i>	62
3.4	DESIGN SCIENCE AS RESEARCH METHODOLOGY.....	63
3.4.1	<i>Problem and Objective Identification</i>	65
3.4.2	<i>Design as an Artifact</i>	67
3.4.2.1	Business Process Modelling Notation as a Method Constructor	68
3.4.3	<i>Evaluation of the Artifact</i>	71
3.4.3.1	Collecting Evidences.....	73
3.4.4	<i>Communications of Research</i>	75
3.5	SUMMARY	75
CHAPTER 4	ASSESSING AND ANALYSING DATA QUALITY IN SERVICE ORIENTED ARCHITECTURES ...	77
4.1	INTRODUCTION	77
4.2	SOA ECOSYSTEM AND DATA QUALITY	77
4.3	MOTIVATIONAL CASE – DYNAMIC OPEN HOME AUTOMATION (DOHA) SMART MEETING ROOM	80
4.4	CONTEXTUAL DATA IN SOA.....	83
4.4.1	<i>Data Constraints</i>	87
4.5	TOTAL DATA QUALITY MANAGEMENT METHODOLOGY TDQM	89
4.6	DQ PROCESS IN SOA.....	90
4.6.1	<i>Building Information Profiles Stage</i>	92
4.6.2	<i>Preparation Stage – Defining “Quality” of Data</i>	97
4.6.3	<i>Execution Stage – Applying Data Quality Statements</i>	106
4.6.4	<i>Reporting Stage – Analysing Results</i>	111
4.7	METHOD FOR INCORPORATING DQ METHODOLOGY WITHIN SOA	112
4.8	SUMMARY	121
CHAPTER 5	INCORPORATING DATA QUALITY MECHANISM IN DYNAMIC HOME AUTOMATION ENVIRONMENT	125
5.1	INTRODUCTION	125
5.2	UBIQUITOUS COMPUTING	126
5.3	DYNAMIC OPEN HOME AUTOMATION (DOHA) SERVICE-ORIENTED PLATFORM	129
5.3.1	<i>Device Profile for Web Services</i>	131

5.3.2	<i>DOHA Service</i>	137
5.3.2.1	Service Composition Model	139
5.4	DOHA DQ SERVICE.....	143
5.5	SUMMARY	151
CHAPTER 6	ARTIFACT EVALUATION.....	153
6.1	INTRODUCTION	153
6.2	EVALUATION PROCESS.....	154
6.3	EVALUATION CRITERIA.....	158
6.4	FACTORS AFFECTING EVALUATION PROCESS	162
6.5	WORKSHOP LAYOUT	166
6.6	WORKSHOP PLANNING AND SETTINGS.....	167
6.7	CONDUCTING THE WORKSHOP.....	170
6.7.1	<i>Service/Information Profiling</i>	170
6.7.2	<i>Composing DQ Statements</i>	170
6.7.3	<i>Execution of DQ Statements</i>	173
6.7.4	<i>DQ Analysing and Reporting Stage</i>	174
6.8	RESULTS AND DISCUSSION.....	175
6.8.1	<i>Completeness</i>	176
6.8.2	<i>Consistency</i>	178
6.8.3	<i>Ease of use</i>	179
6.8.4	<i>Interface Design</i>	180
6.9	SUMMARY	180
CHAPTER 7	CONCLUSION.....	183
7.1	SUMMARY AND CONTRIBUTIONS	183
7.2	LIMITATIONS AND CRITICAL DISCUSSION	188
7.3	FUTURE WORK	189
PUBLICATION LIST	191
REFERENCES	192
APPENDIX A: SOA SURVEY	205
APPENDIX B: SUS SURVEY TOOL	209
APPENDIX C: CODAMOS ONTOLOGY	211
APPENDIX D: SERVICE CODE LISTING (PARTIAL)	212

List of Figures

FIGURE 1-1: SPREADING MALARIA DISEASE – A); AND POOR DATA QUALITY IN SOA – B)	4
FIGURE 1-2: OVERVIEW OF RESEARCH OBJECTIVE AND REQUIREMENTS	9
FIGURE 1-3: THESIS ORGANIZATION.....	10
FIGURE 2-1: ‘PUBLISH – DISCOVER – BINDING’ MECHANISM IN SOA	14
FIGURE 2-2: SOA BENEFITS FROM IT AND BUSINESS PERSPECTIVES.....	16
FIGURE 2-3: FOUNDATIONS OF SERVICE-ORIENTED PARADIGM.....	17
FIGURE 2-4: POINT-TO-POINT AND POINT-TO-HUB INTERACTION MODELS.....	21
FIGURE 2-5: ORCHESTRATION VERSUS CHOREOGRAPHY (PELTZ 2003)	23
FIGURE 2-6: SOAP MESSAGE FORMAT.....	24
FIGURE 2-7: WSDL DOCUMENT FORMAT	25
FIGURE 2-8: SERVICE ORIENTED ARCHITECTURES’ LAYER DIAGRAM	26
FIGURE 2-9: DAQUINCIS ARCHITECTURE (SCANNAPIECO, VIRGILLITO, AND MARCHETTI 2004)	38
FIGURE 2-10: POINTS OF VIEW FOR DEFINING INFORMATION/DATA QUALITY DIMENSIONS	40
FIGURE 2-11: TYPES OF DATA. DIFFERENT SYNTACTICAL REPRESENTATIONS OF THE SAME REAL WORLD OBJECT.....	45
FIGURE 2-12: EXTRACTION OF UNSTRUCTURED DATA - PROCESS AND METHODS	48
FIGURE 3-1: RESEARCH METHODOLOGY PROFILE	57
FIGURE 3-2: DESIGN SCIENCE PROCESS IN DQ RESEARCH.....	65
FIGURE 3-3: BPMN FLOW OBJECTS.....	70
FIGURE 3-4: BPMN CONNECTING AND ARTIFACT OBJECTS.....	71
FIGURE 4-1: SOA ECOSYSTEM AND DATA QUALITY. ROLES AND EXAMPLES.....	78
FIGURE 4-2: COLLABORATION ROOM CASE SCENARIOS	81
FIGURE 4-3: SMART ROOM SERVICE COMPOSITION	82
FIGURE 4-4: PART OF HARDWARE SETTINGS – SENSOR AND ACTUATORS DEVICES.....	83
FIGURE 4-5: CONTEXTUAL DATA IN RELATIONAL DATABASE SYSTEMS	84
FIGURE 4-6: CONTEXTUAL DATA AND SERVICE ORIENTED ENVIRONMENT	85
FIGURE 4-7: RELATIONSHIP BETWEEN DATA VARIABLES	86
FIGURE 4-8: DATA CONSTRAINTS ON DIFFERENT LEVELS.....	88
FIGURE 4-9: TOTAL DATA QUALITY MANAGEMENT CYCLE (R. WANG AND STRONG 1996).....	90
FIGURE 4-10: DQ ASSESSMENT PROCESS IN SOA	91
FIGURE 4-11: BPMN DIAGRAM OF THE PROFILING PROCESS IN SOA.....	93

FIGURE 4-12: IBM WEBSPHERE'S SERVICE REGISTRY GRAPHICAL EXPLORER (TOP) AND LIQUID XML'S WSDL VIEWER (BOTTOM)	94
FIGURE 4-13: PROCESS OF DEFINING "QUALITY" IN SOA	98
FIGURE 4-14: EXAMPLE OF A PREDICATE AND ITS ARGUMENTS.....	101
FIGURE 4-15: XML SNIPPET CODE FOR STORING SIMPLE DQ STATEMENTS	103
FIGURE 4-16: EXAMPLE OF SIMPLE STATEMENT IN THE SMART ROOM CASE	103
FIGURE 4-17: STRUCTURE OF A COMPLEX DATA QUALITY STATEMENT	104
FIGURE 4-18: XML SNIPPET OF DQ COMPLEX STATEMENT	105
FIGURE 4-19: SEMI-STRUCTURED APPROACH OF STORING DQ STATEMENTS	105
FIGURE 4-20: EXAMPLE OF COMPLEX STATEMENT IN THE SMART ROOM CASE	106
FIGURE 4-21: BPMN PROCESS OF EXECUTION STAGE APPLIED TO THE SERVICE ENVIRONMENT	108
FIGURE 4-22: XML LOG STRUCTURE FORMAT	110
FIGURE 4-23: SOA LIFE CYCLE AND REVERSE ENGINEERING	115
FIGURE 4-24: INCORPORATING NEW FEATURES INTO SOA USING REVERSE ENGINEERING.....	121
FIGURE 5-1: DIMENSIONS OF UBIQUITOUS COMPUTING	127
FIGURE 5-2: DYNAMIC OPEN HOME AUTOMATION	130
FIGURE 5-3: DEVICE PROFILE FOR WEB SERVICES STACK.	132
FIGURE 5-4: SEQUENCE DIAGRAM OF A 'PRINTING' PROCESS USING THE DPWS PROTOCOLS	136
FIGURE 5-5: STRUCTURE OF A DOHA SERVICE.....	138
FIGURE 5-6: STRUCTURE OF SERVICE COMPOSITION MAP (HOLGADO-TERRIZA AND RODR'IGUEZ- VALENZUELA 2011)	140
FIGURE 5-7: SERVICE-ORCHESTRATION PRESENTED BY DIRECTIONAL GRAPH	142
FIGURE 5-8: DOHA PROACTIVE BEHAVIOUR CONCEPT	144
FIGURE 5-9: DQ TOOL AND SOA DOMAIN DEVELOPING PLAN	145
FIGURE 5-10: DOHA SEMANTIC LAYER	147
FIGURE 5-11: DQ SERVICE LAYERS	149
FIGURE 5-12: SERVICE DEVELOPMENT OF SMART MEETING ROOM CASE	150
FIGURE 6-1: EVALUATION PROCESS.....	155
FIGURE 6-2 EVALUATION CRITERIA.....	160
FIGURE 6-3 EXOGENOUS FACTORS INFLUENCING THE EVALUATION PROCESS (BASED ON GE 2009)....	163
FIGURE 6-4 MANAGE DQ STATEMENTS.....	172
FIGURE 6-5 EXECUTION OF DQ STATEMENTS.....	173
FIGURE 6-6 REPORT AND ANALYSE SCREEN	174
FIGURE 6-7 SUS CURVE (SAURO 2011)	176

List of Tables

TABLE 2-1: SOA DESIGN PRINCIPLES PART 1.....	18
TABLE 2-2: SOA DESIGN PRINCIPLES PART 2.....	19
TABLE 2-3: CONTRIBUTIONS IN SOA ON DIFFERENT LEVELS	27
TABLE 2-4: COMPARISON BETWEEN PRODUCT MANUFACTURING AND INFORMATION MANUFACTURING PUT INTO SERVICE AND SOA CONTEXT	29
TABLE 2-5: DQ CRITERIA SUMMARY	32
TABLE 2-6: DATA QUALITY METHODOLOGIES	33
TABLE 2-7: DATA QUALITY RESEARCH IN WEB INFORMATION SYSTEMS.....	37
TABLE 2-8: SUMMARY TABLE WITH QUALITY DIMENSIONS ON DIFFERENT LEVELS PRESENTED WITH EXAMPLES	42
TABLE 2-9: DQ METHODOLOGIES' COMPARISON – DIMENSIONS, DATA TYPES, IS TYPES AND APPLICATION CONTEXTS	49
TABLE 2-10: DQ COMPARISON – PHASES AND STEPS.....	51
TABLE 2-11: RESEARCH QUESTIONS CONCERNING SCOPING THE PROBLEM.....	52
TABLE 2-12: RESEARCH QUESTIONS CONCERNING ARTIFACT DESIGN.....	53
TABLE 2-13: RESEARCH QUESTIONS CONCERNING EVALUATION OF ARTIFACT.....	53
TABLE 3-1: SUMMARY OF RESEARCH METHODOLOGIES	64
TABLE 4-1: INFORMATION PROFILING TABLE	95
TABLE 4-2: DATA PROFILES IN SOA SMART MEETING ROOM (SIMPLIFIED)	95
TABLE 4-3: VARIABLES DEPENDENCIES AND DESCRIPTIONS	96
TABLE 4-4: VARIABLE DEPENDENCY IN SMART MEETING ROOM CASE.....	96
TABLE 4-5: DQ STATEMENT DECOMPOSITION	101
TABLE 4-6: ROLE GROUPS IN SOA LIFE CYCLE	116
TABLE 4-7: DELIVERABLE AND TOOLS USED IN EACH STAGE OF SOA CYCLE	117
TABLE 4-8: SUMMARY OF DATA QUALITY ASSESSMENT PROCESS IN SOA.....	123
TABLE 4-9: RESEARCH QUESTIONS ADDRESSED IN CHAPTER 4.....	124
TABLE 5-1: SYSTEM MESSAGES DEFINED IN WS-EVENTING PROTOCOL	134
TABLE 5-2: SYSTEM MESSAGES DEFINED IN WS-DISCOVERY PROTOCOL.....	135
TABLE 5-3: RESEARCH QUESTIONS ADDRESSED IN CHAPTER 5	152
TABLE 6-1 EVALUATION CRITERIA DESCRIPTION	161
TABLE 6-2 INFORMATION PROFILING IN SMART MEETING ROOM	171
TABLE 6-3 WORKSHOP PLANNING SETTINGS	168
TABLE 6-4 EVALUATION PROCESS SETTINGS	169
TABLE 6-5 RESEARCH QUESTIONS ADDRESSED IN CHAPTER 6.....	182

List of Abbreviations

Abbreviation	Full Name	Reference
B2B	Business to Business	1.4
BPEL	Business Process Execution Language	2.2.3
BPMN	Business Process Modeling Notation	3.4.2.1
DOHA	Dynamic Open Home Automation	4.3
DPWS	Device Profile for Web Services	5.3.1
DQ/IQ	Data Quality interchangeable with Information Quality	2.4
DQM	Data Quality Management	2.5
DSR	Design Science Research	3.4
ESB	Enterprise Service Bus	2.2.2
IoT	Internet of Things	5.1
IS	Information Systems	Chapter 1-7
IT	Information Technology	Chapter 1-7
JSP	Java Server Page	5.4
OWL*	Web Ontology Language	4.4.1
P2P	Peer to Peer	2.2.2
QoC	Quality of Context	4.6.2
QoS	Quality of Service	2.5.2.2
RE	Reversed Engineering	4.7
SLA	Service Level Agreement	1.4
SLR	Systematic Literature Review	2.5
SOA	Service Oriented Architecture	2.2
SOAP	Simple Object Access Protocol	2.2.3
SoC	System on a Chip	4.3
SUS	System Usability Scale	6.2
TDQM	Total Data Quality Management	4.5
UC	Ubiquitous Computing	5.2
UDDI	Universal Description Discovery and Integration	2.2
UGr	University of Granada	5.3
WS*	Web Service	5.3.1
WSDL	Web Service Definition Language	2.2.3
XML	Extensive Markup Language	2.2

Abstract

Assessing and Analysing Data Quality in Service Oriented Architectures. Developing a Data Quality Process

Plamen Petkov

Over the past decade, the Service Oriented Architecture (SOA) approach has become a preferable way of building information systems. This is largely due to its ability to enable rapid changes in systems by recombining and scaling existing services. However, the more complex the SOA becomes, the more likely are data quality (DQ) issues to be encountered. Despite numerous SOA projects failing due to data quality problems, many organizations and individuals are still ignoring the importance and necessity of data quality. In spite of the large number of studies that have been done on SOA, findings in literature and practice have shown that very little has been investigated about the DQ aspect. Most of the DQ evaluation approaches to date do not consider the services' context and semantic accuracy of the data.

The objective of this research is to investigate challenges in data quality within service composition. More specifically, the goal is to create a method of detecting and analysing semantically inaccurate data within a specific Service-oriented context. In order to reach the given objective, a DQ methodology was proposed which suggests techniques and methods for profiling and assessing data. The proposed approach was developed by following the Data Quality Management (DQM) model. Additionally, to conduct this project, Design Science (DS) oriented methodology for conducting research, which focuses on the development of artifacts, was used. The application and usability of the proposed DQ approach was demonstrated in a home automation system – a specific type of SOA environment.

The contribution of this research is that application of the approach allows practitioners to detect poor data within SOA environment, preventing damages and reducing expenses. It also provides researchers with methods that will serve as a foundation for research in improving data quality and the decision making in SOA field.

Chapter 1 Introduction

1.1 Service-oriented Approach as Design Style

More than ever before, different organizations, research institutions, and governments are relying on trustworthy information and quality services in order to succeed and thrive. Many enterprises have taken the leap into using advantages of different paradigms and concepts of Information System (IS) to achieve their objectives. IS is a vast and complex area of study which enables people and organizations to collect, process and distribute data (Jr 1975; Coy 2004; Jessup, Valacich, and Hall 2008) using Information Technologies (IT), that is, software and hardware (Chandler and Munday 2003). With the emergence of IS, web services have become a powerful and effective tool for governing and utilizing the business processes and data for many companies and institutions. Moreover, to respond to the fast-changing IT trends, enterprises need increasingly more agile and cost saving instruments.

In reply to the need for adaptation, IS have been changing from closed, monolithic, centralized and static architectures to composite dynamic and distributed architectures, (Adelman, Moss, and Abai 2005; Sangwan et al. 2006) whose elements and their correlations may vary dynamically. In comparison with monolithic IS where data is stored and accessed by a single entity, in composite architectures the emerging scenario is open – because new services can appear and disappear; decentralized – because no single authority coordinates all the service evolution and development processes; and finally this type of architecture is dynamic – because the composition may change dexterously.

Service Oriented Architectures is an IS model that accommodate all features of the composite, distributed and web architectures. This approach offers businesses a flexible and agile way of integrating new services and thus improving the efficiency of the busi-

ness process. Owing to SOA key principles such as reusability, interoperability, standardization, etc. enterprises can benefit in various ways (Erl 2005; Marks and Bell 2008) including: reducing the costs of operation and maintenance (Krafzig, Banke, and Slama 2005), lessening the time for deploying new services (L. O'Brien, Merson, and Bass 2007), and increasing agility of service management (Papazoglou and Heuvel 2007). According to recent surveys, conducted by two independent organizations with 137 IT professionals in one (Austvold and Carter 2005) and over 400 in the other (TechTarget 2010), responded that the primary drivers to undertake a SOA project were cost saving and the shorter time needed to respond to IT market demand. The methods of developing completely new applications, creating adaptors for legacy systems or rewriting present applications are now outdated (Byrne et al. 2008). Principally boosted by Web services connectivity, service-oriented architectures are now considered the preferred way of designing an information system.

1.2 Importance of Data Quality

Today's fast evolving IT society is exposed to a huge amount of information. We are being constantly bombarded by digital information in every aspect of our life – from social media and education, through transportation, healthcare and economics. High-quality information is a key requirement for any society to function effectively and efficiently. Several groups of society are particularly dependent on the information they deal with - e.g. stock traders, bankers, engineers, physicians, etc. For example, a farmer would highly rely on the quality of weather forecast in order to harvest his crop. Another good example of how quality of information is related to quality of life is the healthcare system. The quality of healthcare data impacts every decision that is made along the patient care continuum. The demand for accurate and reliable data has never been more important.

The sharing of today's healthcare data is constantly increasing. Healthcare data is created at the source by providers such as physician groups, hospitals, skilled nursing facilities and pharmacies during the normal course of business. The data is then transformed into secondary data where it is shared with entities like payers, third party vendors, pub-

lic health agencies, government agencies, health information organizations, etc. Each entity may have different purposes for collecting and using the data. Purpose could range from the clinical, administrative, or financial aspects of data. Data is collected in numerous ways and formats; it uses different technologies and standards, and is often dependent upon the care setting (e.g. who and how facilitates data). Technologies and data standards will vary from pharmacies to laboratories to acute care facilities; each likely to have a different technology vendor and complying with different data standards.

Incorporating data quality issues into service composition and execution is a major problem regarding the application functionality. Fishman represented the problem with bad data quality in SOA by comparing the different services with animals and humans – they are different by nature but share the same diseases (Fishman 2009). For instance, Figure 1-2 shows how “Mr. Mosquito” (the SOA environment) can transmit Malaria disease (bad data) from infected deer (provider service) to “Lindsey” and her dog “Toby” (consumer services). Malaria parasites are micro-organisms that can infect many animal species such as livestock and various mammals via blood transfusion. Related to SOA, integrating an application which does not comply with the quality rules may generate inaccurate or inconsistent data.

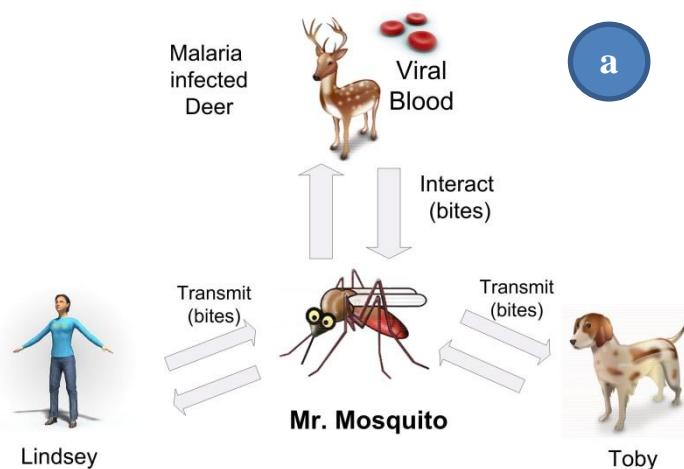


Figure 1-1: Spreading Malaria Disease

This, in turn, can affect all other applications and cause errors to accumulate, something are typical for system development lifecycle. Figure 1-2 illustrates the service composition and service incompliance.

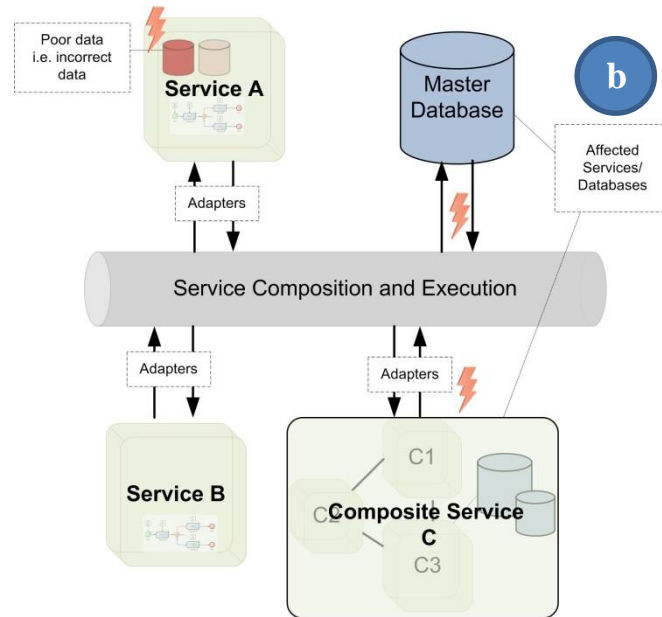


Figure 1-2: Spreading Poor Data Quality in SOA

As it can be seen from the diagram, incompliance at service level (e.g. Service A) can ‘spread’ all poor data through the service environment to all other applications and its databases (i.e. Master database, Composite Service C). Note that Service B does not have its own database but it supplies the whole infrastructure with functionalities and thus it has been involved indirectly in the process of distributing poor data – e.g. affecting the master database. Therefore, finding if there is a problem or not becomes a challenging task considering all diverse services.

Indeed, in more complex architectures, managing the services and quality of their data can be difficult to handle (P Petkov and Helfert 2012). It could be really challenging to govern such architectures without having the awareness of the data, processes and events running within the environment. To support the process of development, orchestration and evolution of such services compositions, data monitoring techniques must be integrated (Kazhamiakin, Metzger, and Pistore 2008) .

1.3 Research Problem and Objectives

To assess ‘something’ in terms of data quality, it requires ‘something’ else. Adding that little something extra to the information, give trails or heads of what to be assessed. This additional information could be supporting metadata, report about the provenance of the assessed data, or it could be a document that specifies the meaning of data. Many would ask why is the need to assess data? Possibly to ensure correctness of data involved in making a decision about important action, plan or judge current data conditions, or predict a future event? Regardless what the reason is, trying to conduct data assessment without considering the context of the data is like “shooting in the dark - the venture is much more likely to fail than it is to succeed” (Fishman 2009). Context is a key word in accessing information. It can provide essential background information, reason or reference, or anything that can deliver and contribute to describing meaning. Context becomes the provisioning of an anchor point from which to venture, a place which to succeed (Fishman 2009).

In Chapter 2.2 Service contract is identified as a key concept in SOA paradigm. It defines service capabilities and all aspects of its behaviour such as specific functionalities, agreement policies, deliverables, etc. However, an important piece of information that service contracts never include is about the meaning of the data that providers and consumers exchange. They also lack to define any specifics about how a provider or a consumer will go about meeting the requirements of the contract. In addition, since consumers vary just as much as providers, there might be multiple contracts for a single service. Moreover, service contract also does not express the semantic requirements (meaning) for the data that a service produce at its full. Rather, it only provides a link to data transformers, that most of the time do some syntactical changes over data. For this reason detecting semantically inaccuracies require something more than a just a service contract. Recent studies have tried to tackle the problem with service compatibility (e.g. interoperability) by introducing different add-ons and standards (Kang et al. 2009; Ejigu, Scuturici, and Brunie 2007; Papazoglou and Heuvel 2007). Such technology, for example, is Web Ontology Language for describing semantics (OWL-s) which is used

in conjunction with the service contract to express the semantics of the service. Nevertheless this technology does not include any data quality insurance mechanism - e.g. it cannot ensure the quality of the data running through the service.

On the other hand, the literature review outlined in Chapter Two demonstrates that a considerable amount of research with respect to the Data Quality focuses upon the growth in the number of data sources in terms of size and scope. Academics and practitioners have addressed the Information Quality (IQ) challenges broadly from two perspectives – “managing” and “improving” the IQ. In this study, terms Data Quality and Information Quality are considered inter-exchangeable, meaning they refer to the same concept. Research in IQ has focused on developing constructs such as frameworks, dimensions and assessment methodologies. Much of the research concentrates on identifying new dimensions and developing novel processes (Pierce 2004; English 1999; R. Wang, Storey, and Firth 1995; Pipino, Lee, and Wang 2002). Other studies have put more focus into developing DQ methodologies for different organizational contexts (Lee 2003; Ballou et al. 1998; Yair Wand and Wang 1996). Also, observations in this research have shown that most of the DQ solutions are not empirically evaluated, or assessments are carried out on very large scale contexts. As a result, the evaluation of most data quality methodologies remains ambiguous or unreported due to data protection policies.

Regardless of the fact that some data quality methodologies are evaluated in particular contexts, literature does not present a full spectrum of DQ applications e.g. there is a lack of studies regarding DQ in the SOA context. The reason for this shortfall lies within the foundation of the SOA concept. Current DQ methodologies are designed to assess and manage information either on organisational level (e.g. people and processes that operate with the information), or when it comes to the evaluation of data in particular system they rely on different programmatic algorithms such as data mining data cleansing, which in turn, require direct access to the datasets under investigation. On the other hand, several fundamental principles embedded in SOA suggest that data should be abstracted e.g. services hide logic and physical structure of data from the outside world.

This feature is not considered within existing theories of data quality management and assessment methods provided by them become inapplicable to SOA since there is no direct access and transparency of the datasets.

1.4 Research Scope and Requirements

To satisfy growing customers' needs and ensure higher quality of service (QoS) levels, while offering low-cost services and faster delivery to the market, providers usually collaborate together and compose their services to form inter-organizational virtual enterprises. This requires the collaboration between multiple processes, services, and partners. Each organization provides services to different customers and requires services from other organizations. To get control over the use and the delivery of services and to supervise relationships between multiple organizations and ensure the performance of the outsourced services, partners often use agreements. These agreements allow the specification of common collaboration parameters between service consumers and producers within the involved parties. They also specify guarantees and obligations of each party. An agreement is defined as a (service) contract between the (service) consumer and (service) supplier also known as Service Level Agreements (SLA). SLA describes functional and non-functional characteristics, specification around parties, obligation and penalties as well as termination policies.

In summary, inter-organisational or Business-to-Business (B2B) service compositions are systems that enable the use of services that are outside the scope managed composition. In big corporate SOAs, a dedicated module called B2B Gateways is introduced to facilitate the communication between companies. This component provides a centralized point for transformation of multiple data sources using standards that tackles interoperability issues e.g. data and process adaptors. Major concern related with B2B SOA, however, is related to the security of communication between one business services and the other one. Several studies have addressed this issue presenting frameworks and solution that aim to increase the security in inter-organisational types of SOA e.g. WS-Security WS-Policy and WS-Federation, etc. One withdrawal of this approach before intra-organisational one is the ability of the system to directly observe data flow

beyond the scope of the enterprise, mainly due to the security mechanisms stated above. In such case, a data quality process would allow the identification of the source of possible poor data but it would not provide any indications around why certain problem occurred. In this research both inter and intra - organisational systems are taken under consideration.

Another aspect that needs focus in this study is in what way data quality issues are detected and analysed. Detection of poor data quality would require checking the data against predefined user and environmental conditions. The DQ checking process may be divided into two categories – real-time detection and scheduled DQ checks. Real-time detection is characterised by the ability to identify problems e.g. poor information as soon as they occur. The time needed for detecting issue depends on factors such as the responsiveness of the SOA environment, communication infrastructure, network load, service capacities (i.e. response time and throughput) as well as the responsiveness of the monitoring tool. On the other hand, scheduled checks occur only at particular moments defined by the data quality evaluator. This type of DQ checks aims to detect problems in batches, for a certain window of time (for example from 12am to 6pm, for 24h, one week, etc). This approach requires the data to be persistent – e.g. stored in accessible database. Considering this, the time required to detect a problem is proportional on the size of the data records and services involved in processing information from the database. This method relieves SOA environment in terms of system and infrastructure load since it does not require constant checking mechanism which would add extra overhead to the business process execution times. An important requirement when executing scheduled checking of particular data is that process manipulating it and requirement constraining it and context affecting it are taken into consideration. Results are expected to be accurate only if all these factors are timely considered together. In this research, both on-demand (real-time) and scheduled data detection and analysis are discussed.

Considering the shortcomings mentioned above and in section 1.2, a framework, a set of processes that would be capable of defining and assessing and analysing data quality in

service oriented context while considering SOA principles is needed. The objective of this study is to build and evaluate such methodology which will ultimately be used as a foundation for reference in practice and further research. Since every SOA is different and DQ by nature is “multi-dimensional” (e.g. quality is different for every user), the DQ framework must examine SOA technologies specifications as well as take into account users’ definitions of “quality”. The DQ framework has also to be comprehensive enough to be used by a wide range of people with a different knowledge base. On the other hand, the evaluation of my research must be rigorous – i.e. conducted in a specific SOA context. Ultimately, it is believed that fulfilling these requirements would aid practitioners (e.g. managers and architects) who venture on a SOA project to detect and assess poor quality of information. Figure 1-3 presents an overview picture of the research objective and its requirements.

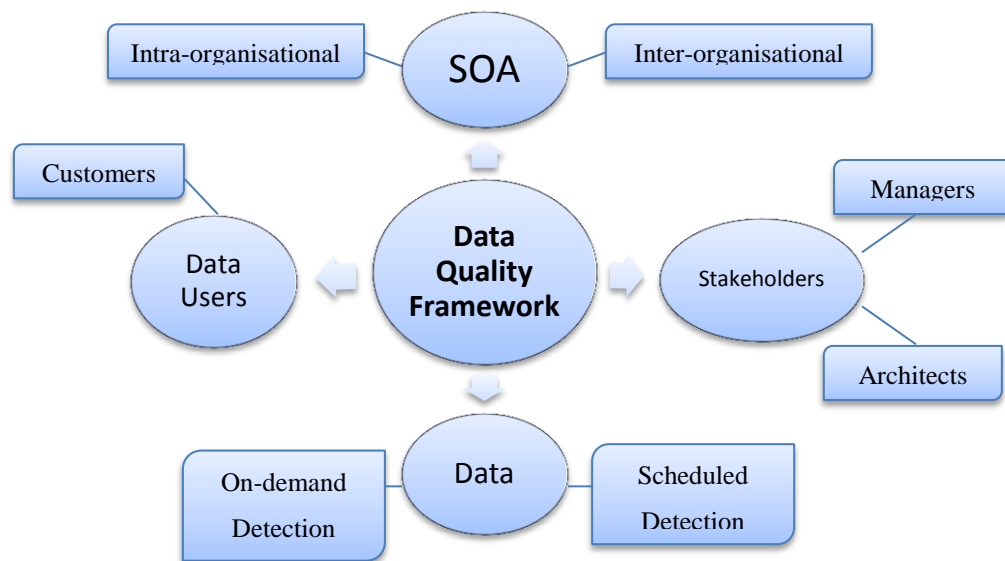


Figure 1-3: Overview of Research Objective and Requirements

1.5 Thesis Organization

Following paragraph presents an overview of thesis structure and content. Figure 1-4 illustrates the sequence of the thesis along with the topics associated with each of the

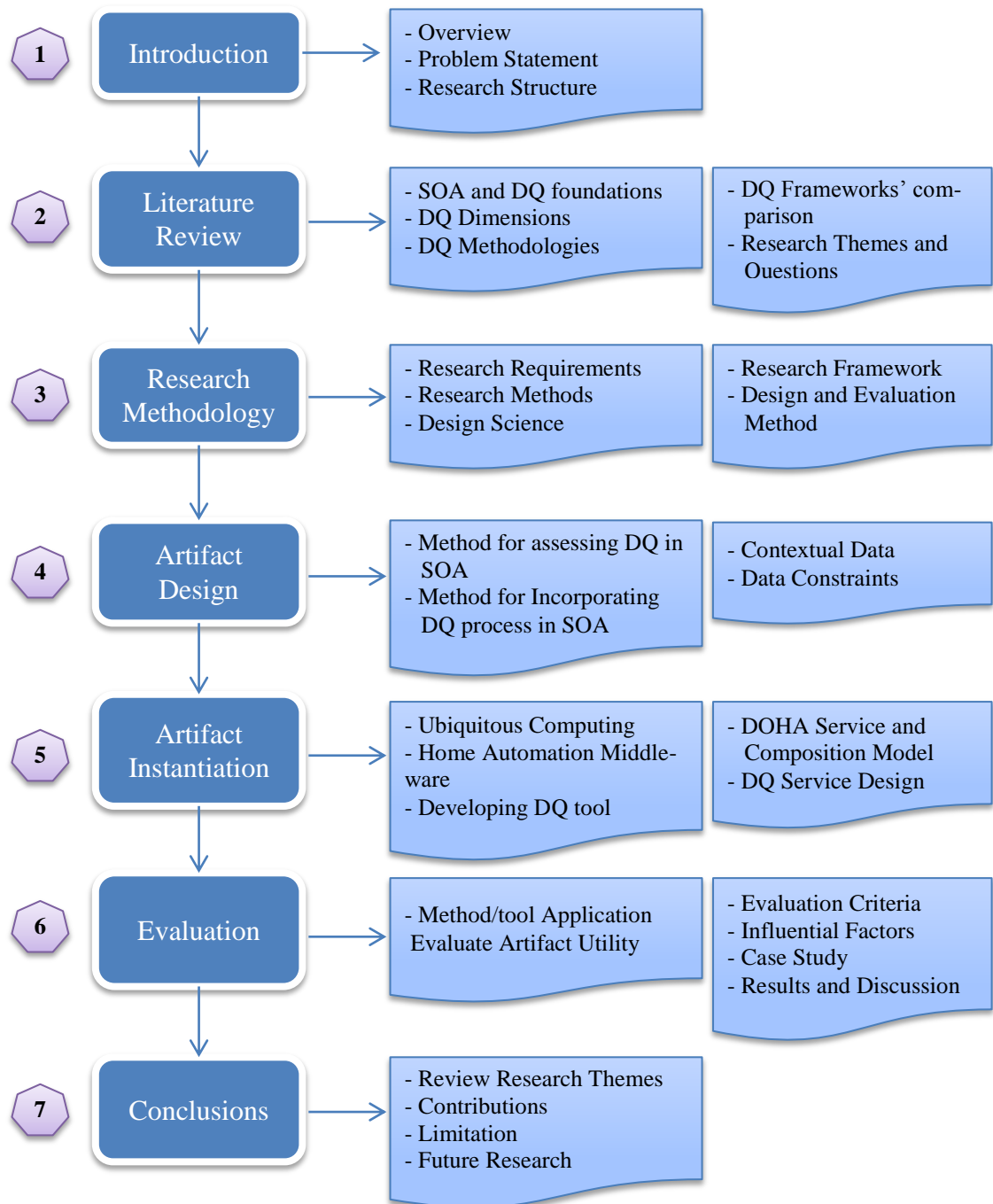


Figure 1-4: Thesis Organization

chapters. The remainder of this thesis is organized as follows: Chapter 2 sets the stage for understanding and positioning this research. More specifically, it outlines all relevant theories in both SOA and DQ domains. Further on in this chapter, an extensive review addressing the limitations of current data quality and SOA research is presented. This chapter is then concluded with a summary of existing research weaknesses and a highlight of research questions considered under investigation. Chapter 3 presents the approach employed to conduct the research study. Based on the research objectives research methodology requirements are firstly defined. Then, various research approaches this study benefits from are investigated. Following the brief discussion, main approach and its comprising techniques are explained. This chapter also explains how the artifact is designed and built. Based on the review in Chapter 2 and research methodology, Chapter 4 proposes an IT artifact to answer the second research question. Particularly, a process for assessing DQ in SOA is proposed, along with approach for incorporating this process within particular SOA context. Following latter approach, in Chapter 5 explores particular service oriented environment and instantiate the process proposed in Chapter 4. Chapter 6 describes the method used to evaluate instantiated artifact. In this chapter, a discussion on evaluation criteria and influential factors that affect results of the study is delivered. Then evaluation execution is presented followed by discussion and conclusions about the utility of the instantiated artifact. Finally, Chapter 7 completes the thesis by summarizing research findings and outlining possibilities for further research.

Chapter 2 Literature Review

2.1 Overview

The reviewing of existing literature relating to a topic is an essential first step and foundation when undertaking a research project. This chapter will position this research within the fields of Information Systems, Service Oriented Architectures, and Data Quality. In order to conduct academic research in a comprehensive and professional tone, it is important that a methodical review of past literature be completed (Booth, Papaioannou, and Sutton 2011). There is a need to investigate what is already known about a specific area in order to identify gaps in the literature, prior to completing any studies. Moreover, (Hart 2001) suggests that the accent on the quality of the literature review is also of great importance; it must be of enough breadth and depth; contain rigor, consistency, clarity and brevity and critically contain effective analysis and synthesis. The literature review must provide a solid framework for advancing the body of existing knowledge. The literature review must facilitate the development of theories and have the potential to close a plethora of open questions while at the same time open many new avenues for future research (Webster and Watson 2002). Finally, (Booth, Papaioannou, and Sutton 2011; Hart 2001) summarizes that a comprehensive and effective literature should be identified by the following features:

- place each work in the context of how it contributes to and understanding of the subject under review;
- describe how each work relates to the others under consideration;
- identify new ways to interpret, and shed light on gaps in, previous research;
- identify and resolve conflicts across seemingly contradictory previous studies;
- identify what has been covered by previous scholars to prevent you needlessly duplicating their effort;
- to signpost the way for further research; and

- to locate your original work within the existing literature;

Since the area of Data Quality research incorporates a number of disciplines, and in order to put this research in context, first section initially examines Service Oriented Architectures. In this section, the emphasis is on defining some main concepts in SOA research as well as investigating the structure of a typical architecture, its building blocks and layers and how these different layers relate to each other. In this section and investigation of the current state of research regarding each SOA layer is also conducted. These theoretical explorations will then be followed by a discussion of the challenges facing SOA research. Following the previous section, an analysis of the relationship between DQ and IS is presented. Furthermore, key concepts used in this research with regard to DQ are defined. The literature review then completes by examining the challenges posed by DQ and SOA research to the central research question - the challenge of measuring data quality in service oriented architectures.

2.2 Fundamental Concepts of Service Oriented Architectures

Over the past two decades, many definitions of SOA have been published. The universally accepted normative OASIS Reference model for SOA (MacKenzie, Laskey, and McCabe 2006) defines SOA as “.....*a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains. It provides a uniform means to offer, discover, interact with and use capabilities to produce desired effects consistent with measurable preconditions and expectations*” Presented in a different way, a service composition is a process of combining simple services in order to make more complex ones. The service composition model provides a key benefit: ability to develop new function combinations rapidly. A SOA composition is implemented by services sending messages to invoke other services. This approach gives much greater flexibility than traditional architectures.

The service-oriented paradigm defines that service could take two main roles - service provider and a service requester. A service discovery agency (e.g. Universal Description

Discovery and Integration UDDI)) may act as an intermediate between provider and requester, offering functionality that promotes available services. The service provider defines a service description and publishes it (to the agency). On the other hand, after finding a suitable service, the service requester is able to invoke that service (The Open Group 2015a). In this regard, the service composition encompasses the process of searching and discovering relevant services, selecting suitable (web) services of best quality and finally composing these services to achieve an overall goal that, usually in the business context, aims to support an underlying business process. Note that sometimes consumers could consume and provide at the same time. The same rule applies for providers - they can provide and consume simultaneously, The Publish – Discover – Binding schema is depicted below Figure 2-1.

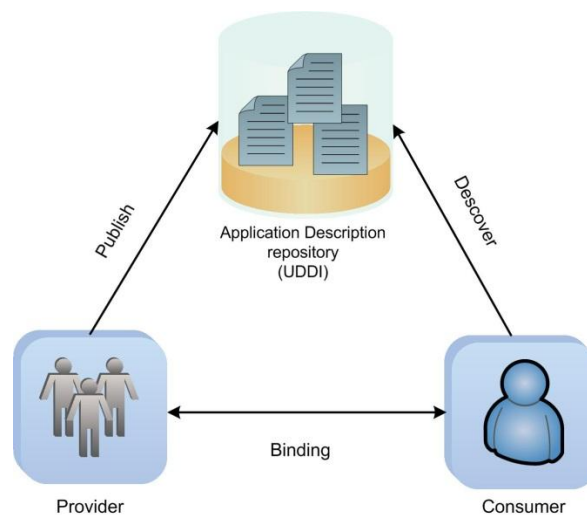


Figure 2-1: ‘Publish – Discover – Binding’ Mechanism in SOA

2.2.1 Drivers for SOA

SOA Drivers are those properties of the paradigm that make it stand before conventional client-server architectures. Clear service descriptions are a starting point for service re-use, which can provide another major benefit of SOA:

- Using existing software modules rather than writing new ones means lower development time cycle and testing costs and in many cases even greater saving – lower maintenance costs.
- Today, the majority IT solutions are designed as “information silos” that cannot readily exchange information between each other and other applications. In contrast to the use of these large applications, the use of finer-grained software services gives freer information flow within and between enterprises. Integrating major applications is often expensive. SOA can save integration costs.
- Organizing internal software as services makes it easier to expose its functionality externally. This leads to increased visibility that can have business value as, for example, when a logistics company makes the tracking of shipments visible to its customers, increasing customer satisfaction and reducing the costly overhead of status enquiries.

Business processes are often dependent on their supporting software. It can be hard to change large, monolithic programs. This can make it difficult to change the business processes to meet new requirements (arising, for example, from changes in legislation) or to take advantage of new business opportunities. On the other hand, service-based software architecture is easier to change – it has greater organizational flexibility, enabling it to avoid penalties and reap commercial advantage. (This is one of the ways in which SOA can make an enterprise or any system more “agile”.)

SOA is a strong candidate paradigm for the realization of the agility, flexibility, and automation of the business processes that span large distributed systems. It is an approach which levers the systems to remain scalable and flexible while growing. Companies that need customizable solutions or use IT for competitive value, companies seeking to leverage IT capabilities for business’ advantage are just examples of enterprises seeking to employ SOA. SOA supports the realization of strategic goals that enables business and IT to collaborate in order to achieve:

- Greater flexibility in strategic applications

- Faster time to value from IT
- Modernized strategic applications
- Lower the lifetime cost of applications or infrastructure
- Reuse as a goal to bring products or capabilities to the market faster

SOA provides the potential to elevate the responsiveness and cost-effectiveness of IT through a design paradigm that emphasizes the realization of strategic goals and benefits. Figure 2-2 presents unexhausted list of SOA drivers from both technical and business point of view.

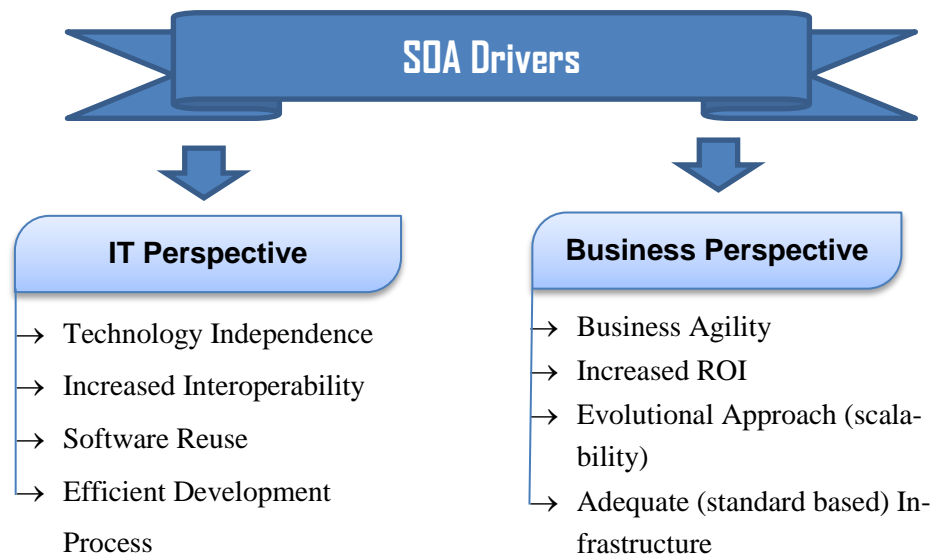


Figure 2-2: SOA Benefits from IT and Business Perspectives

2.2.2 SOA Design Patterns

The goal of the SOA approach is to have various units of solution logic represented and exposed as services. This, however, requires the development of practices and standards in order to help software developers and integrators to identify designs and integrate services. To do so, SOA community had developed SOA design principles. These are generally accepted industry practices that provide rules and guidelines in order to determine exactly how solution logic should be decomposed and shaped into software ser-

vices. The systematic application of service-orientation design principles leverages the creation of services with functional contexts that are agnostic to any business process or application. These agnostic services are therefore capable of participating in multiple service compositions accomplishing the ultimate objective of SOA (Bean 2009). Literature identifies many principles but the most popular are the Loose Coupling, Service Reusability, Discoverability, Autonomy, Abstraction, Composability, Statelessness and the Standardized Service Contract principles. See Figure 2-3: Foundations of Service-oriented Paradigm

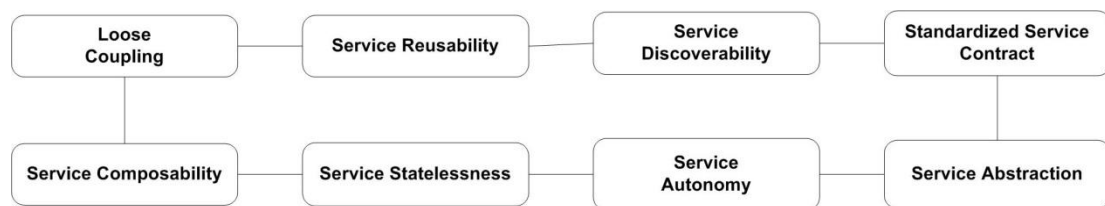


Figure 2-3: Foundations of Service-oriented Paradigm

Table 1 and Table 2 elaborate further more by giving overall textual and graphical description of SOA design principles.

2.2.2.1 Key Principles affecting Data Quality

Despite of the many benefits different SOA strategies bring to the business, there is still a great number disadvantage that should not be overlooked. Particular attention should be paid to the effects of SOA key principles impose on the quality of information. Next few paragraphs discuss some of the drawbacks that SOA strategies can cause in relation with Data Quality:

- **Service composability and granularity.** Service oriented architecture are commonly associated with high complexity, especially if service composition granularity is very fine (e.g. one service is built on many other smaller ones). This involves the use of higher number processes, which in turn, increases the chances of data errors to be introduced. This is particularly probable during the

Table 2-1: SOA Design Principles Part 1


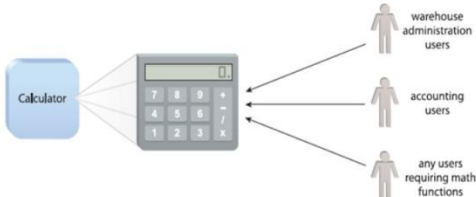
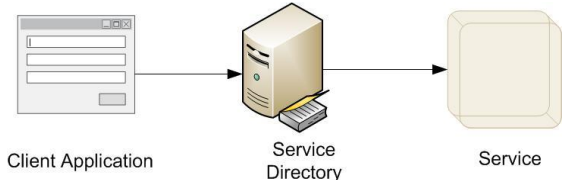
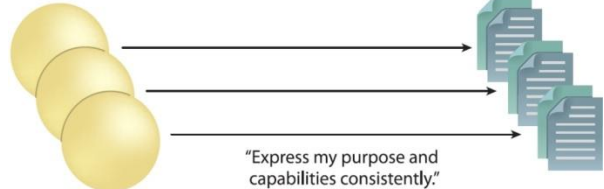
Principle	Description	Graphical representation
Loose Coupling	Coupling refers to a connection or relationship between two things. Services should be designed to have minimal dependencies on each other by achieving logical separation of concerns. Service dependencies should be reduced to minimum in order to minimize the required changes when an upgrade in the system is needed.	 <p>"I was implemented before you were created, but the fact that you've formed a dependency on me affects my ability to evolve"</p> <p>"I was designed to interact with you and I therefore have a permanent runtime dependency on you"</p>
Service Reusability	Reusability implies that the solution logic is divided into services with the intent of maximizing reuse. The services must contain and express agnostic logic and can be positioned as reusable resources. Reuse is a core part of typical service analysis and design processes, and also forms the basis for key service models.	 <p>A generic calculator program designed to provide the same functionality supplied by the Stock Counter program, but packaging it in a generic user-interface equipped with several additional calculation features.</p>
Service Discoverability	Applications should learn about the services of the system in a consistent way. Service discoverability principle implies two requirements: service contracts are equipped with appropriate metadata; a service registry exists in order to store the service description records.	 <p>Client Application Service Directory Service</p>
Service Contract	All service description, purpose, communication protocol, and message format should be documented in a service contract. In order to make all clients understand this contract, it should be written in a standard-based service description format.	 <p>"Express my purpose and capabilities consistently."</p>

Table 2-2: SOA Design Principles Part 2

Principle	Description	Graphical representation
Service Autonomy	This principle ensures that the service benefit is acquired only by the service implementation, and thus, no service is controlled by another service.	
Service Abstraction	Services encapsulate the logic they provide from the outside world avoiding the proliferation of unnecessary service information internal implementation, technology, logic, and function away from users of the services.	
Service Composability	Services are designed as reusable units that can be reconfigured easily to reflect new requirements and business processes, and thus, can be used in different applications. This principle affects directly the business agility. So any application will be composed of any number of services	
Service Statelessness	Service statelessness suggests differing state information as much as possible in order to minimize resource consumption. It encourages incorporating state management to keep services in a stateless condition whenever possible.(e.g. storing session data in a database)	

process of extraction, transformation or loading data from and to service. High volumes of the data traffic could dramatically magnify these problems

- **Service autonomy and abstraction.** Services have control over the logic and data they encapsulate. Additionally, process and database logic are hidden from the “outside world”, in order to provide reusability and better efficiency needed for reduced cost. Resulting from this principle, physical data is visible only inside the service, hence, processes manipulating internal database can cause problems due to service upgrades, mass data upgrades, or internal service database redesign. Consequently, this would affect all other services communicating with affected one.
- **Use of different service interfaces, internet protocols, and standards.** A fundamental principal in SOA is the use of services’ functionalities over the network or Internet. This requires utilization of standardised network and communication protocols such as HTTP/UDP and data transfer technologies such as REST, SOAP, CORBA, etc. Additionally, SOA promotes the implementation of various interfaces to improve usability and interoperability. However, each discrepancy in communication technologies as well as the use of many and different interfaces would introduce additional amount of processing overhead. Such processing delays might negatively affect the timeliness of the data, and from there on, its accuracy.

Another important component that is tightly related to SOA design principles is the messaging infrastructure. Messaging infrastructure is advocating certain networking concepts and standards, which in turn mediate the messages between different services, and in this way allowing them to communicate efficiently. In the service oriented environment there are two significant infrastructure (integration) patterns available: point-to-point (P2P) and point-to-hub (Bianco and Kotermanski 2007). Figure 2-4 provides a simplified comparison of point-to-hub and P2P integration topologies.

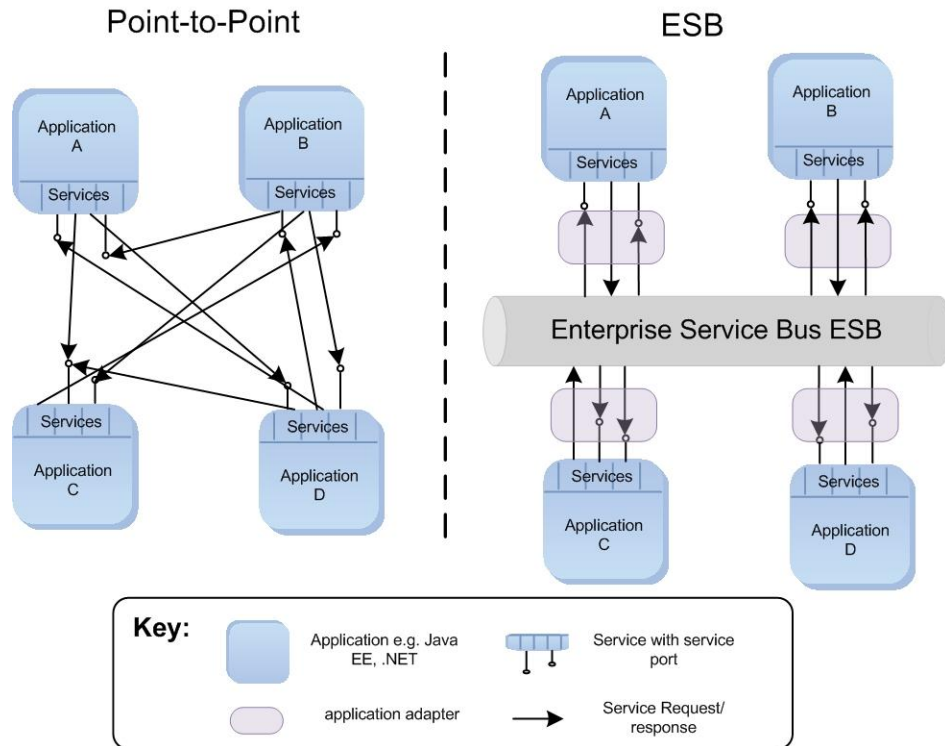


Figure 2-4: Point-to-Point and Point-to-Hub Interaction Models

The point-to-point (or also known as peer-to-peer) approach enables the direct connection between the applications without any mediator. This means that all connectivity, logic, data and security policies are distributed among the application. Integrating services using this approach is usually very fast because it requires less thinking ahead of time (e.g. what common standards to be used). Also, the integration time is generally quick, and thus, the initial integration projects cost less. Furthermore, the rate of data exchange between the systems is usually very fast because there are no mediation layers between the systems to add any overhead (other than the usual machine and process boundaries). Despite aforementioned advantages, this model also brings some drawbacks. For example, expanding the service composition would result in a higher number of common relationships between services. This would exponentially increase the complexity of the system. Every time an application interface is changed or upgraded, it would require modification of all applications that are dependent on the upgraded application. This makes this type of integration unsuitable for big, long-term projects.

In the hub-and-spoke approach, the interaction between service' consumers and providers is mediated by brokering software. In the SOA space, this brokering software is usually called service bus or Enterprise Service Bus (ESB) (Ortiz 2007). This element could be considered as a core in service-orientation environment as it enables principle such as reusability and interoperability. Each application is designed to interact with the ESB, allowing it to manage the routing and transformation of messages between applications. Other functionalities that ESB provides are data transformation (e.g. through data adaptors), transport mediation (e.g. incorporating various transport protocols), intelligent routing and security (Papazoglou and Heuvel 2007). It does utilize a number of benefits over the P2P approach. First, it reduces the number of system integration points which, in turn, saves time and money, especially when making changes to an existing system. Secondly, it centralizes logic, providing a greater control. For example, if a new functionality is introduced, necessary changes to the process logic could be committed directly from the ESB system. However, these advantages do not come for free. The overall system complexity and initial implementation cost increase by adding an ESB to the architecture. Thus, adopting this approach may not be appropriate in environments with a small number of applications and services, or in projects with a tight schedule. Another drawback stems from the fact that ESB acts as a mediator between services. Performance may be negatively impacted due to additional message hops and message transformation performed by the ESB. Finally, this centralized approach further complicated matters because a single mistake within the integration logic could bring down the entire architecture.

SOA community distinguished two major styles of service composition that are in a way related to SOA infrastructure and integration. The terms orchestration and choreography describe two aspects of creating processes from composite services in SOA. These terms overlap somewhat, but Figure 2-5 illustrates their relationship to each other at a high level.

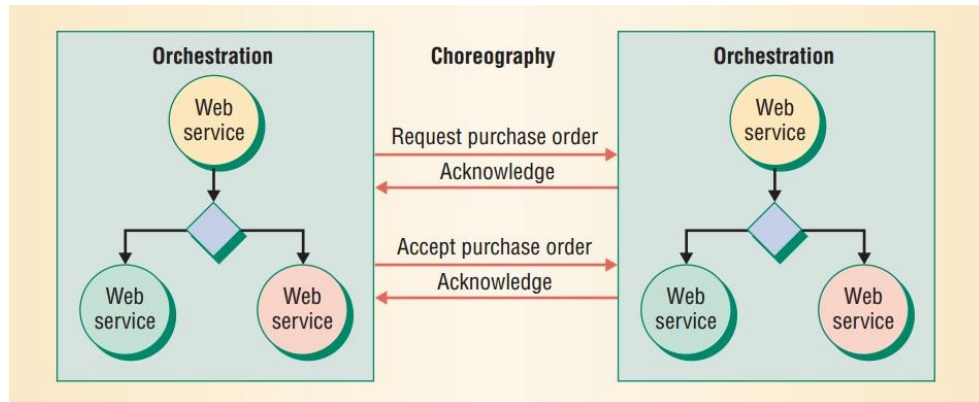


Figure 2-5: Orchestration versus Choreography (Peltz 2003)

Orchestration refers to an executable business process that can interact with both internal and external services. Here, the interactions occur at a message level. They include process logic and task execution order, and they can span applications and organizations to define a long-lived, transactional, multistep process model. In other words, there is always one service that schedules and directs the others (e.g. orchestrator).

Orchestration always represents control from one party's perspective. On the other hand, the choreography approach is more collaborative and allows each party to describe its part in the interaction (Peltz 2003). Choreography tracks the message sequences among multiple parties and sources - typically the public message exchanges that occur between services - rather than a specific process that a single party executes. Described in another way, choreography promotes the composed services to interact and cooperate without the aid of a directing service.

2.2.3 Web Services Approach

Web services are one of the commonly used approaches to realize the concept of SOA (Newcomer and Lomow 2005). One definition of web services is: "A self-contained, modular applications that can be described, published, located, and invoked over a network" (Wahli 2006). Web services can execute different sized business functions e.g., simple data requests or wrap legacy system to make them network enabled. They also can call other web services to achieve certain business func-

tion. These web services will be called by the frontend application in order to achieve specific functions to the user.

As mention in previous paragraphs, web services need to follow strictly certain standards. Most accentuated standards in literature and practice are Simple Object Access Protocol (SOAP), RESTful, Web Services Description Language (WSDL) and Universal Description, Discovery and Integration (UDDI) standards. A short description is given below. Other standards that cover security, web management, and transportation exist such as WE-Security, Business Process Execution Language (BPEL), e-business XML (ebXML), Simple Mail Transfer Protocol (SMTP), File Transfer Protocol (FTP), etc.; descriptions, however, will be omitted due to the scope of the thesis.

SOAP: (Simple Object Access Protocol) is the standard format for the messages between the web service architecture components. As depicted in Figure 2-6, it is based on XML (Extensible Markup Language), and it is independent of programming languages.

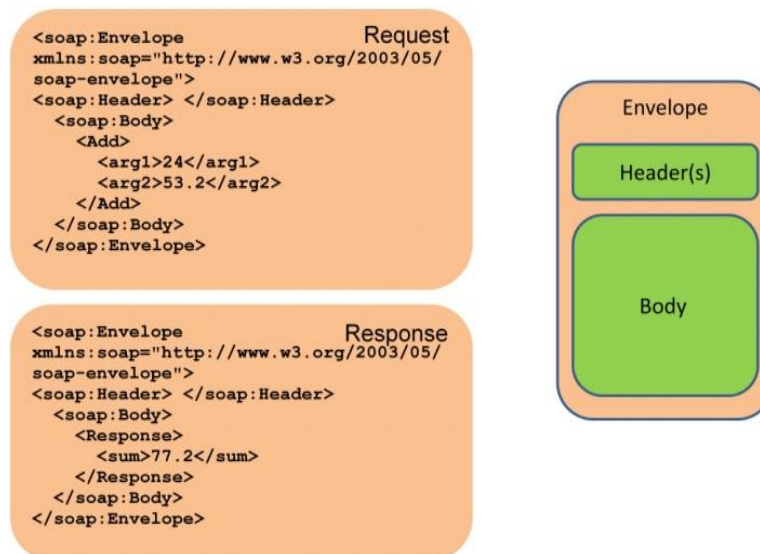


Figure 2-6: SOAP Message Format

WSDL: Web Services Description Language is also based on XML and it describes the implementation of a service. It is used by both the service provider and the ser-

vice requester. The service provider specifies the operations that it provides and the parameters and data types of these operations. The WSDL document includes the information about the location of the web service and message format. An example of a WSDL document is presented in Figure 2-7.

```
<message name="getTermRequest">
  <part name="term" type="xs:string"/>
</message>

<message name="getTermResponse">
  <part name="value" type="xs:string"/>
</message>

<portType name="glossaryTerms">
  <operation name="getTerm">
    <input message="getTermRequest"/>
    <output message="getTermResponse"/>
  </operation>
</portType>
```

Figure 2-7: WSDL Document Format

UDDI: Is a web service and it is used to register the description about the web services and make it available for other service requesters to discover. For example, If industry publishes an UDDI standard for flight rate checking and reservation, airlines could register their services into an UDDI directory. Travel agencies could then search the UDDI directory to find the airline's reservation interfaces. When the interface is found, the travel agency can communicate with the service immediately because it uses a well-defined reservation interface.

2.3 Service Oriented Research

Research in Service-oriented Architectures is vast and highly diverse. This is due to the large amount of technologies, standards and design patterns involved into the SOA paradigm. Next paragraphs present just some of the most notable work done in the service composition environments. Figure 2-8 depicts layer diagram of SOA and it is used to present contribution in a more structured way. As seen, SOA architecture is divided into four main layers – Presentation, Business, Application and Data layers. Two additional layers – Security and Communication are also integrated into the scheme. In addition, unlikely other layer diagrams e.g. (Mos et al. 2008; Kreger,

Brunssen, and Sawyer 2012; The Open Group 2015), this research introduces the Quality perspective. Quality dimension could be applied to all six layers but the majority of the literature associates it with business layer, service layer, and data layer.

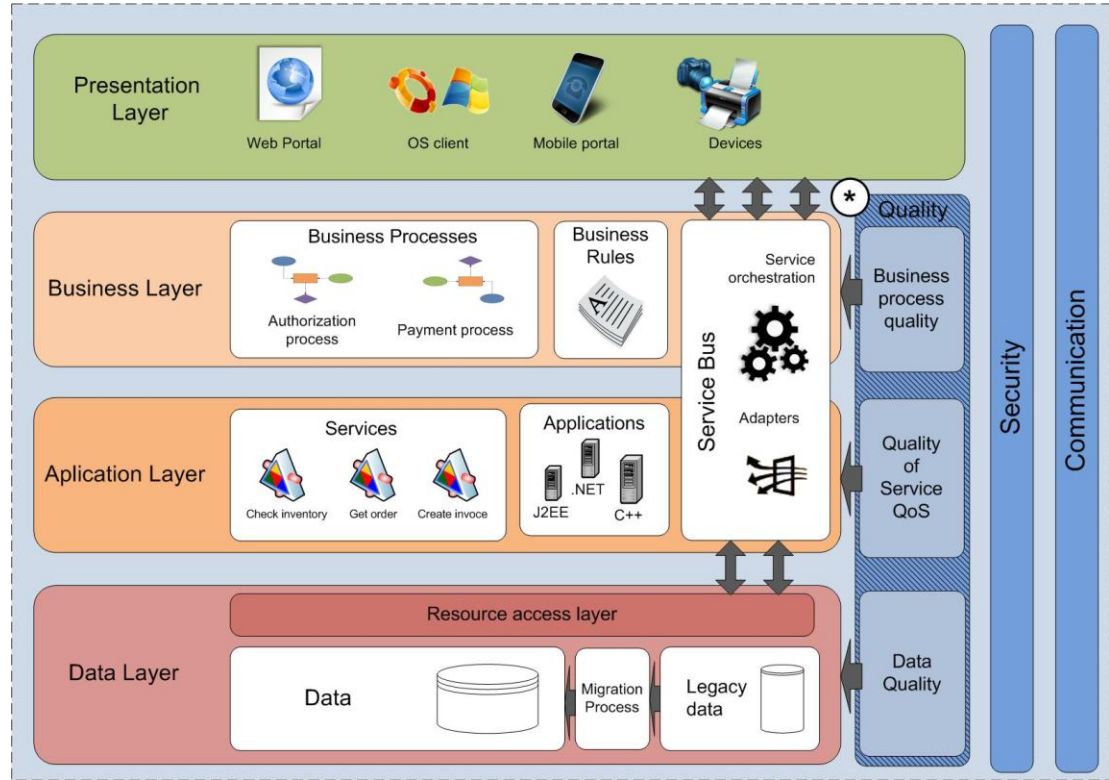


Figure 2-8: Service Oriented Architectures' Layer Diagram

Conducted systematic literature review regarding separate layers shows that research focuses on constructing business processes and rules (Debevoise 2007) - business layer; service integration and governance (Erl 2005; Bennett et al. 2011) - application layer, data integration (Bauchot and Lalanne 2015; Goel 2006) and migration (Channabasavaiah, Holley, and Tuggle 2004) - data layer, as well as security (Carminati, Ferrari, and Hung 2005; 2006) and communication (Kazhamiakin and Pistore 2006). With respect to the quality of these layers, most of the research focuses on monitoring quality of business processes e.g. (Baresi and Guinea 2009; Kang et al. 2009), Quality of Service (QoS) (Zhai, Zhang, and Lin 2009) and resolving problems with data on a low, schematic level (Byrne et al. 2008; Comerio and Truong 2010). An overall summary of the research done in the area of SOA is presented in Table

2-3: Contributions in SOA on Different Levels. Note, that the table contains a non-exhaustive list of contributions made in different areas and dimensions of SOA and there are many more. More detailed research in SOA is presented in sections 5.2, 5.3 and through the whole document.

Table 2-3: Contributions in SOA on Different Levels

SOA layers and dimension	Main contribution	Reference
Business	A framework for monitoring the business process execution	(Baresi and Guinea 2009)
	An OWL-based semantic business process monitoring framework	(Kang et al. 2009)
	Business Process Management with a Business Rules Approach: Implementing The Service Oriented Architecture	(Debevoise 2007)
Application	An Open and dynamic QoS computation model for web services selection through implementation and experimentation with a QoS registry.	(Kazhamiakin and Pistore 2006)
	An approach for repairing and replacing failed services them with new services and ensuring the new service process still meets the user specified end-to-end QoS constraints.	(Zhai, Zhang, and Lin 2009)
	An approach to integrate dynamic aspect-oriented programming into a SOA platform to adapt existing services at runtime to new requirements.	(Irmert, Meyerhöfer, and Weiten 2007)
Communication and Security	A method for modelling data security constraints to build a web services	(Carminati, Ferrari, and Hung 2006)
	An approach that deals with asynchronous interactions and the ability to analyse the data exchanged via communication protocols.	(Kazhamiakin and Pistore 2006)
Data	The value of applying the data quality analysis pattern in SOA	(Byrne et al. 2008)
	Service-oriented data quality engineering and data publishing in the cloud	(Comerio and Truong 2010)

Despite, the efforts and studies that had been done on each of the areas above, investigations in the literature and practice (as shown later in this chapter) showed that methodological data quality approaches are not clearly defined. Next section discuss-

es some of the most common data quality methodologies as well as the criteria that are used to compare them.

2.4 Information Systems and Data Quality

As more and more people have become aware of the significance of the Data Quality, an extensive researched in the area has been developed over the last twenty years. Since the majority of the literature suggests the relationship between Information Quality and Information Systems, it is essential to elucidate the meaning of information systems. Some studies define information systems as computerized database systems which main goals are to store, process, and analyse data and to report meaningful reports (J. A. O'Brien 2003)(Strong, et.al., 1997). The term is also broadly defined as “any means for communicating knowledge from one person to another, such as by simple verbal communication, punched-card systems, optical coincidence systems based on coordinate indexing, and completely computerized methods of storing, searching, and retrieving of information” (McGraw-Hill, 2003.). Similarly, in this thesis, term “information system” describes the database, software, and hardware used for collecting, storing, searching, processing, distributing and analysing data by people or organizational processes.

Following the description of Information systems, a different approach in information quality research is suggested by (Ballou et al. 1998) in which information manufacturing is compared with product manufacturing. Product manufacturing is a processing system that transforms raw materials into finished products. Likewise, information manufacturing can be treated as a process of transformation of raw data into information product that can be used by the information consumer. The product has a description of what the consumer needs. When the product meets the consumer's requirements it is said to have achieved quality. Stemming from this discussion, information products can be considered as a result of the process of manufacturing raw data into valuable information (Pierce 2004). Ballou et al. 1998, Huang, Lee, and Wang 1998 describe the concept of the information manufacturing system as “a system that operates on raw data to create information products”. A comparison table

depicting relationship between information manufacturing and product manufacturing is presented on Table 2-4.

Table 2-4: Comparison between product manufacturing and information manufacturing put into Service and SOA context

	Input	Process	Output
Product Manufacturing	Raw Materials	Assembly Line	Assembled(Finished) Products
Information Manufacturing	Raw Data	Information System	Information Product
Service Level	Input Data	Service	Output Data
Architectural level	Service Provider	Service Consumer and Provider	Service Consumer

In relation to Huang's comparison of Product Information with Information manufacturing, this research extends the aforementioned knowledge by putting information manufacturing in Service Oriented Architectures context. This extension is presented in Table 2-4 and it is based on the fundamental principles rooted within Service Orientation. Some of these principles are described in section 2.2. The comparison between Information Manufacturing and Service Oriented Architectures is divided into two levels – service and architectural. On the service level, manufacturing of information is seen from a single service point of view, where service is seen as a process that transforms raw input data into processed output data. On the other hand, on the architectural level, information manufacturing is seen as a composite process where raw input data is provided by service provider then a service consumes it, process it, and ultimately provides it to third services, which in its turn, consumes the processed output data.

2.5 Comparison of Data Quality Frameworks

This section provides detailed analysis on some of the most relevant data quality research from the past two decades. Investigations on DQ literature (R. Wang and Strong 1996; English 1999; C Batini and Scannapieco 2006; Ge 2009; Foley 2011; Loshin 2001; Lee et al. 2009) indicated that there are numerous criteria that can be used to build, analyse and compare data quality methodologies. Levis et al. conducted literature review in information quality comparing numerous DQ approaches based on criteria such as “methodology”, “key concepts”, “DQ philosophy” and “DQ approach” (Levis, Helfert, and Brady 2007). Other studies have compared DQ framework according to their, ability to define data quality, dimensions and metrics they cover, tools and methods to implement DQ quality as well as costs associated with DQ initiative (C Batini and Scannapieco 2006; Ge 2009; R. Wang and Strong 1996; Lee et al. 2009). Taking into account works stated above, the following topics were found to overlap and discussed in the most of the Data Quality research:

- dimensions and metrics that are selected in the methodology
- types of data that are considered in the methodology
- types of information systems that are considered in the methodology
- phases and steps that compose a DQ methodology
- techniques and strategies of assessing and improving data
- types of costs that are accompanying data quality issues

The analysis and comparison of Data Quality frameworks are derived based on categorisation drawn after conducting a systematic literature review (SLR). In its core, literature review aims to identify trends and possible gaps in the domain of the DQ. SLR is divided into two activities - broad and advanced literature search. For initial examination of the DQ literature, board study was performed. It concentrates on finding a wide variety of research relevant materials. The main focus is to gather paper abstracts, conclusions, prefaces, and references of the found materials in order to collect as many potential leads to relevant information as possible.

Dimension and metrics are identified as the most discussed elements in every DQ methodology. DQ dimensions and metrics are concepts that theoretically describe the quality of information from various points of view. Examples for DQ dimensions are data accuracy, data timeliness, data consistency, etc. It is possible that multiple metrics can be associated with each quality dimension (e.g. in real life, dimension *weight* can be measured in *kilograms*, but also in *grams*). Types of data criterion investigate the form of data that has been considered in observed methodology. Data could be structured, semi-structured and unstructured. The type of information system and the application domain of observed DQ methodology is another important factor. Types of IS are, for example, monolithic, distributed, web, etc. and DQ application domains – healthcare, e-commerce, military, etc. Probably the next most crucial element, along DQ dimensions, is the “phases and steps”. This criterion involves the sequences of activities (e.g. data and process analysis, measurement of quality, improvement steps, etc.). Another factor taken into consideration while investigating the variety of methodologies is different techniques and strategies accompanying stages of assessment and improvement. It is important to distinguish between “phases and steps” and “techniques and strategies” criteria. The techniques and strategies can be observed as ways (tools) for accomplishing goals in different phases of the overall DQ process. For instance, assessment techniques commonly considered in DQ literature are “questionnaires”, “record sampling”, “record matching”, etc. while “normalization”, “record linkage”, “data integration”, etc. are methods for improving information quality. Lastly, despite being of not great focus for this literature review, the topic for the costs associated with data quality issues is slightly discussed. English, (1999) classifies the cost of poor quality into two categories: process costs, such as costs associated with the re-execution of the whole process due to faulty data; and costs associated with poor data itself. The costs of bad data quality are strongly contextually dependent (e.g. particular application) in comparison with the cost related to the DQ process. For example, in healthcare and military, mishandled information can cost not only billions of dollars but also human lives. Table 2-5 presents a summary of identified criteria.

Table 2-5: DQ Criteria Summary

Criteria	Type	Description
Phases and Steps	Definition	Data analysis - examines data schemas and aims for understanding the data
		Process analysis - provides model for processes producing and updating data
	Assessment	DQ requirements analysis - setting the new quality targets
		Identification of critical areas, which selects the most appropriate databases and data flows to be assessed quantitatively and measuring stage where metrics are defined and assessment is executed
Types of Data	Improvement	Steps for evaluation the costs and strategies and techniques for improvement.
	Structured	A collection of data components that are constructed in a regular and characteristic way
	Semi-structured	Data which have structure but also has some degree of flexibility
	Unstructured	Generic sequence of symbols typically defined in natural language
Types of IS	Monolithic	Composed of single tear where separate applications do not share data.
	Data warehouse	Centralized data collections are built from multiple databases
	Distributed IS	A collection of modules, typically divided in tiers, such as presentation, logic, and data tear, coordinated by a workflow
	Cooperative IS	large scale IS that can be defined as a large-scale information system that communicate with multiple systems of different and organizations sharing common objectives
	Web IS	Using web technologies and client/server way of transferring data
	Peer to Peer (P2P)	IS, where there is no distinction between clients and servers
Techniques and strategies	Assessing	Assessment methods (i.e. questionnaires, sampling, matching etc.)
	Improvement	Improvements methods (i.e. normalization, record linkage, data integration, process redesign (process driven), etc.)
Types of Cost		Costs associated with poor data quality
		Costs of assessment and improvement activities

The identification of the criteria listed above is also aligned with the purpose of the literature review – 1) to identify possible withdraws of current data quality method-

ologies; and 2) to assist constructing a DQ method that would be applied in SOA context.

Throughout the course of literature review, a number of frameworks were identified to be the most significant and relevant to the in Data Quality research. Table 2-6 presents a summary of these DQ frameworks/methodologies along with their authors and abbreviations. Note that table lists only the most relevant works. However, more works are discussed throughout this chapter. Detailed description about the criteria is given in next sections as well.

Table 2-6: Data Quality Methodologies

Authors	Name
(Yair Wand and Wang 1996)	Total Data Quality Management (TDQM)
(Jeusfeld, Quix, and Jarke 1998)	Data Warehouse Quality methodology (DWQ)
(English 1999)	Total Information Quality Management (TIQM)
(Lee et al. 2002)	A Methodology for Information Quality assessment (AMIQ)
(Loshin 2004)	Cost-effect Of Low Data Quality Methodology (COLDQ)
(Scannapieco, Virgillito, and Marchetti 2004)	Data Quality in Cooperative Information Systems (DaQuinCIS) Comprehensive methodology for Data Quality management (CDQ)
(Su and Jin 2006)	Activity-based Measuring and Evaluating of product information Quality methodology (AMEQ)
(Long and Seko 2002)	Canadian Institute for Health Information methodology (CIHI)

2.5.1 Types of Information Systems and Application Contexts of DQ

Generally speaking, application context of a data quality methodology is tightly related to a particular domain of information system. The term domain (or context) in sub-chapter comprises two perspectives – (1) type and (2) application of the information system. Literature provides a variety of classification for types of information

systems. Among the most common are data warehouse, monolithic, distributed cooperative, web, and peer-to-peer information systems (C Batini and Scannapieco 2006). On the other hand, the application of those types could be any area of knowledge e.g. healthcare, education, military, business, etc. In most of the cases, researchers in data quality develop their own methodologies for a particular type or types of information system and then they apply it to a specific case in order to evaluate their work.

The most widespread type of the information systems in the 70-90s were **the monolithic information systems**. In these systems, applications are single-tiered - meaning that data and application logic are merged into one whole entity. The majority of the DQ methodologies can be applied to this type of systems, but there are few who are specifically focused on single tier systems e.g. Loshin, (2001); Y Wand & Wang, (1996); etc. Later, after the invention and popularization of the Internet in the 90s, a shift from monolithic to network-based, **distributed information systems** had begun. These systems comprise more than one tier – typically presentation, application and data tiers. With the increase of the complexity and size of such systems, issues with distributing poor data across systems had begun to get out of control. To fill the gap, researchers in information quality had developed various solutions, for instance, R. Wang and Strong (1996); Long and Seko (2002); English (1999) frameworks . Another type of IS that is very closely related, and in fact part of the distributed systems, are **cooperative information systems**. A cooperative information system is a system that interconnects multiple autonomous systems that share common goals. Data challenges in such systems involve data exchange and integration. Batini and Scannapieco (2006); Scannapieco, Virgillito and Marchetti (2004) deliver frameworks that offer several services and tools for aforementioned systems. Along with the developing of distributed and cooperative IS, another type of IS e.g. **Data Warehouse (DW)** had emerged. Data warehouses are collections of data retrieved from multiple databases. The huge number of data that is processed and stored, as well as the rate of changing the information, are typical challenges in such systems. With this regard, several solutions are proposed, for example, The Data Warehouse Quality (DWQ) methodology proposed by Jeusfeld, Quix, and Jarke (1998).

Another type of information system that deserves particular attention, and where Data Quality could be applied, is **Web Information Systems**. Web Information Systems are newly emerged (i.e. last 20 years) type of information systems that uses Internet and Web technologies to provide information services to users and/or other information systems/applications (e.g. desktop or server/database applications). The main feature of this type of IS is that it make the information accessible to the users/applications anywhere in the world by using multimedia and transportation technologies such as HyperText Transfer Protocols (HTTP), HyperText Markup Language (HTML), JavaScript, etc. A typical web-based information system consists of front-end and back-end component. Web browsers, for example, are used as front-end element to visually represent service's content whereas the logic of the application, as well as its database, are considered to be part of the back-end component. The domain of Web Information Systems is relevantly new in comparison with information quality. As a result, there are several issues that challenge DQ researchers when it comes to the Web – the problem with the huge number of unstructured documents; and the need of introducing new types of quality dimension, such as *accessibility*, and *reputation*, for instance. Accessibility measures the ability of users to access data, given their culture, physical status, and available technologies. Reputation (or trustworthiness) is a property of data sources measuring their ability to provide correct information and is particularly relevant in Web-based systems. Despite the lack of exhaustive research in the area, however, there is still a good number of research works that aims to solve some of the problems listed above.

Numerous methodologies for assessment and evaluation of specific qualities websites are delivered by Pernici and Scannapieco (2003), Cappiello, Francalanci, and Pernici (2003), Katerattanakul and Siau (2002), Calero, Caro, and Piattini (2008), and Aladwani and Palvia (2002). These works aim to assess web DQ according to dimensions such as accuracy, timeliness, consistency, proposed by early studies (e.g. R. Wang and Strong (1996); Loshin (2001); R. Wang et al. (1998)) but also to extend them by adding dimensions only applicable for Web systems. For example, Pernici and Scannapieco (2003) introduce a framework that associate information quality dimensions with elements of the web page, the page itself or group of pages. Authors

extend previously proposed methodologies for data quality design and management in Web information systems by considering dimensions such as *volatility* and *compleatability* (Mecca, Atzeni, and Masci 1998). Another stream of research is aiming of assessing the quality of web portals. Calero et al., (2008) propose DQ tool for evaluating web portals. The methodology focuses on assessing quality from the point of view of data consumers. In this relation, Aladwani & Palvia, (2002) developed an instrument for validation and measuring user-perceived web quality. The instrument measures four dimensions of web quality namely specific content, content quality, appearance, and technical adequacy.

The quality of Web content is of growing importance since the number of documents presented using Web technologies is rapidly increasing. Data from several studies i.e. (Wren 2008) have indicated that 40% of the web content disappears within one year, whereas additional 40% is modified, leaving only 20% of its original form. On the other hand, Brewster Kahle has suggested that the average life-span of a web page is 44 days and the web content changes completely few times a year. As a result, preservation of Web content becomes more and more important. To reflect that, Capriello et al. (2003) propose a data quality model that supports preservation of Web data in order to increase its quality. The term *Web preservation*, or also as known as *archiving*, is a process of storing all significant versions of a web page in order to prevent loss of information on the web. Finally, Information Quality Measurement (IQM) framework is strongly focused on Web information systems, as it considers a wide set of existing tools to evaluate information quality in the Web context. Such tools are site analysers, traffic analysers, port scanners, performance monitoring systems, Web mining tools and survey tools to generate opinion-based user feedback (Eppler and Muenzenmayer 2002). Several information quality criteria can be measured with the help of these tools. IQM provides systematic sequential steps to match information quality criteria with measurement tools. This framework uses some of TDQM dimensions such as accuracy completeness and consistency.

A summary of most common data quality frameworks in the Web information systems, developed in the last decade, is presented in Table 2-7: Data Quality Research in Web Information Systems”.

Table 2-7: Data Quality Research in Web Information Systems

Author	Sub Domain	Model/Framework
Pernici & Scannapieco, 2003	Web IS general	4 categories, 7 activities of DQ design and architecture to DQ management
Katerattanakul & Siau 2001	E-commerce Web	4 categories associated with 3 categories of data user requirements
Cappriello et al. 2003	Web IS general	5 categories/dimensions including volatility and accessibility; 3 main phases
Calero et al. 2008	Web Portals	4 categories/dimensions; 33 attributes Consumer oriented data assesment process
Aladwani & Palvia, 2002	Web Page	4 categories/dimensions; 25 attributes User-perceived web quality
Eppler and Muenzenmayer, 2002	Web Portals	16 IQ –criteria and indicators Practically oriented, supporting tools

Another application domain of Data Quality is **Peer-to-Peer (P2P) systems**. In this type of systems, there is no distinction between client and servers. P2P are a type of co-operation systems comprised of components (nodes) that characterise with no central coordination and no central database. DQ issues, identified by researchers this domain, are commonly related with trustworthiness (i.e. reputation) of the components (nodes). DaQuinCIS (Data Quality IN Cooperative Information Systems) (Scannapieco, Virgillito, and Marchetti 2004) is a data quality framework for peer-to-peer cooperative information systems that provides a *rating service* in order to measure the reputation of the nodes. The *rating service* module is centralized and its

function is to associated trust values to each data source in the system. The trust values, used to determine the reliability of the source are represented with different metrics. The DaQuinCIS architecture is illustrated in Figure 2-9. Other researchers, e.g. (Gackowski 2006), propose techniques for measuring and improving the credibility of data values produced form different nodes.

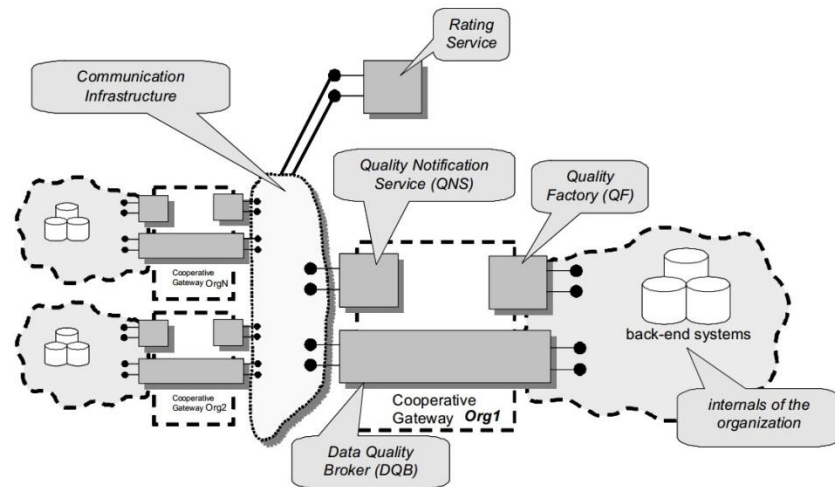


Figure 2-9: DaQuinCIS Architecture (Scannapieco, Virgillito, and Marchetti 2004)

With respect to the application context of data quality methodologies, literature provides some examples. For example, the use of TDQM methodology had been reported by U.S.A. Department of Defense (DoD), including its Medical Command for Military Treatment Facilities (MTF) (R. Wang and Strong 1996). In addition, Kovac and Weickert, 2002 had evaluated TDQM in a Start-Up organizational context by using the methodology to provide a tool for their online retail shopping information system. Other applications of TDQM include health care and insurance companies (Nadkarni 2006). English's (1999) TIQM methodology has been applied in variety cases including customer relationship management (CRM) systems, telemarketing, and healthcare. Another very specifically evaluated methodology is CIHI (Long and Seko 2002). Other methodologies such as methodology for the Quality Assessment of Financial Data (QAFD), and Comprehensive methodology for Data Quality management (CDQ) are reported to be applied in financial data analysis and government areas respectively.

Despite some Data Quality methodologies are evaluated in particular contexts, literature does not present a full spectrum of DQ applications. Also observations, in this research, have shown that most of the DQ solutions are not empirically evaluated or assessments are carried out on very large-scale contexts. As a result, the evaluation process of most data qualities remains ambiguous or unreported due to data protection policies.

2.5.2 Data Quality Dimensions

The area of Data Quality scopes problems related with data and its definition, measurement, analysis, as well as methods and tools for its improvement. In all DQ methodologies, the definition of qualities, dimensions, and metrics is essential activity. Particular attention is paid to data quality dimensions as they are the main constructive block of every DQ methodology. With this respect, data quality literature delivers a comprehensive identification, definition, and classification of the DQ dimensions.

One of the notable pioneers to deliver a study that defines and classifies DQ and its dimensions were Wand and Strong (1996). In their work, they observe the information quality through three different angles namely intuitive, theoretical and empirical angles. These points of view can be considered as three approaches for defining data quality dimensions. The intuitive approach derives information quality dimensions from the researchers' experience or from the requirements of the particular case. The theoretical approach defines data quality dimensions from the real-world perspective. The empirical approach provides information quality dimensions by considering whether or not data is fit for use by the information consumer. Figure 2-10 illustrates the points of view proposed by Wand and Strong (1996).

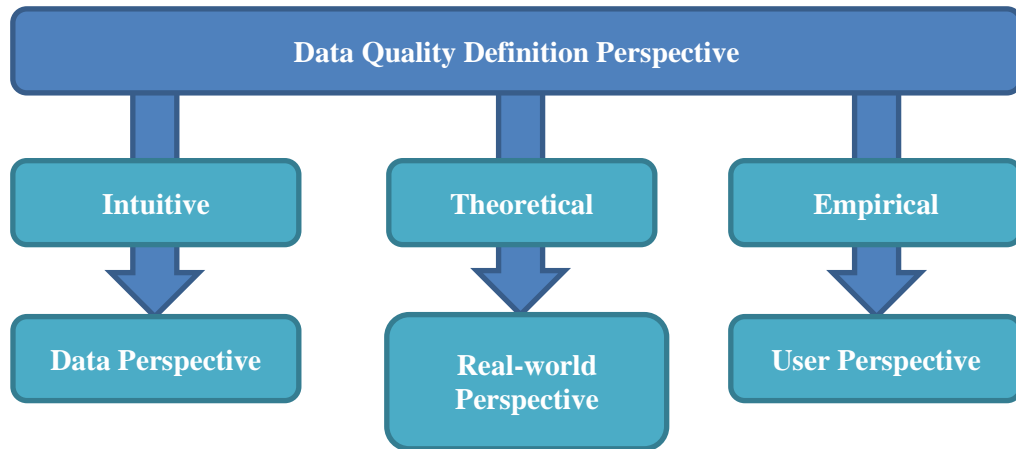


Figure 2-10: Points of View for defining Information/Data Quality dimensions

In order to explain more in depth DQ dimensions that are taken into account in this research as well as various problems associated with them, a semiotic framework for understanding data quality, initially proposed by (Stamper et al. 2000), is accommodated. The use of this theory is derived from the fact that it gives a comprehensive description of the structure of information.

2.5.2.1 Semiotic Theory

Numerous researchers have utilised semiotic theory to express their philosophical position and describe data quality dimensions (Ballou et al. 1998; Stamper et al. 2000; Shanks and Darke 1998; Helfert 2001). Semiotic theory concerns the process of analysing the symbols. It can be used to understand technical and social aspects of information within composite environments. Stamper et al. define six groups, namely physical, empirical, syntactic, semantic, pragmatic and social group (Stamper et al. 2000). The framework of Shanks and Darke is based on semiotic theory and uses four of the semiotics levels defined by Stamper. Despite general agreement among researchers, only three semiotic levels are relevant and discussed in this section; these are the syntactic, semantic and pragmatic groups. This decision is based on the scope of this research – to focus on practical application of DQ rather than the theoretical only.

Syntactic. Syntactic is area related to the logic and grammar of the information. Syntactic data quality is concerned with the structure of data. The objective at this level is to present data consistently and completely in the virtual world. This level comprises “consistency” and “completeness” data quality dimensions and problems related with these. Problems at this level are particularly important in the service-oriented context. Nonetheless, many studies have been conducted to tackle such types of issues - at both technical and organizational level (Carlo Batini and Scannapieco 1998; English 1999).

Semantic. Semantics is an area of study which deals with the interpretation and the meaning of data. The data quality dimensions from the research literature operating at semantic level are associated with “accuracy” and “awareness of bias”. Accuracy is an instance of data quality concerned with how well information reflects the state of the real world. Inaccuracy is growing problem when it comes to composite environments. For example, data accuracy $DQ_{acc[S]}$ (the degree of conformity with the real world) of a data subject S is measured by taking the number of the returned data in-compliant statements N_s divided into the total number of tests T_s for the chosen data subject S , and then multiplied by 100 to convert that in percentile of accuracy. This is shown in equation 1.

$$DQ_{acc}[s] = \left[1 - \frac{N_s}{T_s}\right] * 100 \quad (1)$$

Awareness of bias is another dimension that is defined for the purpose of this research. Problems related to the lack of shared knowledge and understanding among disparate parties is a common issue associated with this dimension. Despite in theory these dimensions are widely addressed and supported by many DQ methodologies (English 1999; Yair Wand and Wang 1996), in practice not many efforts have been done to solve the problems concerned with them.

Pragmatic. Pragmatics deals with the usage of the information by services. The contextual aspects of pragmatic issues are addressed by the appropriateness and relevance of the information. Dimensions which link to these issues are “timeliness” and

“currency”. These dimensions are concerned whether information suits to the problem task. By doing so, they are not directly associating with data quality. That is to say, these dimensions do not address explicitly the issues of data quality but rather they deal with problems concerning its usability.

The association between semiotic groups and data quality problems and dimensions is presented on Table 2-8.

Table 2-8: Summary table with quality dimensions on different levels presented with examples

Semi- otic level	DQ Di- mensions	Problem description	Example	Data Het- ero- genei- ties
Syntac- tic	Consistency	Same data values are represented differently among different parties/services	The customer type is an <i>alpha code</i> in one system but a <i>numeric code</i> in another. The day May 1 2013 is represented as <i>1/5/2013</i> or <i>05-01-2013</i> by other parties/services	Representational
	Completeness	Data does not meet fully its requirements and it is not sufficient for its intend of use	The address Grace Park Heights, Ireland is missing <i>Street number</i> and <i>City name</i>	
Seman- tic	Accuracy	The data value do not correctly reflects the real-world condition.	The temperature sensor measures a temperature of 20C while in fact the temperature is 15C. The address 666 Royal St in Greenland actually does not exists.	Ontological
	Awareness of Bias	Lack of shared understanding of meaning among different parties /services	The concept <i>Profit</i> is perceived as <i>gross profit</i> by one of the parties/services, as <i>net profit</i> by another and as addition of <i>gross and net</i> by third party	
Prag- matic	Timeliness and Cur- rency	The entity does not represent the most current information resulting from the output of a service operation	The customer order system has updated the address entity, but the customer master system does not.	Tem- poral

Among semiotics, another line of studies identifies and catalogue various data heterogeneities (Naiman and Ouksel 1995; S Madnick and Zhu 2006). Taking into account heterogeneities that are related to data quality, in this research, three main groups can be categorized and addressed: (1) representational heterogeneity and (2) ontological heterogeneity. Data semantics can sometimes change over time; therefore, the representational and ontological semantics of a source or a receiver can evolve, resulting in (3) temporal heterogeneities. These categories are put into context in Table 2-8 as well.

- Representational heterogeneity – The same data can have different representations in different sources and receivers. However, it represents the same real-world object.
- Ontological heterogeneity – The same term is often used to refer to similar but slightly different objects. Known and quantifiable relationships often exist amongst these objects. In this case, same data can refer to different real world objects.
- Temporal Representational heterogeneity – The representation and meaning in a source/provider or a receiver/consumer can change over time. For example, a price database in Turkey may list prices in millions of Turkish liras (TRL), but after the Turkish New Lira (TRY) was introduced on January 1, 2005, it may start to list prices in unit of Turkish New Lira.

2.5.2.2 Quality of Service and Data Quality Dimensions

Another concept that has to be mentioned that explicitly relates to Data quality and Service oriented compositions is the Quality of Service (QoS). QoS is represented by the service contract which defines functional and non-functional requirements to ensure service interoperability. A service contract is a long-term relationship between service providers and service users on the QoS policy, also referred to as service level agreement. Given the focus of this research, the difference (and relationship) between data quality and QoS should be clarified. On the one hand, Data quality indicates the degree of excellence within the data, its state of completeness, validity, and accuracy that enables it to perform further functions. This, in turn, enables the user to obtain

the necessary information required for operational reasons or to assist in decision making and planning. Data of high quality produces results that need to be reliable and correct. In the service composition context, data quality needs to be accurate and in reliable formats. Ideally, the SOA infrastructure should not interfere with data on this level and, if it is not explicitly required, leave quality assurance to customers, or data users. On the other hand, the quality of service is related to the quality of its internal data such as configuration management records and measurements of service resources usage. Within SOA community this is previewed as meta-data describing service capabilities. In relation to semiotic theory, the different levels which are used to categorize data quality dimensions in Table 2-8 are similar to the levels used to structure system / service interoperability. This is logical since poor data quality in each of the mentioned dimensions would lead to degraded usability of data at the receiving end, hence poor interoperability. QoS properties that are related to Data quality are service *reliability* (e.g. Minimal Time between Failures or Min Time Before Recovery), *responsiveness* (e.g. time needed to for the provider to process and return consumers request), *volume* (e.g. throughput of data in a process flow), *cost* (fee paid by consumer), etc. For example, Timeliness category contains dimensions relating to the end-to-end delay of data flow. Such delay depends on several aspects of the SOA system and service capabilities. One of those factors is the number of running service instances. This number can be changed based on information about system load and resources usage provided by SOA system and service provider. If such information is not fresh (up-to-date) it might be not possible for the SOA to take quick and responsive decisions. Delayed decisions can easily lead to lower QoS. In particular, the number of active instances may not be sufficient to meet assumed response time and to satisfy end-users' expectations.

Data Quality community provides a wide variety of descriptions of quality dimensions – some of them, e.g. (Lee et al. 2002; Long and Seko 2002) better defined metrics that allow to measure and compare features of data sets. However, DQ dimensions lack consistency in definition since the same dimension maybe understand in a different way or the same feature may be called differently by two researchers. This problem has been noticed and addressed by Wang and Strong and further discussion

on this topic will be omitted. In this study, the focus is only on the semantic semiotic level (see Table 2-8 the highlighted section) and aims to solve the problem with poor semantic (e.g. inaccurate) data within service-oriented compositions – e.g. how to detect such data. The dimension and metric were defined previously in this chapter. To conduct objective assessment of the accuracy, other dimensions such as completeness and consistency are considered with a low correlation to the measured dimension. They also are kept stable and believed to have high quality values.

2.5.3 Data Types

The supreme objective of any Data Quality process is to analyse and evaluate the quality of data in an IS. The concept of data itself involves a digital representation of real world objects that can be later processed and manipulated by software procedures through a network. In the field of information systems and data quality, most of the authors differentiate in one way or another three groups of data – a) structured; b) unstructured; and c) semi-structured. On Figure 2-11 the same real world object is stored in three different ways.

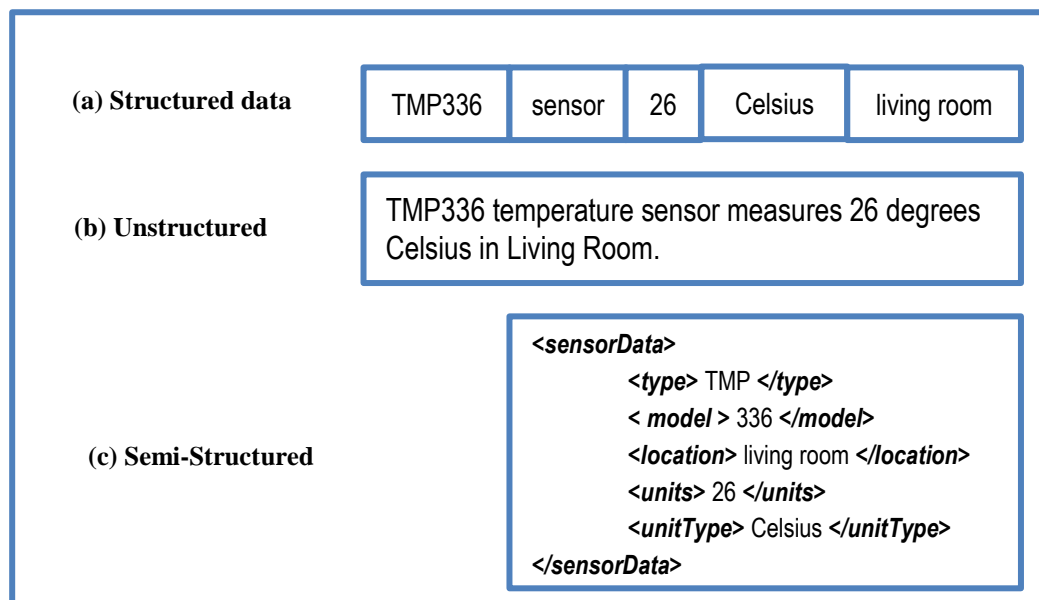


Figure 2-11: Types of data. Different syntactical representations of the same real world object

a) structured data is an item or group of items described by simple attributes defined within a domain. Domains define the type of values that can be assigned to an attribute. These types typically correspond to the programming languages' data types, for instance, Integer (whole numbers), Boolean (true/false), String (sequence of characters), etc. Figure 2-11(a) presents a record (row) of a table containing structured information from a sensor reading. If we consider the value '26' as a data item, then its attribute is 'temperature' and it is defined in the numeric (Integer) domain. Essentially, almost all DQ methodologies considered in this research (see Table 2-6, section 2.5) could be employed in systems using structured data.

b) unstructured data is a generic sequence of symbols or/and groups of symbols grouped in words and sentences. That is to say, unstructured data is presented in any natural language, such as English. Examples for such are any news articles, web pages, blogs, e-mails, etc. Figure 2-11(b) represents a log state expressed in natural language. Unstructured data could not only be text but also images, voice recordings (e.g. interviews), video clips, etc. A huge amount of unstructured data is available on the Internet nowadays and the need of analysing and processing such represents a major challenge for researchers. Data quality techniques become increasingly complex and inefficient as data lose structure. For instance, considering the example with structured and unstructured data in Figure 2-11, the same quality dimension will have different values according to the type of data and the final consumer. For example, accuracy will have different values if the consumer is machine than if the consumer is human being. The accuracy of structured data will be scored with higher mark in case the consumer is a machine. Counter wise, the unstructured data will get higher points for accuracy (e.g. more complete), if the consumer is a human being. Solutions in analysing unstructured data have developed in the past decade but solid research is still undergoing. In order to assess information of unstructured text, data needs to be analysed first. McCallum (2005) describes the process of analysis as a combination of information extraction and data mining techniques. Information extraction is a process of populating a data base from unstructured or loosely structured text. Data mining techniques are then applied to discover patterns that can be later assessed. Information extraction consists of five substantial sub tasks – segmentation, classifica-

tion, association, normalization, and reduplication (see Figure 2-12). To cope with the vast amount of information and make the process automatic and more efficient, extracting tools often have to accommodate various machine-learning methods such as Hidden Markov Model (HMM).

c) semi-structured data is a kind of structured data but in the same way flexible in comparison with traditional structured data (Figure 2-11.(a)). Flexible means that it does not adapt the formal structure of data models associated with relational databases or data tables. Semi-structured data is commonly referred as self-describing or schemaless (Abiteboul, Buneman, and Suciu 2000). A widely spread way to represent such data is the Extensive Markup Language (XML). The goal of the XML is to provide both human- and machine- readable documents. To do so, it defines sets of rules, markers, to separate semantic elements and enforce hierarchies of records and fields within the data (W3Consortium 2006). Alternative to the XML document formats is another technology called JavaScript Object Notation (JSON). JSON is an open-based format that transmits in a read-friendly data objects consisting of attributes-values pairs. This standard has been promoted by the emerging of the web services and utilized by REST. Some of the characteristics of aforementioned standards are that: 1) data can contain field unknown at design time; 2) same data entry may be represented in different ways. For instance, considering the example in Figure 2-11, the name of the sensor include ‘Type’ and ‘Model number’; depending on the situation this name could be stored in a single field (a concatenated type and number – see Figure 2-11(a)) or multiple fields (e.g. separate field for each element – refer to Figure 2-11(c)); and 3) fields might have empty values, due to uncertainty during design time. Methodologies, such as TDQM (Yair Wand and Wang 1996), CIHI (Long and Seko 2002) and DaQuinCIS (Scannapieco, Virgillito, and Marchetti 2004) takes into account semi-structured data and proposes assessment methods and techniques for handling inadequate information.

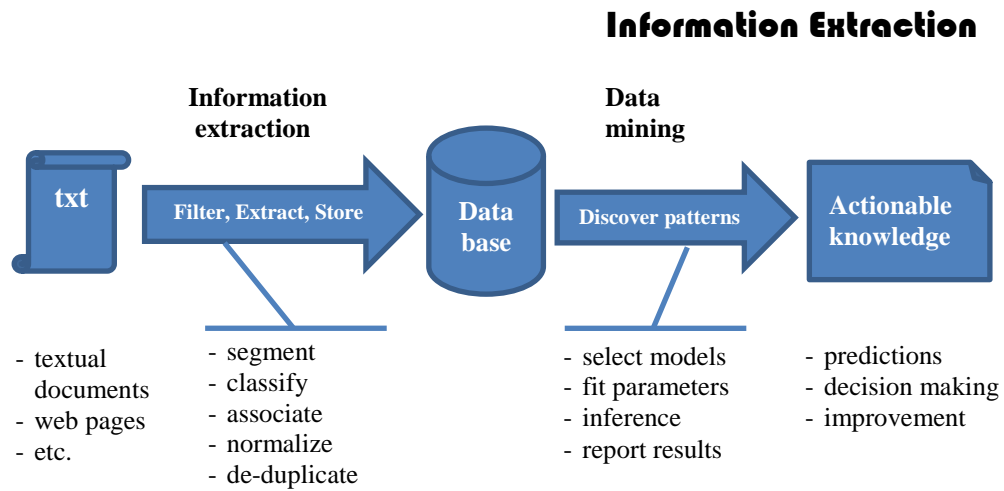


Figure 2-12: Extraction of unstructured data - process and methods

2.6 Summary of Data Quality Methodologies

Based on the analysis presented in previous sections, two comparison tables containing some of the most relevant DQ methodologies are presented further below. Table 2-9 compares approaches considering dimensions, data types, information system type and application criteria, while Table 2-10 depicts different phases and steps of each DQ methodology. Based on Table 2-9 several conclusions can be made. Firstly, with respect to data type criteria, most of the methodologies clearly target structured data type. CIHI (Long and Seko 2002), DaQuinCIS (Scannapieco, Virgillito, and Marchetti 2004) and partly TDQM (Yair Wand and Wang 1996) support semi-structural data types in their profiles. Tied with the data types, the most supportive IS type by methodologies is monolithic, however a few of them concentrate on distributive and cooperative e.g. TIQM (English 1999), CDQ (Carlo Batini et al. 2009), DaQuinCIS; Data warehouses - DWQ (Jeusfeld, Quix, and Jarke 1998) as well as Web IS (see Table 2-7). With reference to DQ dimensions criterion, most of the methodologies define a large number of dimensions. Still, metrics reflecting these dimensions remain vaguely explained, i.e. COLDDQ (Loshin 2001) has over 30 dimensions but only 6 metrics. Consequently, a broad difference in focus across methodologies can be identified. On the one hand, some are more comprehensive but

more general. On the contrary - others are obviously more focused but not that extensive and flexible. TDQM is a project aimed at providing an empirical foundation for data quality. Wang and Strong have empirically identified fifteen IQ criteria regarded by data consumers as the most important. The authors classified their criteria into the classes “intrinsic quality”, “accessibility”, “contextual quality”, and “representational quality”. The classification is based on the semantic of the criteria. It is of use to describe the criteria but not to assess them. Thus, this classification is semantic-oriented (Naumann and Rolker 2000). Semantic-oriented is approach if it is solely based on the meaning of the criteria. This classification is the most intuitive when criteria are examined in a most general way, i.e., separated from any information framework. DWQ project, on the other hand, is based on the criteria of TDQM. The authors define operational quality goals for data warehouses and classify the criteria by the goals they describe. These are accessibility, interpretability, usefulness, believability, and validation. Their approach this classified as goal oriented rather than semantic oriented. A classification is goal oriented if it matches goals that are to be reached with the help of quality reasoning (Naumann and Rolker 2000).

Table 2-9: DQ methodologies’ comparison – Dimensions, Data Types, IS types and Application Contexts

Criteria Author	DQ Dimensions	IS Type Focus	Application Context	Data Type
Wang 1998 TDQM	16 dimensions in 4 categories 5 metrics	Monolithic	General; Military, Healthcare, Finance	Structured Semi- structured
Jeusfeld et al. 1998 DWQ	15 dimensions 5 metrics	Data warehouse	General	Structured
English 1999 TIQM	2 categories – inherit and prag- matic, 15 dimensions 8 metrics	Monolithic, Dis- tributed	General; Customer Relation- ship management, healthcare, etc.	Structured
Lee et al. 2002 AIMQ	14 dimensions 17 metrics	Monolithic	General	Structured

Long and Seko 2005 CIHI	16 dimensions 14 metrics	Monolithic, Distributed	Healthcare	Structured Semi- struc- tured
Loshin 2004 COLDQ	30 dimensions in 4 categories 6 metrics	Monolithic	General	Structured
Scannapieco et al. 2004 DaQuinCI	5 dimensions 3 metrics	Cooperative, peer-to-peer	General e.g. Inter organizations, Busi- ness Finances, etc.	Structured Semi- struc- tured
see Table 2-7	Various – see Table 7, section 2.5.1	Web	Various; Web pages, Web systems, etc.	Structured, Semi- structured, Unstructured
Author Criteria	DQ Dimensions	IS Type Focus	Application Context	Data Type

A phase and steps criterion is the most diversified one. A summary of the approaches can be observed in Table 2-10. Most of the methodologies measure the quality of data subjectively - through questioners and surveys targeting different user groups. On the other hand, DQ frameworks such as TIQM and COLDQ (English 1999; Loshin 2001) employ different statistical and data mining methods in order to objectively measure information quality. Furthermore, only a few (e.g. DWQ (Jeusfeld, Quix, and Jarke 1998)), methodologies consider DQ requirements analysis step. Likewise, limited methodologies i.e. TDQM, TIDQM, COLDQ (Yair Wand and Wang 1996; English 1999; Loshin 2001) take into account process analysis step in the definition stage as mandatory. With respect to improvement stage, most of the methodologies except AIMQ (Lee et al. 2002) and AMEQ (Su and Jin 2006) support evaluation of the cost and different improvement techniques. COLDQ methodology exclusively accentuates on the evaluation of cost as well as strategies for process and data improvement (i.e. process optimization and redesign).

Table 2-10: DQ comparison – Phases and Steps

Phase	Author Step	Wang 1998 TDQM	Jeusfeld et al. 1998 DWQ	English 1999 TIQM	Lee et al. 2002 AIMQ	Eppler et al. 2002 IQM	Loshin 2004 COLDQ	Scannapieco et al. 2004 DaQuinCIS	Su and Jin 2004 AMEQ
Definition	Data Analysis	data consumers, suppliers, manufacturers and managers perspectives	supported	supported	supported	supported	supported	supported	supported
	Process Analysis	supported using IP- UML	n/a	supported	n/a	n/a	supported	supported	n/a
	DQ Requirements	n/a	stakeholders surveys	based on consumers satisfaction	n/a	n/a	n/a	n/a	interviews
Assessment Measuring	Finding critical areas	supported	supported	supported	supported	supported	supported	supported	supported
	Measuring quality	subjective surveys	hierarchical quality assessments	random data samples	questionnaires	data gathering; data analysis	objective and subjective	objective and subjective	questionnaires
Improvement	Evaluation of the costs	supported	supported	n/a	n/a	n/a	supported	n/a	n/a
	Strategies and techniques	Process driven – process redesign	Data driven – schema, data integration Process driven Process redesign	Data driven; Data cleaning, and correction normalization Process redesign	n/a	n/a	Cost optimization process redesign Process control	data driven source trustworthiness; record Linkage	n/a

2.7 Research Themes

The literature review analysis above sketched the boundaries of the area of investigation for this research. Subsequent research questions are organized as to lead the project to successful development and execution. Consequently, successfully conducted research will contribute to the body of knowledge, in the academia, as well as to practice by applying it into the real world.

In this research, the objective is to investigate how to assess and analyse data quality within Service-oriented context. In order to reach given objective, assessment framework will be designed and evaluated. More specifically, this framework will be represented by methods, a methodology, guide for assessing data quality in SOA and it will be, ultimately, implemented as a software tool in a particular SOA case. Drawing upon the literature review, research questions have been framed in order to fill the gap stated in Section 1. Taking into account the problem statement, three main research questions targeting the scope of the problem, the design of the framework, and the evaluation of the outcomes are formulated.

To create and evaluate valuable artifacts, it is important to define suitable requirements for an effective data quality assessment methodology that considers SOA context. Moreover, these requirements will also be taken into account for evaluation purposes. The challenges and questions surrounding scoping of research problem are summarised in Table 2-11.

Table 2-11: Research questions concerning scoping the problem

Research Question Concerning Scoping the Problem	
RQ 1	What are the requirements needed for applicable Data Quality assessment in the service-oriented context?
RQ 1.1	What are main criteria for DQ methodology and how different DQ methodologies compare against them?
RQ 1.2	What are the key principles embedded into Service-oriented Architectures that affect Data Quality?

Designing and developing the artifact must obey the requirements obtained from the first research question. To achieve this, the following second research question and its sub-questions need to be answered, as summarised in Table 2-12.

Table 2-12: Research questions concerning artifact design

Research Question Concerning Artifact Design	
RQ 2	How can be Data Quality measured in a Service-oriented Architectures context?
RQ 2.1	How to define “poor” Data Quality within the context of SOA?
RQ 2.2	How to detect poor data in service-oriented environment?
RQ 2.3	How to design Data Quality assessment method?
RQ 2.4	How to implement the Data Quality assessment method?

The evaluation of the research outcomes will follow the identified requirements and apply in different (real or artificial) cases. Additionally, based on the design and implementation of the Data Quality assessment methods, an instantiated scenario will be developed. To address these objectives, a third group of questions arises as shown in Table 2-13.

Table 2-13: Research questions concerning evaluation of Artifact

Research Question Concerning Evaluation of Artifact	
RQ 3	How to evaluate the utility of the developed Data Quality assessment method?
RQ 3.1	How to define and evaluate effectiveness of the Data Quality assessment method within the boundaries of this research?
RQ 3.2	Does the designed assessment method meet the requirements for its research purpose?

2.8 Summary

This chapter presents an overview of the roots and foremost research in the information quality and SOA domain as well as indicates challenges for future research. Despite the fact that many researchers have contributed a large amount of models, frameworks, and assessment approaches over the last two decades, as this review showed, there remain many open research questions. In this thesis, efforts are focused towards building and incorporating data quality mechanism in service oriented context for detecting inaccurate business (semantic) data.

Based on my review of data quality and SOA research, several conclusions can be drawn with regard research question 1 (RQ1):

- This chapter reaches to the answer of research question RQ1.1 - *What are main criteria for DQ methodology and how different DQ methodologies compare against them?* by conducting a systematic literature review in the area. The output of this review was to identify the most relevant DQ frameworks such as TDQM (R. Wang et al. 1998), DWQ (Jeusfeld, Quix, and Jarke 1998), AMIQ (Lee et al. 2002) TIQM (English 1999) IQM (Eppler and Muenzenmayer 2002), etc as well as criteria for DQ frameworks such as type of data, types of IS, phases and steps that comprises DQ methodology, techniques and strategies and finally dimensions that are used in the methodologies to compare them. As a consequence of this analysis, a gap in the DQ literature was identified. A solution for assessing data quality effectively is needed in organizations and initiatives that follow SOA paradigm. Emerging from the literature review of data quality, few questions can be found significant for developing future DQ assessment process, these are: *What sort of particular DQ problem is tackled?*, *What dimension(s) has to be considered?*, and *What step and phases are required in order to overcome stated DQ problems?* For example, the research scope defines the need of a DQ solution that tackles *poor semantic data problems*. The dimension related to this type of problems is *accuracy*; and the steps and phases required to overcome such

problems as mainly three – DQ definition DQ measurement, and DQ improvement. The scope of this research excludes the improvement stage. Taking all these notes under consideration, in Chapter 4, an assessment framework and algorithm to assess data quality in SOA is proposed.

- To answer the RQ1.2 (see Table 2-11), the relationship between data quality application and the SOA context had to be investigated. An overview of SOA research provided numerous key principles such as Service Composability, Service Reusability, Statelessness, Autonomy, etc. that are needed for an effective service composition. However, a few SOA principles were found to be related to the quality of information. For example, service *abstraction* and *autonomy* suggest that data and process should be abstracted e.g. services hide logic and physical structure of data from the outside world. If any poor quality of information move and in and out of such “transparent” services, its detection could become extremely challenging. More thorough discussion about key factors affecting DQ was presented in section 2.2.2.1. Another concept that would also influence DQ in SOA was the Quality of Service, and it was discussed in 2.5.2.2. An example for such QoS is the “*Timeliness*” category which contains dimensions relating to the end-to-end delay of the service data flow. Such delay depends upon several aspects of the SOA system and service capabilities e.g. network delays, physical service execution time, data load, etc. Such delays could potentially lower the quality of information at the consumers’ end. On the other hand, literature confirmed there is a tight link between the DQ and the context they are applied in. This was discussed in section 2.5.1. Therefore, considering this perspective and SOA principles that affect DQ, a method for incorporating DQ assessment process into particular SOA environment has to be delivered. This method is described in Chapter 4, practically implemented in Chapter 5, and evaluated in Chapter 6.

Chapter 3 Research Methodology

3.1 Introduction

Previously, a review of the literature in the area of SOA and DQ was presented. This chapter outlines the method that was employed to help to answer the questions posed by this research. The discussion includes a rationale for the methodology chosen, a description of the methods implemented, possible limitations of the methodology and individual methods, along with a description of endeavours to overcome these limitations. In order to effectively address research challenges postulated by literature, this chapter begins with defining methodological requirements for conducting research. Then analysis of various research methods that could potentially be employed to fit the methodological requirements is conducted. This chapter concludes with presenting the research methodology accommodated in this study.

3.2 Methodological Requirements

After the research problem has been clarified and research requirements have been identified (Chapter 1), another substantial question arises: “how do we build an efficient DQ process?” On an abstract level, the solution to this problem may require incorporating a research methodology that would meet these research objectives. Taking into account aforementioned, a few research method objectives were pinpointed. Although tightly related, its important one distinguishes between ‘research objectives’ and ‘methodology objectives/requirements. Resulting from the research problem and domain (e.g. Data Quality and SOA), next paragraph builds a profile of research methodology that will reflect research objectives.

One perspective that needs to be considered while selecting a suitable research methodology is that it needs in to provide a prescriptive knowledge which would support the creation of an IS artifacts, in particular, a DQ process. An important requirement

is that this knowledge should not be accommodated from a particular company solely. Just the other way around - knowledge has to be acquired through systematic and in-depth review of high quality domain literature. In other words, research methodology should provide techniques for extracting knowledge from both literature and organization, in order to derive independency and generalization. Another objective during selection is that the chosen methodology needs to suggest appropriate methods for empirical and practical evaluation of the designed process. Figure 3-1 depicts the profile of research methodology that is considered applicable for this research. Table 3-1 in 3.3.5 summarizes these requirements, which are subsequently used to evaluate different IS research methodologies against the research methodology profile.

Answering the research question must be addressed in a rigorous and consistent manner. In general, sought methodology needs to provide process building mechanisms that would aid the developing of research solution. As a result, the question of “Which methodology is suitable in this particular case and why?” unfolds. The next section presents a detailed discussion to answer this question.

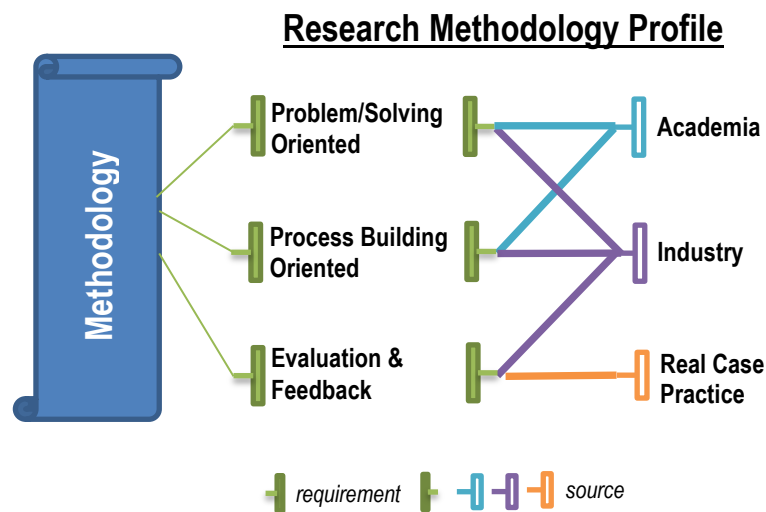


Figure 3-1: Research Methodology Profile

3.3 Methodology Selection

An important stage of this project is to identify a suitable research methodology. Instead of developing entire new research methodology, a decision to evaluate existing research methodologies that closely matches the requirements stated above and can be used as a core of the research process had been made. Observations in literature provided us with a list of five plausible research methodologies, namely Action Research (Cassell and Johnson 2006), Grounded Theory (Martin and Turner 1986), Method Engineering (Brinkkemper 1996), Case Study (Yin 1994) and Design Science (Gregor and Jones 2007; Hevner et al. 2004) which can be beneficial for this study.

3.3.1 Action Research

As one of the requirements is to validate the DQ process in real SOA case, adopting research approach suitable for combining academic and industrial knowledge is required. A seeming method commonly referred in the IS methodology literature is Action Research (AR). It originated from a work of Kurt Lewin during the 1940s and has been summarized as an approach that “combines theory and practice (and researchers and practitioners) through change and reflection in an immediate problematic situation within a mutually acceptable ethical framework” (Rapoport 1970). This definition entails a view of the methodology as an approach aiming at contributing to both knowledge and practice in terms of providing a solution to a specific entity (usually represented by an organizational setting). As a result, Action Research “is highly contextually dependent while attempting to address the specific client’s concerns” (Iivari and Venable 2009). AR focuses on solving a socio-technical problem by developing a new solution and evaluating it in an organizational context. Correspondingly, this type of research is often biased or influenced by the organization that it has been conducted within.

The objective of this study is not to solely solve an immediate organizational problem. However, the instantiation of the artifact will have to be included as part of the overall methodology, i.e. to evaluate the outcome of this research. As a consequence,

despite action research method is suitable to be ingrained in the overall methodology, it is insufficient in its original formulation to drive this research. In addition, the objective of my research is to contribute to both academia and industry. Action Research would limit the findings to the targeted organization, making it difficult to achieve generalizable results (Nandhakumar, Rossi, and Talvinen 2005).

3.3.2 Grounded Theory

Grounded Theory (GT) is seen as a powerful tool for rigorous theory development, and it is defined as “a systematic methodology involving the discovery of theory through the analysis of data” (Martin and Turner 1986). This theory is believed to be ‘grounded’ in the analysis of actual settings and processes (Urquhart 2010). In other words, the theory development process is generated starting from the participants who have experienced the phenomenon under investigation. Thus, “Grounded Theory” is a type of qualitative research, which generates a general explanation (i.e. theory) of a process, action, or interaction shaped by the views of a large number of participants. As outcome, the research questions that the Grounded Theory researcher formulates will focus on understanding how individuals experience the process of interest and identifying the steps in the process (i.e. what was the process? How did it unfold?)

Nevertheless, the outcome of that type of research can hardly deliver a process, which eliminates it as the core of this research case. Despite this fact, some of its techniques may be still used in my research. For example, one of the features that can be taken from GT its feature to refer to the theory in order to establish common vocabulary of an area and defines, with different levels of formality, the meaning of the terms and the relationships between them. As a consequence, I deem that examining existing theories would contribute (to some extent) to developing the artifact (in this research case a DQ process in SOA)

3.3.3 Method Engineering

Another requirement, when searching for a fitting research methodology is to provide techniques which will aid building processes. Method Engineering (ME) is identified

as relevant to be examined as my main output represents a method (process). ME is an engineering discipline for designing, constructing and adapting methods, techniques and tools for IS development (Brinkkemper 1996). Essentially, ME supports the creating of new methods as providing guidelines for combining already existing methods. Situational method engineering is a sub-type of ME. It seeks to develop a method that is aligned to an immediate project situation. The assembly of such methods is done by combining building blocks and guidelines in the form of meta-methods.

Method Engineering provides a versatile way of building processes. However, as a research process, it does not take into account the impact from engaging various practitioners which, in turn, may provide valuable insights to the required method. Also, it does not offer any evaluation mechanisms. Moreover, this approach might seem to be little too rigorous for the needs of building a DQ process in SOA. As it was stated before, SOA is a dynamic domain and technologies and standards changes quickly. Despite that fact, method engineering approach might be still accommodated in building the DQ process as it delivers processes through conjunction of available methods.

3.3.4 Case Study Research

Case Study (CS) research is probably the most referred method in the literature related to qualitative/quantitative research. It is been utilize in many socio and socio-technical disciplines, professions ranging from psychology and political science to education, clinical science, social work, and administrative science information systems (Orlikowski and Baroudi 1991). Numerous studies had provided definitions and insights for case study method. Yin (1994) defines case study as follows: "A case study is an empirical inquiry that: Investigates a contemporary phenomenon within its real-life context, especially when...the boundaries between phenomenon and context are not clearly evident" ((Yin 1994), p. 13). As it could be inferred, a 'case' is a *subject* of inquiry. Subjects could be people, organizations, events, places, institutions, or Information Systems IS that are studied by one or more methods. Information systems comprise of people and computer applications that processes or in-

interprets information. Additionally, Case Study research can be conducted as a single or multiple case studies. Case studies can involve qualitative or quantitative approaches of collecting data, or in many cases a mix of both. Typically, activities that characterize CS research involve exploration, generating and ultimately testing of hypothesis.

Case study research focuses on understanding an issue, problem, or phenomenon using the case as a specific illustration (Stake 2013). Thus, case study research is the approach in which the investigator explores a bounded system (a case in a specific setting/context) or multiple bounded systems (cases in specific settings/contexts) over time, through detailed, in-depth data collection, involving multiple sources of information and reports.

At first glance, CS research seems to consistently fit in the context of this project, mainly because it provides a method for an empirical evaluation of an issue in the real-life context. Moreover, Case Study delivers a worthwhile approach that provides guidelines to help to prove or disprove if an artifact (e.g. concept, theory or process) works or not. For example, in this research, the artifact is a DQ process for detecting poor data in SOA. In the first place, the evaluation of this process might require selecting the right case (e.g. what case of SOA?). Then, after a selection is made, the artifact needs to be incorporated and instantiated within it. As at that stage the artifact (DQ process) is fairly general (abstract), further details about selected specific case (i.e. home automation (a case of SOA)) is necessary. This would require accommodation various methods for data collection, for instance, discussions or code observations. The collected and analysed data would then be used as an input for instantiating the DQ process. At the end, the instantiated process would be implemented as a computer application that would measure and assess data quality in a specific type of SOA – a home automation system. Finally, the software tool would be practically assessed by end users, e.g. data users and administrator.

Despite the fact CS method provides many versatile ways of evaluating and testing artifacts in a real setting, it usually does not provide a method to build these artifacts.

In other words, CS approach on its own is not enough to support this research study. However, it will be used as a method for evaluating the artifact.

3.3.5 Design Science Research

A candidate that meets the criteria for the research process is a Design Science Research (DSR) methodology. Recently, Design Science has received an increased attention in computing and Information Systems research. DSR is an approach for generating knowledge – descriptive and prescriptive to solve problems that stem from both literature and practice, and are evaluated through collaborating of academia and industries. Consequently, this knowledge is implemented in means of IS artifacts. Thereby, artifacts can be either (or combination of) constructs, models, methods, or instantiations (Hevner et al. 2004). Views and recommendations on the DS methodology vary among papers, e.g. (Baskerville, Pries-Heje, and Venable 2009; Peffers and Tuunanen 2007). One main issue with DS, however, is that guidelines provided from the precursors are seldom ‘applied’ (Hevner et al. 2004), suggesting that existing methodology is insufficiently clear, or inadequately operationalized - still too high level of abstraction (Peffers and Tuunanen 2007). Descriptions of activities (procedures, tools, techniques) that are needed to conduct research are only briefly indicated.

In comparison to the previously discussed Action Research methodology, DSR might seem similar but in fact they are effectively different (Iivari and Venable 2009). This is notably true for a paradigmatic comparison where DSR offers a greater variability. AR can be considered as a special case of DSR although the latter is focused on building new IT artifacts unlike AR. Notably, a combination of those two approaches is possible, for example one can include AR method in DSR to evaluate his research. Both research approaches are concerned with practical relevance (Hevner et al. 2004) which constitutes a requirement for this study’s research methodology.

A summary of my findings on suitable research methodologies is portrayed in Table 3-1 The criteria which the comparison is based on are ‘research output’, ‘main activities’, ‘problem solving’ and ‘process building’. Criteria are based on taxonomy which

I found was overlapping within research methodology literature. The research output plays a critical role in choosing methodology. My approach is problem-driven, meaning that I am solving a research problem and satisfying business needs. The motivation behind choosing DSR instead of the others is that the design of IT artifacts is more focused on the artificial creation of solutions to encountered problems and industry requirements. For that reason, DSR, unlike AR, offers clearly defined research outputs in the form of such IT artifacts. At the same time, the design of processes is merely addressed in literature (Braun et al. 2005). However, DSR turns to fill this gap by providing a more coherent approach to building processes. Since the main aim of this research is to provide a DQ process for SOA, I consider DSR as the best suited research option. Finally, I deem DSR as legit methodology because it supports building and evaluation through collaboration with practitioners.

For these reasons, I consider DSR to be the most suitable approach among others previously observed in this chapter. Table 3-1 summarizes my argumentations and supports my choice. In the following section, I further elaborate with more detailed description about the accommodated DS approach.

3.4 Design Science as Research Methodology

The aim of this section is to portray the successful usage of the DSR methodology introduced previously, and apply it in a manner that would aid developing a method for analysing and measuring data quality in SOA. Additionally, I aim to clarify the adaptation of the DS approach by specifying the techniques used to find, solve and evaluate the research problem described in Chapter 1 and 2. The clarification made below is based on the process-oriented reference model described by L. Ostrowski and Helfert (2012). It was introduced as a part of the DS methodology for researchers focussing on process construction. The scheme illustrated in Figure 3-2 presents the Design Science process followed in this research project. Furthermore, it demonstrates relevant techniques and methods selected to accompany the DS process. Likewise, an overview of the research outputs at each step is presented in the diagram.

Table 3-1: Summary of Research Methodologies

Criteria Research Approach	Main Activities (Phases)	Research Output	Problem Solving	Process Building
Action Research	Diagnosing Action Planning Action Taking Evaluating	Specific Organizational solution	yes	no
Ground Theory	Theory Generating Theory Evaluation	Abstract Knowledge Theories	no	no
Method Engineering	Selection of Method fragments Method assembly Method performance	Specific Methods Method fragments	no	yes
Case Study	Environment analysis Observation Evaluation	Phenomenon investigation Generalization Tests	no	no
Design Science	Analysis Design Evaluation Communication	Construct Models Meth-ods(processes) Instantiations	yes	yes

The first step represents “problem identification and motivation” (1) followed by “defined objectives” (2) activities. Next is the phase of designing and developing the artifact (3). This step consists of two sub-steps: Meta Design and Concept Evaluation. Then, to evaluate the artifact in practice, “evaluation stage” (5) is followed. The last part of the process is the “communication” step, which links all previous stages through a set of publications (5). Subsequent sections provide further explanation about each step involved into DS process. As it can be inferred, DS method offers great flexibility by allowing the use of various traditional methods such as case study and survey research

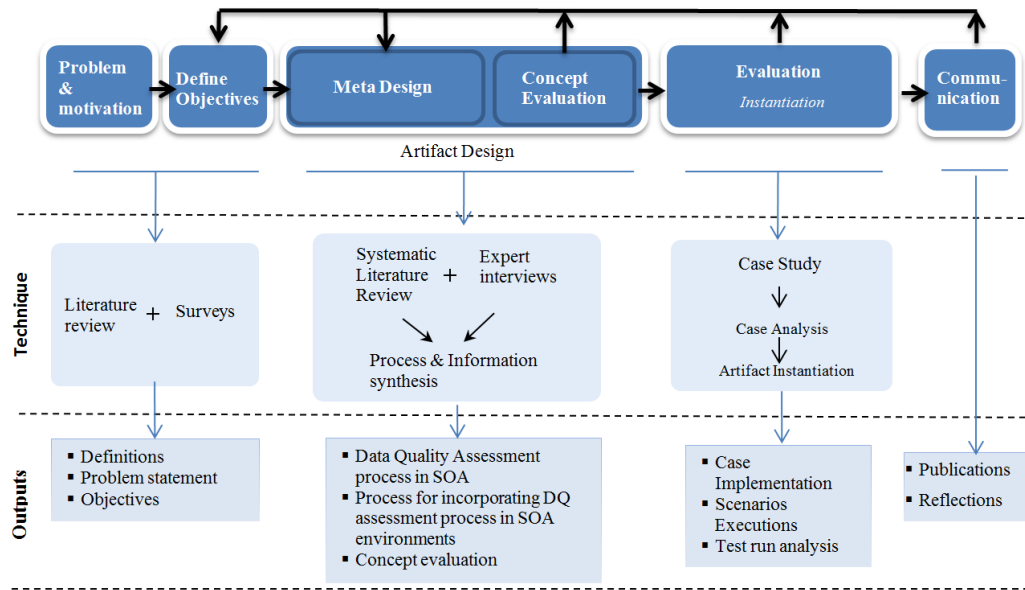


Figure 3-2: Design Science Process in DQ research

3.4.1 Problem and Objective Identification

The problem statement and motivation are identified during this stage. This may be seen as preparation, gathering knowledge, or building a foundation upon which the artifact is being constructed. The latter is accomplished by obtaining information mainly from the literature. Systematic Literature Review (SLR) (Okoli and Schabram 2010) was chosen in order to obtain more context specific results.

The ultimate goal of the literature review is to identify topics from high-quality sources. Each topic should be accompanied with a meaningful description and rationale of selection. It is a good routine to keep the search for materials transparent and replicable as much as possible (Ł. Ostrowski 2012). The goal of this phase is to extract all the relevant concepts that define Data Quality and SOA, which are believed to be its enabler factors. Moreover, it aims to identify possible approaches to detect and analyse DQ. On the other hand, knowing if the information can be trusted (e.g. of high quality) can be difficult. As a consequence literature has to meet certain criteria e.g. “Which journal/conference was the work published/presented in? What year was it published? How relevant is it for the study?”, etc. then based on the answers all literature sources received a score.

In its core, literature review aims to define and answer the research question defined during the scoping phase. SLR is divided into two activities - broad and advanced literature search. For initial scoping the research problem and motivation a broad literature study was performed. It concentrates on finding a wide variety of research relevant materials. The main focus was to gather paper abstracts, conclusions, prefaces, and references to the found materials in order to collect as many potential leads to relevant information as possible. The rigor of the systematic search process is one factor that distinguishes this approach from others. It is iterative and benefits from identification of existing systematic reviews, assessing the volume of potentially relevant materials, and using various combinations of search terms derived from the initial scope.

In this research context, the target of the broad search was to gather as much possible information about problems related with DQ. Then, we referred these findings to the domain of SOA. The types of literature inspected were consisting mainly of periodic issues, reports, journals as well as professional blogs and forums over the Internet. Main keywords that were used to perform searching were, for example, “service oriented architectures”, “data quality”, “data issues in service oriented architectures”, etc. At that point, articles from most common issues were selected as contacting potential knowledge to support the solution. The output of this step was to have a complete list of the literature that is considered for the review. In this case, the process was repeated. First, I reduced the total amount of articles to 39 by reading the abstracts of all the papers I had previously collected. Then, I have carefully read all the papers. In particular, I have verified their consistency with the research questions that were defined in section 2.7.

In addition to the findings in the literature, an independent survey has been conducted on the data quality issues in service oriented architectures, using the questionnaire in Appendix A. Targets of this survey were practitioners developing solutions in the SOA domain. 34 people were surveyed of which 6 projects coordinators, 10 system architects and 18 software developers. Questions in the survey were determined based on identified areas in the conducted literature review. “SOA, architectural

styles and technologies, data challenges and issues” Results confirmed the problem identified in the literature. Specifically, they reported that nearly 75% of surveyed companies underestimate the problem of the poor semantic data.

Based on the results from the literature and conducted survey, it was clear that in complex service environments data quality problems are persistent and information flow can be difficult to handle without having awareness of the data. Hence, it was set as the research objective to develop a process that will assess data quality in the SOA context.

3.4.2 Design as an Artifact

As it was discussed in Chapter 1, the objective of this research is to investigate how to assess and analyse data quality within SOA. In order to reach given goal, DQ assessment process is designed. More specifically, this process is composed of a few sub-processes processes that take into consideration SOA principles, data quality principles, and research objectives. The final output is a process for assessing DQ in SOA which is, ultimately, implemented as a software tool. Further aspects about the content of the artifact are discussed in Chapter 4, Chapter 5 and Chapter 6 of this thesis, as well as in Petkov P., Helfert M., and Pham, M (2012); Petkov P. and Helfert M. (2013a); and Petkov P., Rodríguez-Valenzuela, S., and Helfert (2013) seminal works.

To build the process in the Meta-phase, one of the techniques that are used is advanced literature extraction. The advanced search focuses on analysing and assessing the actual relevance of the found materials. Main and secondary objectives along with exclusion from the initial scope criteria are intended to identify those materials and to provide direct evidence about the solution for the domain. In order to reduce the likelihood of bias, these criteria may be refined during the search process. Moreover, the quality of those materials is assessed. The aims are to weight the importance of individual information; to lead the interpretation of findings and determine the strengths of results; to provide recommendations for further research. As a result of performing the advanced literature review, specific domain knowledge is extracted.

Particularly, this knowledge is in form of principles such as DQ and SOA principles (C Batini and Scannapieco 2006), processes (e.g. different data assessment methods) and information models (e.g. various XML models for representation data, etc.)

Following the DS model on Figure 3-2, professional feedback is gathered in forms of individual interviews. Specifically, each interview was as carried out as a short discussion. The duration of the discussions were approximately 40 minutes of which 20 minutes were devoted to familiarizing the expert bodies with the topic. As a result, five experts' opinions were gathered. This provided the study with valuable information and suggestions on how to build and improve the designed DQ artifact. Knowledge obtained from discussions is then structured in the form of notes and tables in different subtopics. Continuous gathering of information from practitioners is particularly valuable as it will expectantly improve the quality of the artifact.

The actual designing and developing of the artifact is performed during the “information/process synthesis” stage. This is the phase where obtained knowledge from the systematic literature review and discussions with practitioners in the form of methodologies, principles, and concepts (e.g DQM (Oracle 2008)) is fused. To construct the artifact Business Process Modelling Notation (BPMN) was employed. The result is presented in Chapter 4.5 and Petkov, P., Rodriguez-Valenzuela, S., Helfert (2013). BPMN provided this study with a notation that caters for the design and development of processes that can be interpreted by business analysts and technical developers. Section 3.4.2.1 presents the BPMN objects that are used for the purposes of this study. The construction of the artifact also employed Method Engineering (ME) approach. As stated in Section 3.3.3, ME leverages designing new solutions (processes) by combining already existing ones.

3.4.2.1 Business Process Modelling Notation as a Method Constructor

The development of research artifact is a core element of the method engineering approach. It is essential to model in a consistent manner the elements that make up the Data Quality process. The necessity for all stakeholders to fully understand the map-

ping of the process to the system/case is critical for successful implementation. These stakeholders include system architects, developers, data analysts as well as business and non-technical users. Therefore, any modelling notation employed must be widely understood by each of the communities. It is also important that the modelling of the processes allows for a high degree of flexibility (e.g. easy to be adjusted to given situation). This is an important requirement since by nature, DQ method is iterative and SOA technologies and environment are very dynamic.

There are a number of well-established modelling techniques that can be employed to model processes. Examples include Use Case Diagram, Activity Diagrams in UML and Business Process Modelling Notation (BPMN). The Business Process Management Initiative (BPMI) has developed a standard Business Process Modelling Notation (White 2004). The reason for the development of BPMN was to create a bridge for the gap between the business process design and the process implementation. BPMN is based on a combination of flowcharting techniques and graphical models for business operations. An examination of the BPMN core concepts suggests that it is intuitive and easy to understand. One of the advantages according to the BPMI is the simplicity of its mechanisms for creating business process models while also capturing the complexity of business processes. For the purposes of this research BPMN language elements are categorised into three groups; 1) flow objects, 2) connecting objects and 3) artifacts.

Flow objects. BPMN classifies three elements which are flow objects. Graphical shapes representing such objects are given in Figure 3-3.

- **Event:** describes something that occurs or happens. They normally modify the flow of process and are represented by a circle. Three types of event are available – start, intermediate, and end.
- **Task:** a generic term for work that is performed. Tasks can be atomic or compound. A task can have sub-tasks. BPMN supports also looping tasks.
- **Gateway:** Represented by a diamond shape, determines decisions, forking, merging and joining of paths. Gateways could be generic, event, and join.

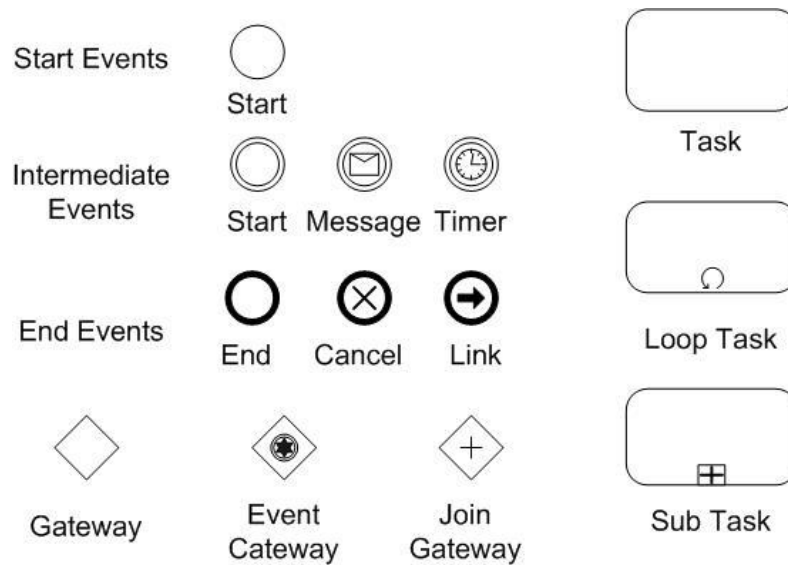


Figure 3-3: BPMN Flow Objects

Connecting Objects. The flow objects are connected to each other by connecting objects outlining the basic structure of the business processes. There are three connecting objects that provide this function. Connecting object shapes are presented in Figure 3-4.

- Sequence Flow: Used to show the order or sequence that activities are performed in. Sequence flow always has direction.
- Message Flow: Displays the flow of messages between two process participants. Similar to sequence, message flow also has direction.
- Association: Used to associate data, text and artifacts with flow objects.

Artifacts. BPMN allows artifacts to be added to as appropriate for the context of the business processes being modelled. In this research, such entities were used for describing the process model. Different types of artifacts objects are depicted in Figure 3-4.

- Data Object: A shape to show how data is required or produced by activities.
- Group: A group is represented by a rounded corner rectangle. This can be used for documentation or analysis purposes.
- Annotation: Instrument to provide additional text information.

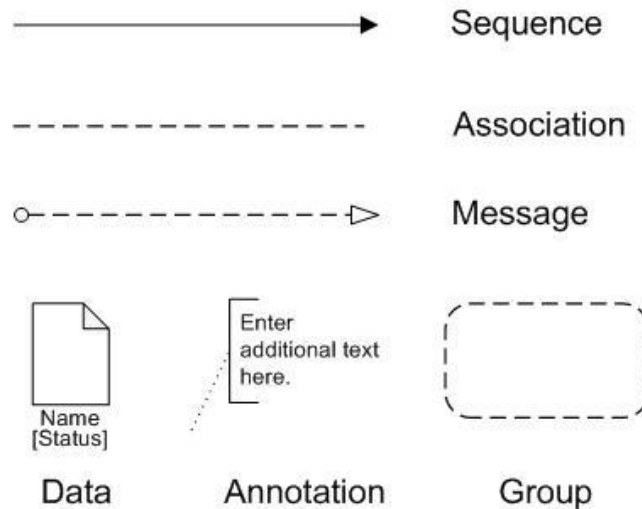


Figure 3-4: BPMN Connecting and Artifact Objects

3.4.3 Evaluation of the Artifact

As mentioned in Section 3.2, an important requirement of the designed artifact is that it needs to be practically evaluated in real world settings. On the other side, one of the advantages of the DS methodology is that it can encompass various traditional methods, e.g. case studies, experiments, Action research. Considering the requirement and the discussion in section 3.3, Case Study method has been accommodated to evaluate and demonstrate the utility of the research artifact. A case study is an empirical inquiry that investigates a phenomenon within its real-life context (Carlsson and Henningsson 2011). To evaluate the DQ process (e.g. software component) in an operational manner, participant's feedback from different scenario executions is collected.

The "design in practice" is important step validation of the artifact. Avoiding this step might question the reliability and utility of the designed artifact within real settings. As stated in Section 3.2, an important requirement for the outcome of the study is that it must be pragmatically evaluated. DS considers collaboration with practitioners as vital part of the research progress. However, the evaluation of this process might require selecting the right case and collaborator (e.g. what case of SOA?)

For this study, the researcher has chosen an academic partner, in the name of University of Granada, who is embarking on a project called Dynamic Open Home Automation (DOHA). DOHA is a SOA-based platform for the access, control, and management of home automated systems, composed of a set of lightweight and independent services (devices). The motivation behind choosing academic collaborator before any other particular company is based on the fact that: first the research is practically oriented rather than theoretical. Secondly, the project is independent and conducted by non-profitable organisation, which in turn, reduces the risk of bias. The third reason why this partner was chosen is that DOHA project incorporates all service-oriented principles (Erl 2005) to deliver a home automation middleware. Last but not at least, DOHA middleware lacks of mechanism to detect and assess mishandle poor data which makes it very good candidate for incorporating a data quality process.

Ultimately, the utility of the artifact is evaluated by implementing a software add-on, a data quality service, to existing middleware which is then tested in a real environment. However, the implementation of that kind of module required additional information about DOHA system; for example: “Which key principles of SOA are incorporated in DOHA middleware?” “What data formats are used to represent information?” “What standards and protocols are employed for service communication?” “What design pattern DOHA middleware implements to orchestrate services?” To answer all these questions, additional information is required. This information is collected through continuously bi-weekly architect and developer’s discussions that aimed to answer the questions listed above. In addition, to gain more insight of the DOHA architecture, documents and code observations may be necessary. Fortunately for this study, DOHA middleware is open-source platform and accessing the code and documentation would not present any difficulties. This was yet another reason why this particular collaborator and SOA case was selected. Along with the process of data collection, the partner is also asked to provide feedback and proposals of how to make the process more clear and efficient for future adoption. Thereby, the aim is to incorporate the DQ mechanism in for DQ software tool based on collected additional information about DOHA system.

3.4.3.1 Collecting Evidences

An important choice that is recognized as being fundamental for any research study is between qualitative and quantitative approaches.

On one side, quantitative research methods were originally developed in the natural sciences to study natural phenomena. As the name suggests, such studies involve the investigation of phenomena through numbers. This can subsequently entail the usage of mathematical models, statistical ones, or computational techniques (Given, 2008). Examples of quantitative methods that are well accepted in the social sciences include survey methods, laboratory experiments, formal methods (e.g. econometrics) and numerical methods such as mathematical modelling (Myers 1997).

On the other hand, qualitative studies were developed in the social sciences to enable researchers to study social and cultural phenomena. Examples of qualitative methodologies are action research, case study research and ethnography (Myers 1997). Qualitative data sources include observation and participant observation (fieldwork), interviews and questionnaires, documents and texts, and the researcher's impressions and reactions.

There is also a third approach of collecting information e.g. "mixed methods" data collection. This approach normally involves a combination of the two methods mentioned above e.g. *quantitative* and *qualitative*. Mixed methods research can be viewed as an approach which draws upon the strengths and perspectives of each method, recognising the existence and importance of the physical, natural world as well as the importance of reality and influence of human experience (Johnson and Onuegbuzie, 2004). For instance, if the study aims to find out the answer to an inquiry through numerical evidence, then the literature suggests use of the quantitative research. However, if the aim of the study is to explain why particular event occurred or why particular phenomenon is influential, then qualitative research is suggested.

A variety of reasons led me to decide to opt for the qualitative approach. In the first place, Qualitative research methods are being used increasingly in evaluation studies, including evaluations of computer systems and information technology. Quantitative

methods are excellent for studying evaluation questions such as user satisfaction, completeness, error rates, etc., in which selected features of the information technology, the organization, the user, and the information needs generally are treated as independent, objective, and discrete entities, and as unchanging over the course of the study (Scacchi 1982). Secondly, the outcome of quantitative evaluation is to understand how the user perceives the system under investigation. Since Users' perspectives are not known in advance, it is difficult to determine or understand these through quantitative approaches only. By investigating users' perspectives in depth, qualitative methods can contribute to the explanation of users' behavior with respect to the system, and thus to the system's successes and failures and even of what is considered a "success" or "failure" (Kaplan and Shaw 2004). Thirdly, it allows for increasing the utilization of the artifact under investigation. Often administrators, systems architects, and practitioners find purely quantitative studies insufficient because such studies do not seem related to their own understanding of the situation and the problems they are encountering. By providing evaluation findings that connect more directly with these individuals' perspectives, qualitative methods can increase the credibility and usefulness of evaluations for such decision makers (Patton 2015). Finally, qualitative evaluation provides a basis for improving the artifact under development, rather than assessing an existing one. The outcome of this research is a novel artifact that is possible to undergo improvement in future. Using qualitative methods can help in identifying potential problems as they are forming, thereby providing opportunities to improve the system as it develops. These evaluations also allow for varying and changing project definitions and how the system and organization are mutually transformative, thereby enabling learning by monitoring the many experiments that naturally occur spontaneously as part of the processes of implementation and use (Kaplan and Duchon 1988).

In general, the evaluation purpose of this study is to answer how "useful" DQ process/tool is, as well as to explain the reasoning behind the outcome e.g. "Why is DQ process/tool useful?" To achieve this I have chosen not only quantitative approach e.g. System Usability Scale (SUS) survey tool but also the qualitative one thoroughly described in previous paragraphs. Mixing both methods is extremely useful for vali-

dating findings since it provides a way to use qualitative data to augment or confirm qualitative outcomes. Normally this process would involve collecting both types of data (qualitative and quantitative) at roughly the same time; assessing information using parallel constructs for both types of data; separately analyzing both types of data; and comparing results. Results are presented in chapter 6 of this study.

3.4.4 Communications of Research

Dissemination of the results of the evaluation requires adequate resources, such as people, time, and money. Developing academic papers and preparing presentations may be difficult for community members who have other commitments (Parker et al., 2005). Hence the results of the research must be presented in a manner allowing for their technical and management application. This requires sufficient detail that allows for the technical implementation of the artifact and also, from an organizational perspective, a description to cater for its implementation. This study aims to provide a detailed method design and description both in technical and organizational terms. My work has been presented to the IS community (P Petkov and Helfert 2013b; P Petkov, Ostrowski, and Helfert 2013; P. Petkov and Rodríguez-Valenzuela, S., Helfert 2013; Plamen Petkov, Helfert, 2012)

3.5 Summary

The purpose of this chapter was to describe how this research study was conducted. It presented an overview of a number of research methodologies that were identified as possibly suitable, considering research context. This chapter also explained the Design Science research philosophy and argued why it was chosen as a main research approach. Thereby, an illustration of adapted research framework as well as introduction of research process in the form of an IT artifact ‘build’ cycle was presented. Moreover, an explanation of how DSR fits in a greater research context by examining various ways of collaboration between industry practitioners and academic researchers was given. Furthermore, it is relevant to know how to shape the research effort. This chapter partially answered to several research questions. Firstly, it gave an insight of how to design and implement data quality assessment method (RQ 2.3 and

RQ 2.4). Design science was widely coined in the literature as an approach for developing various IT artifact such as IT processes (Hevner et al. 2004; Venable, Pries-Heje, and Baskerville 2012; Gill and Hevner 2013). Data quality assessment is IT process that requires incorporation of particular design methods in order to meet research objectives, e.g. to provide a comprehensive, transparent and practically oriented data quality execution. To achieve transparency and comprehensiveness, Business Process Modelling Notation was suggested as a method constructor. The decision was based on the fact that BPMN offers well-established process modelling elements and combination of flowcharting techniques that are well interpreted by both business and technical people. To achieve practicality the DSR strongly supports the collaboration with practitioners. DSR suggested that developing process should be cyclic and rigorous with constant input from industry partners as well as academia. It also suggests that developed artifact must be relevant, e.g. implemented within real world case settings. Consequently, this leads to the partial answer to the third research question RQ 3 - How to evaluate the utility of the developed Data Quality assessment method? This section does not provide a full answer but it elaborates by providing discussion of evaluation strategy (section 3.4.3) as well as methods for data collection (section 3.4.3.1) Following chapters 4,5 and 6 compliments answering RQ2 and RQ3.

Chapter 4 Assessing and Analysing Data Quality in Service Oriented Architectures

4.1 Introduction

In the previous chapters, the definitions of SOA and DQ were presented along with raising the issue about the lack of DQ methodology currently used in SOA. Particularly the need for developing a method to detect semantic data quality issues in SOA was highlighted as a key concern. In this chapter, a description of the DQ framework will be given. This framework is built on methods and guidelines. Also, the actual DQ method, which suggests a careful examination of the data subjects and their contexts, will be discussed. Data subject is any data variable under data quality scrutiny. Furthermore, methods for building and executing data quality statements along with the guidelines for incorporating the DQ process are presented. This chapter's main objective is to answer the following research questions:

- What is the DQ approach employed? What particular methods and techniques are used to achieve the research objective?
- How is DQ methodology incorporated into SOA?
- Who is going to benefit from this research - roles and actors in SOA?

It is crucial to recognize different groups of people - their roles and responsibilities in each cluster of SOA ecosystem since ultimately they are the ones that benefit from this research.

4.2 SOA Ecosystem and Data Quality

To put Data Quality into context, Figure 4-1 is taken into consideration. It portrays two perspectives of the service oriented architectures - the technology and the application perspectives.

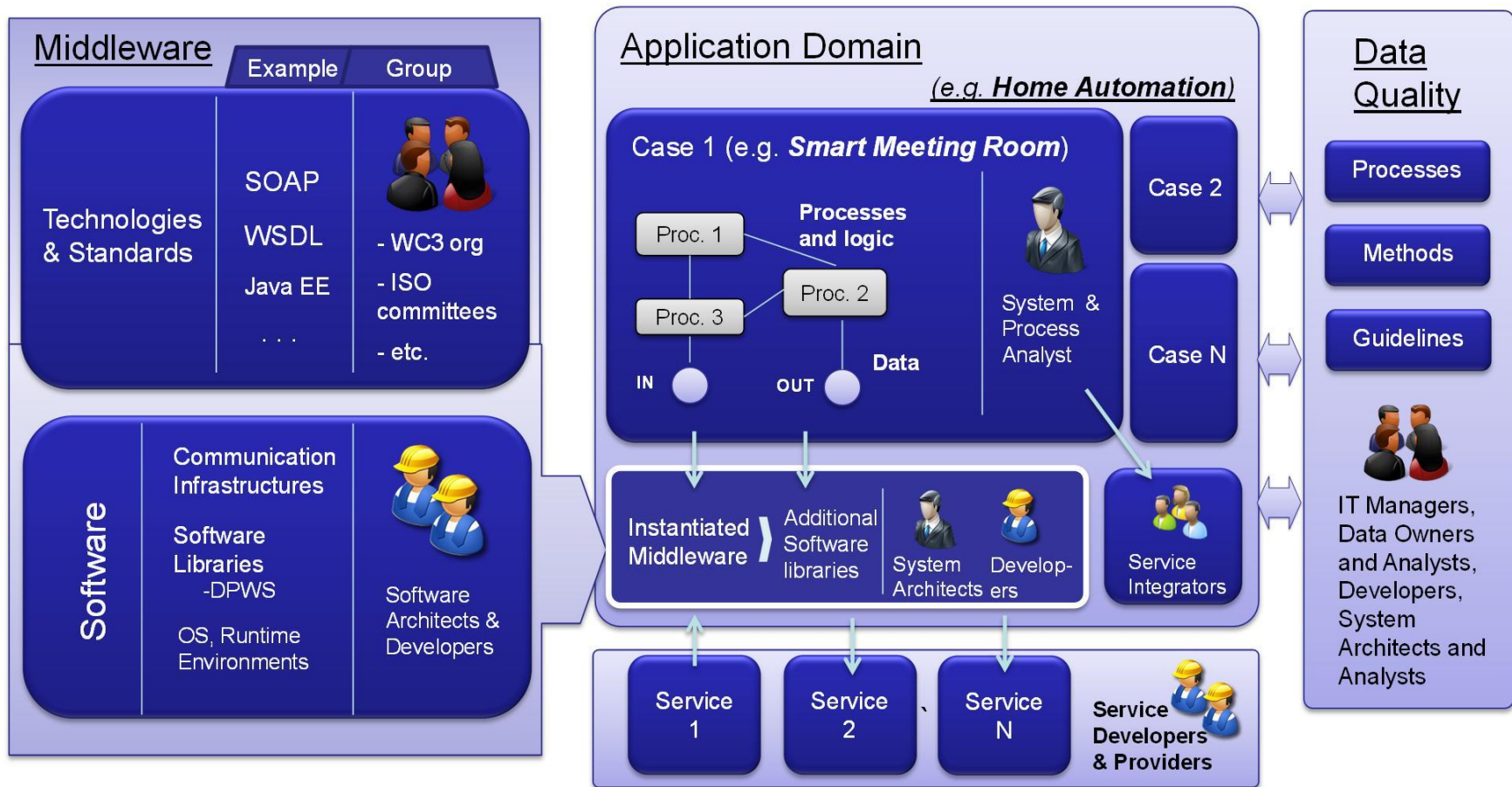


Figure 4-1: SOA Ecosystem and Data Quality. Roles and Examples

The figure above also incorporates the data quality perspective. Additionally, it outlines entities and actors involved in the SOA and DQ process. Ultimately, the image represents an overview of the research context and output which are further elaborated in following few paragraphs.

Service-oriented paradigm leverages the development of new functionalities and solutions by combining already existing ones. It provides the guidelines and principles that promote all the advantages of services' based applications - such as greater flexibility levered by loose service coupling, increased return of investment in terms of service reuse, the shorter time to deliver solutions, etc. Never the less, there is a critical need for standardizing the way SOA is built, executed, and governed. To do so, SOA communities and organizations, such as World Wide Web Consortium (W3C), Advancing Open Standards for Information Society (OASIS) and International Standard Organization, are constantly debating and consolidating for greater homogenization of standards. As noted earlier in this thesis, service oriented approach relies heavily on following strict specifications and rules. Examples of standards and principles that have been widely approved and accepted by practitioners (at the time writing this) were discussed in Sections 2.2.2 and 2.2.3. The increasing popularity of the SOA approach and the benefits it delivers, along with the reconciliation of standards by offered by international consortiums and organizations has motivated a lot of industries to deliver practical solutions for SOA. Such solutions are SOA middleware - a special type of software that aids the interconnection of services, deferent software packages and libraries.

The other perspective taken into consideration is the context of the SOA application. Service compositions can be extremely diverse and solutions can differ from one another to meet specific needs. This is a somewhat logical outcome since business (or application) requirements in each case are unique. For example, if the requirement is better communication security between services, SOA architect would employ a SOAP-based middleware solution. Or if the requirement is greater expressiveness of the message envelope (e.g. different message format), he might choose REST based solution. This is just a simple example of how re-

quirements affect decision making and diversity in SOA. There are many other factors involved when building SOA solution and these factors ultimately depend on the case requirements. Many times middleware does not meet some of the requirements in its standard form. Therefore it needs to be adjusted by adding new functionalities. For example, such new functionality might be a process that monitors the quality of data running through the environment. The design, development, and incorporation of new features must be carefully executed and coordinated by SOA architect and middleware developers after all requirements are thoroughly discussed and described by IT managers.

The third standpoint is about how data quality relates to the SOA. The data quality framework comprises of processes and suggestions that have to be considered in cases where detecting poor data is an important objective. More specifically, the framework proposes methods, step-by-step guidelines, as well as different factors and points that have to be taken into account while embarking on a DQ project. As seen in Figure 4-1 these methods apply to a particular SOA context. To make the process more generalized, however, the DQ framework considers both the technological and application perspectives of SOA. A Data Quality initiative requires all aspects of a given case to be examined therefore needing adequate cooperation between different IT groups, from managers, through system and data analyst, to developers. Subsequent sections describe some fundamental concepts related to assessing semantic data quality.

4.3 Motivational Case – Dynamic Open Home Automation (DOHA) Smart Meeting Room

The researcher of this study in conjunction with developers and experts from University of Granada (UGr) had developed a real world system to meet the needs of BIG group. BIG is a research group situated in Dublin City University that carries out weekly meetings in one of its team collaborating rooms. The room is also used by other research groups within DCU. To avoid the hassle of manual preparation of computers and desired comfort every time before meeting, they requested a solution that will activate all necessary settings automatically. Such activities include setting the computers, adjusting the room temperature, control of the lu-

minance, starting monitors projector and computers, loading presentation, etc. Figure 4-2 depicts some of the options the host can choose between before and after a meeting.

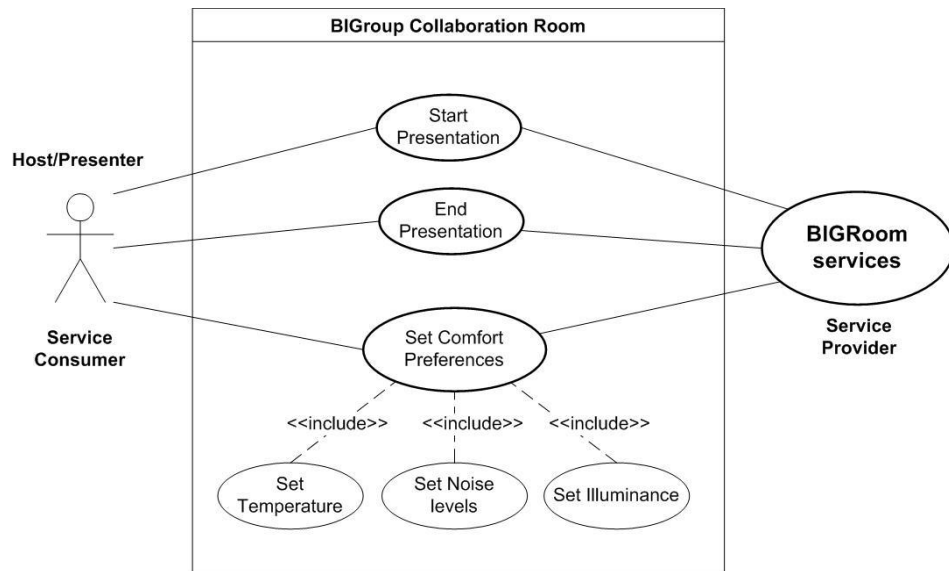


Figure 4-2: Collaboration Room Case Scenarios

One should note that each device/process in the smart meeting room is seen as a service e.g. a black box which delivers certain functionality(ies). This case scenario is based on the SOA paradigm and follows principles listed in section 2.2.2 for service transparency and autonomy. The communication infrastructure that services are utilising to talk with each other is provided by UGr and it is based on a SOA middleware project called Dynamic Open Home Automation (DOHA). DOHA is a platform for the access, control and management of home-automated systems which are composed of a set of lightweight and independent services accommodating SOA design patterns. Further discussion about the structure of DOHA middleware, standards and protocols will be given in Chapter 5.

Figure 4-3 represent simple composition map of deployed services. The main service is the *Presentation handling service*. It is responsible for processing users' preferences and controlling others services such as *temperature* and *ambient light control services*. The *ambient light control service* involves the *ambient lights sensor service* and *windows blinds service*. Round shapes in the figure represent logical services whereas square describes a device.

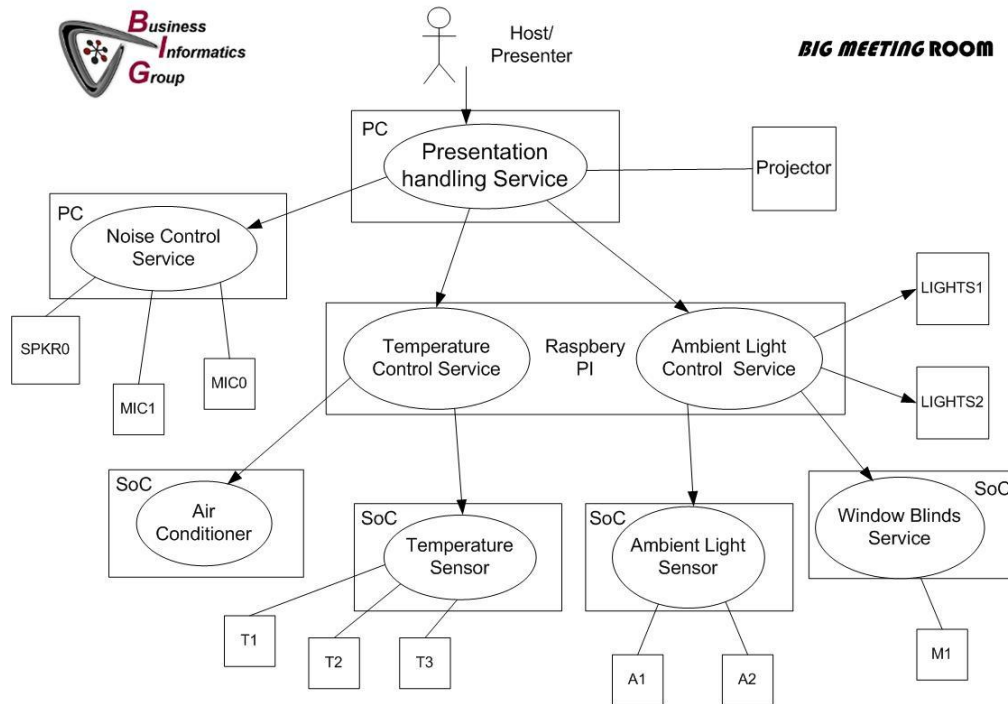


Figure 4-3: Smart Room Service Composition

As it could be observed behind every logical service stands a device. Devices can host more than one service such as the case with the Raspberry Pi mini pc – it runs *Temperature Control* and *Ambient Light Control* services. The number of services hosted on a device depends on the architecture design and the physical performance capabilities of the device. Most of the devices in ambient spaces have limited processing power and can host only one service such as the *ambient light sensor service*. It is running on a System on a Chip (SoC) device that can handle input from two sensors. The picture above also involves devices that do not host any logical services e.g. T1, T2, A1, MIC1, etc. These devices represent pure physical services, i.e. sensors and actuators. Figure 4-4 outlines part of the hardware configuration used for the case. Related to the diagram in Figure 4-3, these are represented as rectangular boxes. Further description about hardware will be omitted since the idea is to observe system from a logical point of view - as a composition of services (black boxes with inputs and outputs), and not from hardware point of view. It is not of users' concern what physical device lies behind each service.

An import characteristic about this case is that system presented above does not exist in isolation. Certain external factors such as “time of the day”, “outdoor temperature”, “room physical properties”, etc. are bound to influence the environment. As it will be seen in next sections correct identification of these factors and their dependencies are crucial when assessing DQ.

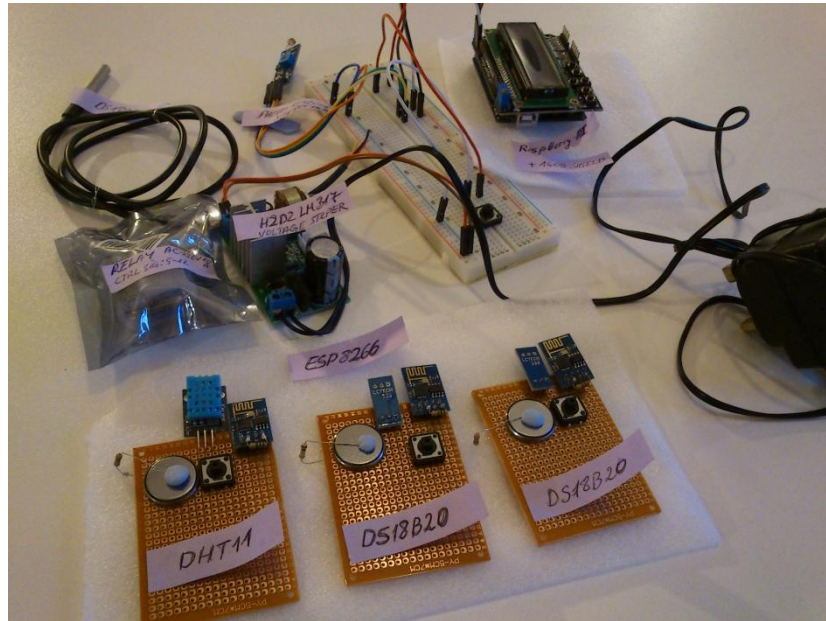


Figure 4-4: Part of Hardware Settings – Sensor and Actuators Devices

4.4 Contextual Data in SOA

The meaning of the word ‘context’ is ambiguous; literature provides countless definitions that try to describe its notation. Generally, ‘context’ is defined and used within a particular area of study. However, most of the definitions suggest that it expresses additional information to explain and understand a thing – something that is written or spoken. Within the data quality area, context is commonly used to specify a scope or a boundary of a study area or a discourse (YW Lee 2003). Context is a differentiator and relationship builder that stipulates the bond between contents and their environments. It connects and shapes contents and the environment, which in turn structures and influences the activities via the context. Context is the differentiating and connecting entity that inherits the imprints for further processes and activities (YW Lee 2003). Contexts are factors pertaining to

information in use, hence information quality is often defined as “fitness for use” (Strong, Lee, and Wang 1997).

In practice, contexts in data quality have been implicit. Nevertheless, they have been a decisive part of the resolving data quality problems. Naturally, exploring context is a critical direction for practitioners in creating their solutions. Madnick et al. stated that once the contexts of data are established, recorded and stable, solutions, such as context interchange technology (Goh et al. 1999), can be incorporated for reconciling differences in data and systems integration between sending and receiving entities (SE Madnick 1995). This solution approach is predominantly sufficient for a diverse heterogeneous system that requires data aggregation. However, in today’s dynamic service oriented environments it is very likely for contexts to be unstable or not clearly defined. This arise challenges related to the definition of the data and the clarification of its meaning over time.

Bertossi et al. 2008 accentuate that data quality is highly contextually dependent. They formalize ‘context’ as “...a system of integrated data and metadata of which the data source under quality assessment is a particular and special component” (Bertossi, Rizzolo, and Jiang 2008). Furthermore, they elaborate that the context for quality assessment of data in a certain instance D of schema S is given by an instance I of a possibly different schema C, which could be an extension of S. In order to assess the quality of D, it has to be “put in context”, which is achieved by mapping D (and S) onto the contextual schema C and data I. Actually, C can be more complex than a single schema or instance, namely a collection of database schemas and instances interrelated by data - and schema mappings. See Figure 4-5.

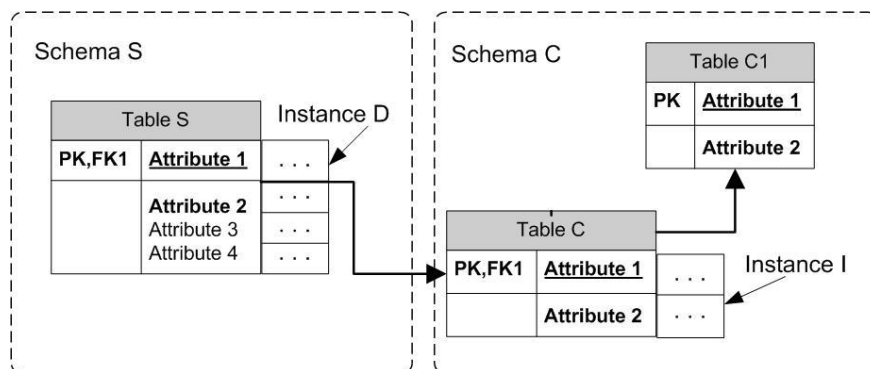


Figure 4-5: Contextual data in Relational Database Systems

Related to the work presented above, in this research, an alternative view of a term contextual data is presented. Specifically, contextual data is any data provided by services that are external and independent of the observed service environment. This data is often related/or is a function of the external services. The relation between external and internal services is presented via service mapping schema. The illustration below (Figure 4-6) visually defines the meaning of contextual data in this study.

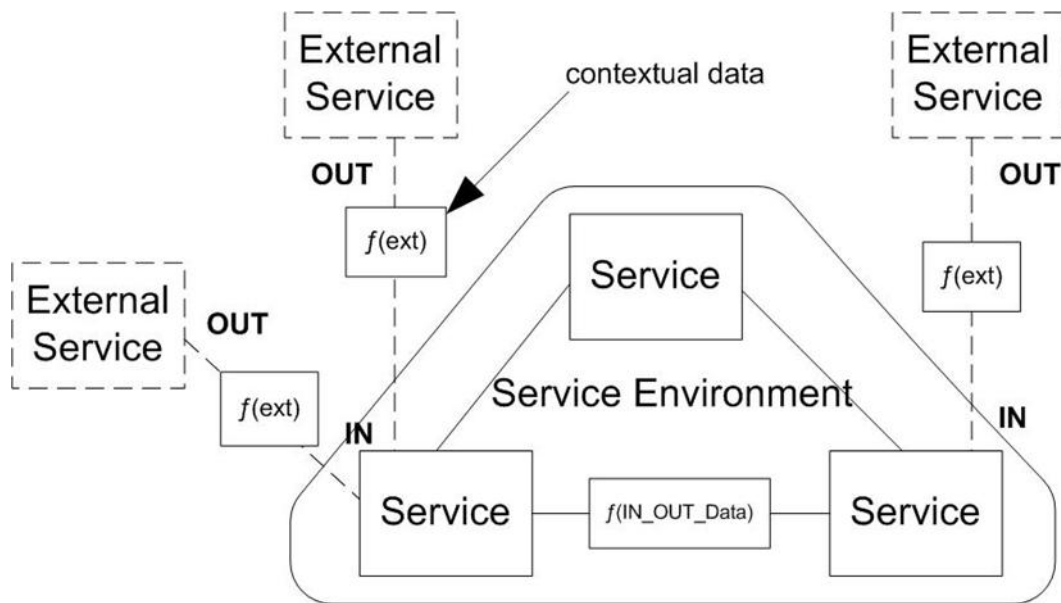


Figure 4-6: Contextual Data and Service Oriented Environment

It presents a closed service environment, consisting of three (in this case) services and an open – external system comprising three other (external) services. The latter are related to the internal composition through a specific shared parameters (variables) – normally processed by functions – $f(IN_OUT_Data)$. To further elaborate, take into account the Smart Room case introduced in section 4.3 –Figure 4-3: There are two services (sensors) that are measuring the amount of light – inside (sensor A1), and outside (sensor A2) of the room. Bearing in mind the concept explained above, the shared variable is luminance (physical concept defining the intensity of light bouncing off area of surface). The luminance inside and outside the room are measured independently; however there is dependency function that defines how much indoor luminance will increase/decrease based on the lumi-

nance outside, the number and size of windows in the room, the state of the lights in the room and the position of the shutters. An example of a relationship between variables and contexts is depicted on Figure 4-7. The rounded square defines the scope of the context, e.g. variables affecting measured/observed “indoor illuminance” variable. It is important to note that context can have different values, i.e. it can be instantiated. As mentioned before, contextual data is crucial for assessing DQ, hence, quality of the context must be also taken into consideration. In order to ensure the quality of the context, all dependencies must be thoroughly analysed. To ensure the contextual data is properly measured and discrepancies between virtual and real world are kept to their minimum, it is suggested that data is collected from at least three independent sources. For example, the “weather condition service” could obtain data about the “solar UV index” from three different separate weather services. Then, the three results could be compared against each another and the one that deviates the most from the other two could be discarded. Normally, the deviation margin value is set by the DQ administrator. Alternatively, another approach could be applied by implementing an algorithm for calculation the average value obtained from the three sources. Additionally, particular attention must be paid to the external factors e.g. the “time of the day” which is dependent on the locale where the smart room is based.

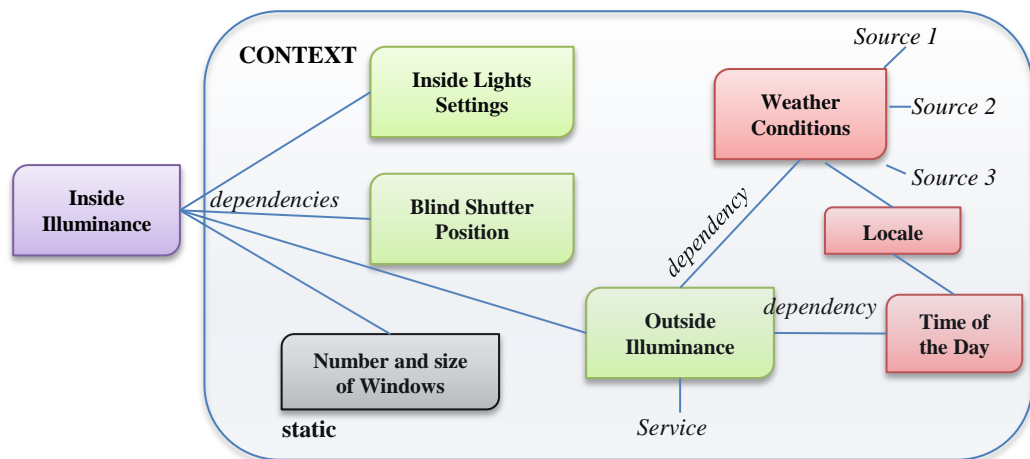


Figure 4-7: Relationship between Data Variables

One should note that due to dynamic services’ nature, the generated contextual data can have different values, e.g. instances. This implies that contextual data can vary over time suggesting instability and uncertainty of the monitored internal

data. Taking the case above - if luminance outside decreases due to weather change or time of the day, luminance inside would change respectively depending on the context functions – $f(ext)$. Thus, it is very important that changes in context are thoroughly examined and taken into account when assessing the quality of observed internal services' data.

4.4.1 Data Constraints

In traditional database science, a data constraint is a limitation which administrator places over the data that users can enter into a column or group of columns. A constraint is part of the table definition. Common types of data constraints are: primary key constraints - value(s) in specified column(s) must be unique for each row in a table; foreign key - value(s) in specified column(s) must reference an existing record in another table; not null constraints - value in a column must not be NULL (empty); etc. Normally, data constraints are referred to as integrity rules.

Data constraints in this research have slightly different definition from the traditional data constraints mentioned above which are commonly used in Data Base Management Systems DBMS and widely discussed in the majority of the Data Quality literature. In service-oriented environments, data is fetched and processed by a particular application (service) and then delivered to the service consumer through its interface. On the other hand, a fundamental principle in SOA is service abstraction, which in turn implies generalization and lack of information about the kernel process and data of the service. Service is seen as a black box, which obscures the access to the low-level database management systems (e.g. behind the service interface) by the data administrators. Rather, they have to constrain data on a service level. Figure 4-8 outlines a fragment of a service oriented environment that includes a service provider and a service consumer.

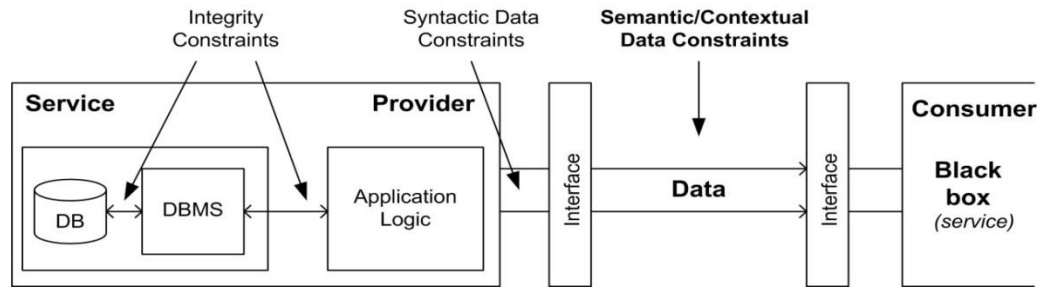


Figure 4-8: Data constraints on different levels

In the picture above both services are seen as black boxes, meaning that internal structure of the data and the processes that modify it are not available to the service data administrator. As it can be observed, “Integrity constraints” only apply on a very low level - the physical database level. Sometimes, however, the integrity constraints could be secured through the application logic. Description and examples for the latter were given in a previous paragraph. Following the integrity constraints, position and role of the syntactic data constraints are also illustrated in the figure. These types of constraints are typically handled by the service’ interfaces, which ensure that processed data conforms to the standards for representation described in the service contract. Semantic constraints are not explicitly defined in to WSDL, but there are several technologies such as OWL-semantics (Kang et al. 2009) that allow describing the service semantics (i.e. service dependencies). Despite the fact OWL-semantic standard tackles data integration issues and ensures service interoperability, it does not secure in any way that service will deliver accurate information. OWL describes the context of the data but not ensure data flow is accurate.

Semantic Data constraints in this study are defined as meaningful statements about the data. Considering the schema on Figure 4-8, data constraints may be defined as logical statements to ensure the accuracy of information running within the internal service environment. In this case, we speak about semantic data constraints. Additionally, data constraints related to external services can be also defined. This is very important since these definitions give the data a context. In this case, we refer to contextual data constraints or dependencies. A semantic data constraint, for instance, can be as the following statement about the “Luminance” data: “Luminance in the room has to be lower than 3000 lumens”. Whereas an

example of contextual data constraint (e.g. dependency) is: “Weather is sunny when measured outdoor luminance is higher than 5000 lumens (and vice versa)”. The latter gives a meaning of the word “sunny”. It is also deducted that this statement applies in day time – e.g. between 7 am and 7 pm. In a nutshell, data constraints are definitions within which a given set of data is given meaning. Data constraints are the actual entities that define the meaning of information within SOA as well as its context. Later in this study the word ‘predicate’ is referred as ‘data constraint’ or ‘quality statement’.

4.5 Total Data Quality Management Methodology TDQM

Currently, there are a number of different approaches for measuring data quality (Lee et al. 2009; English 1999; Ge 2009; Foley 2011; Scannapieco, Virgillito, and Marchetti 2004; Ballou et al. 1998; Redman 2001; Loshin 2001). One that provides a full spectrum of methods and guidelines is Wang’s Total Quality Management Methodology (TDQM). TDQM is grounded on Data Quality Management which involves the establishment and deployment of roles, responsibilities, policies, and procedures concerning the acquisition, maintenance, dissemination, and disposition of data (Geiger 2004). A partnership between data owners/users and technology groups is essential for any data quality management effort to succeed. The data consumers are responsible for establishing the quality rules that govern the data and are ultimately responsible for verifying the data quality. While the Information Technology (IT) group is responsible for establishing and managing the overall environment – architecture, technical facilities, systems, and services – that acquire, maintain, disseminate, and dispose of the electronic data assets. As for the process of assessing information, TDQM outlines four steps with respect to definition, measurement, analysis and improvement of the data. Figure 4-9 represents this process.

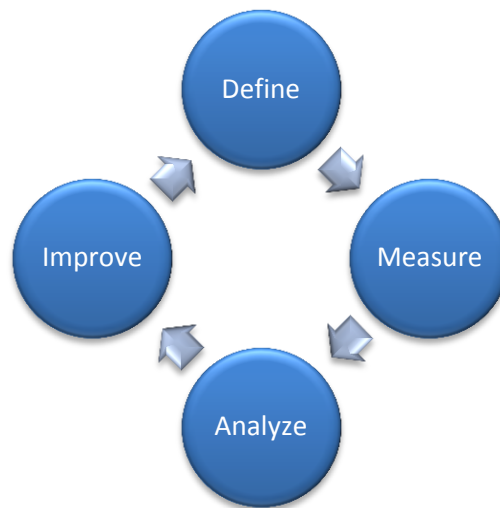


Figure 4-9: Total Data Quality Management cycle (R. Wang and Strong 1996)

Although TDQM is fairly old and is designed to assess DQ in monolithic information systems, it provides a wide range of comprehensive and systematic techniques from which this study can benefit. The adoption of the TDQM cycle is recursive in nature, where lessons are learned at each stage. The knowledge gained at each stage provides the basis and many of the inputs for the subsequent stages, thus ensuring that the dynamic of the environment is catered for throughout the life cycle. The individual method fragments provide the necessary process and product models that capture the information associated with each of the stages. An explanation of the constructs that were employed along with their detailed processes for both construction and application are outlined. TDQM is used as an overarching approach or philosophy to ensuring that each phase of the DQ lifecycle is examined. A novel method that considers each phase of the lifecycle is then proposed.

4.6 DQ Process in SOA

This section describes the actual approach for assessing semantic (contextual) data quality in service-oriented environments. Accordingly, a four-stage process containing following steps is developed:

- a) **Building information profiles stage.** The goal of this stage is to aid data consumer to understand the intended use of assessed data. As most of the

data quality assessments are context related, users firstly have to understand the purpose of assessed data.

- b) **Preparation stage.** At this step, users/data consumers define the quality of the data in a form of ‘quality’ statements.
- c) **Execution stage.** This is a programmatic process that executes data quality statements built in the previous stage.
- d) **Reporting stage.** At this stage, data quality reports are generated. These reports provide the basis for the data quality analysis and improvement. Based on data quality statements, users evaluate the extent to which information products are fit for the intended use. Since ‘quality’ is subjective, outcomes might differ from person to person. The techniques of improvement are omitted from this research.

In order to recapture my assessment approach for SOA, I position my data quality process in the context of information quality management. Consequently, I map the assessment methods and procedures to the TDQM cycle as denoted in section 4.5. The first two strategic steps a) and b) of assessing contextual data are in the ‘defining’ phase. The other two e.g. c) and d) are in the ‘measuring’ phase. Figure 4-10 presents an overview of the solution as well as outlines the relationship between TDQM cycle and the approach for assessing the quality of data. More detailed breakdown of each stage is discussed in the following sections.

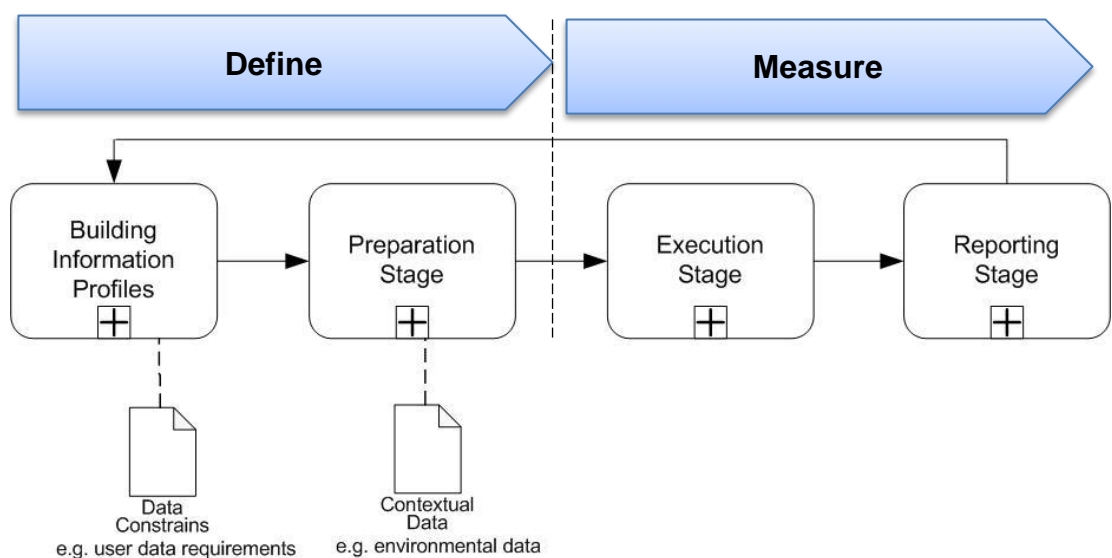


Figure 4-10: DQ assessment Process in SOA

4.6.1 Building Information Profiles Stage

Data profiling is the process of gaining an understanding of the existing data relative to the quality specifications. It is a starting point to any data quality initiative. Profiling is a basis in understanding anomalies and assessing data quality, but also used to discover, register, and assess enterprise metadata (Loshin 2001). Other benefits of the process of profiling include improving data quality, reducing the time for implementation of major projects, and most importantly - improving the understanding of the data. Understanding of the data is a vital criterion which lays the foundation for assessing data quality process. Specialists in data warehouses system typically associate data profiling with “source system analysis” (Geiger 2004). In warehouse systems, bad data might often occur because of faulty data acquisition, delivery processes, or due to interpretation problems. With source system analysis, data warehouse team aims to allocate origin of mishandled data. This is often handled through obtaining additional information about data sources such as when data was delivered, who/what made the transaction, which rules and processes were involved in governing the data in its way to the data warehouse.

The idea of “profiling” in this research is similar to the “source system analysis” in data warehouses. The main difference here, however, is that source system could be a destination as well e.g. provider service could be consumer too. In this case, profiling must be performed on each of the systems (services). The goal is to build an overview, a comprehensive map, of the systems (services) and their dependencies involved into governing particular piece of data. In other words, the aim of this stage is to build information profiles of the service composition and data used by the services.

Traditional data profiling target analysis of particular database/system in order to obtain additional information about data. To achieve the aforementioned objective, various statistical methods such as calculating mean values, standard deviation, frequency, etc. are typically adapted. This additional information about data (e.g. metadata) can be data type, length, null values, string patterns, etc. (Loshin 2009). Normally, data analysts use specifically-built software tools to ease the process of profiling (Rahm and Do 2000; Singh and Singh 2010). In contrast with traditional data profiling, in SOA, data analyst do not directly have access to ser-

vice databases. This makes using the above-mentioned techniques impractical. Also, the goal here is not to profile particular dataset but rather to describe dependencies of the data with the services that control it. Hence, a manual approach of building profiles needs to be accommodated.

A simple BPMN diagram of the profiling process in SOA is presented in Figure 4-11. The process of building information profile begins with identifying the number of the services and data subjects involved into the composition. At this stage the body in charge (normally this is system or data analyst) is getting familiar with the case. Throughout this process, he has to cooperate with SOA architect in order to obtain needed information. Additionally, during this stage, the scale of the DQ endeavour is determined. Next step of the process is to examine the service and their data relationship.

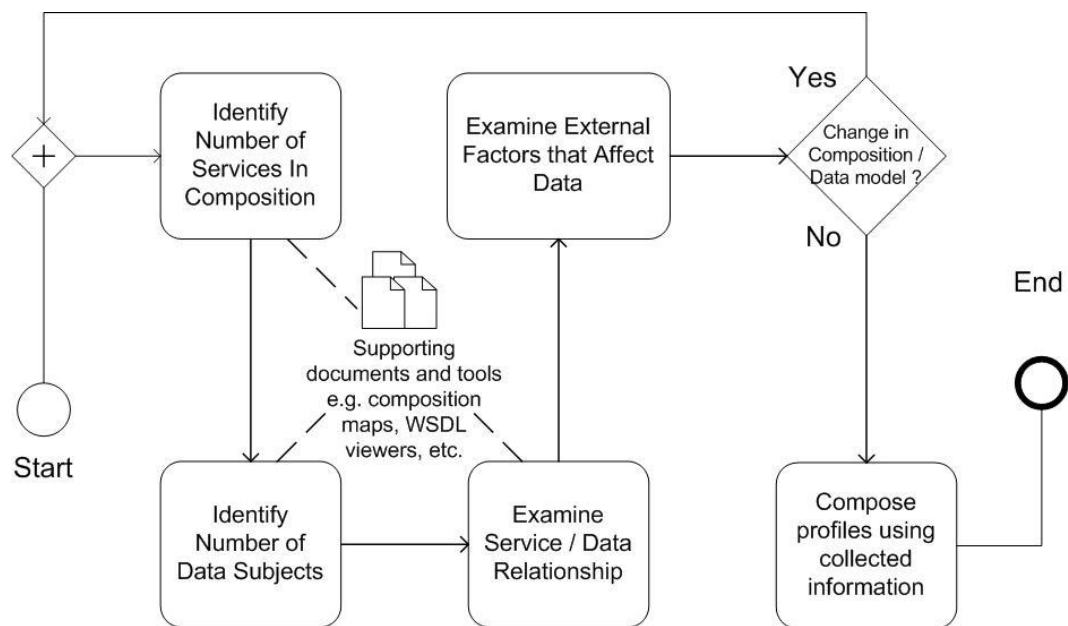


Figure 4-11: BPMN Diagram of the Profiling process in SOA

This method involves manual analysis of the data, services, and processes involved in composition. However, depending on the case, the process of building service profiles could be levered by the use of additional tools that are traditionally integrated into SOA middleware. Examples of such tools (see Figure 4-12) are IBM Websphere's Service Registry Graphical Explorer (Wahli 2006) and Liquid XML Studio's WSDL viewer (Liquid XML 2015).

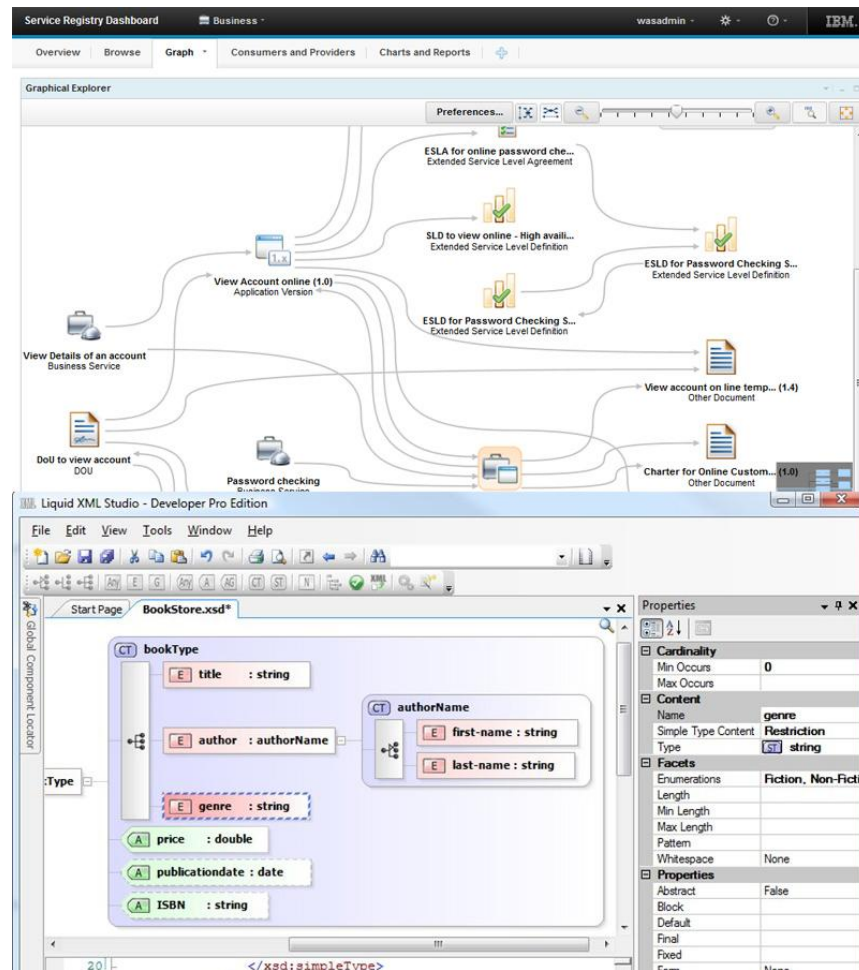


Figure 4-12: IBM WebSphere's Service Registry Graphical Explorer (top) and Liquid XML's WSDL viewer (bottom)

In situations where such features are not present, data analysis must rely on any available sources that describe service contract and information about the data – e.g. developers' sheets, design and data schemas, textual documentation, etc. To aid data analyst with examining the data relationships, this research proposes a template table that classifies service' (meta-)data into a “what”, “who” and “how” sections. As it can be observed from Table 4-1 the following services characteristics must be considered:

- **data** – indicating if it is either input or output, identifying its types, attributes, format, and structure;
- **processes** - including the steps and activities that are in charge of data modification and transformation;

Table 4-1: Information Profiling Table

<i>Who</i>	<i>How</i>	<i>What</i>				
Service	Process	Data Elements				
Service Name/Id	Operation/sets of operations	Element Name	Element Description	Element Type	Element Scale	Element Role(in/output)
{number id}	{function / operation name}	{variable name}	{description}	{type}	{unit}	{input / output}
...

Considering the Smart meeting room motivational case presented in section 4.3, Table 4-2 presents a simplified example of data profiles following the template described above. Note that information is not exhaustive and more detailed version of the table is presented in Chapter 6, section 6.7.1

Table 4-2: Data profiles in SOA Smart Meeting room (simplified)

<i>Who</i>		<i>How</i>	<i>What</i>				
Service		Process	Data Elements				
Service Name	Id	Operation/sets of operations	Element Name	Element Description	Element Type	Element Scale	Element Role
<i>Service Brightness Sensor</i>	<i>0002</i>	<i>Operation GetBrightness</i>	<i>brightOut</i>	<i>Room Illuminance</i>	<i>int</i>	<i>lumens</i>	<i>out</i>
<i>Service Brightness Sensor</i>	<i>0004</i>	<i>Operation GetShutterPosition</i>	<i>posIn</i>	<i>Shutter Position</i>	<i>int</i>	<i>percentile</i>	<i>out</i>
<i>Service Control Temperature</i>	<i>1001</i>	<i>Operation TemperatureControl</i>	<i>tIn</i>	<i>Room temperature</i>	<i>float</i>	<i>degree Celsius</i>	<i>in</i>
...	

After a successful analysis of the service environment, a process of defining and examining factors that affect data follows. Section 4.3 defined the meaning of Contextual data. It also outlined its role in constructing “quality” of information. Reflecting this notation, during this part of the profiling process, system/data analyst must take into account all external factors that influence service oriented environment or in a way affect its data. Table 4-3 identifies key areas service information needs structured against.

Table 4-3: Variables Dependencies and Descriptions

Data Subject (assessed variable)	Dependant/External variable	Service	Operation	Description dependencies
{data under scrutiny}	Dependant variable or external variable	Service Name	Operation/sets of operations	Description of the dependencies

Similarly to the Data Profiles, if the case of Smart meeting room is considered, an exemplary Table 4-4 containing the “room temperature” variable and its dependable variables and factors is presented. Analogously to data profile, table presented below does not provide full information about relationship and variables involved in the entire case, however, it portrays a good example of how information can be structured for better readability and clarity.

Table 4-4: Variable Dependency in Smart Meeting Room Case

Data Subject (Variables)	Dependent variables/external factors	Service	Operation	Dependencies/Description
1 Room Temperature		ServiceGetTemperature	OperationGetTemperatureSensor	Gets room temp
		ServiceSetTemperature	OperationSetTemperature	Sets room temp
		ServiceControlTemperature	OperationTemperatureControl	Sets room temp
		ServiceAmbientComfort	OperationSetAmbientComfort	Gets and sets room temp
			GetOutsideTemperature	
	OutsideTemperature	WeatherService	GetLocalTemperature	Room temperature would increase if outside
	Outside Illuminance	ServiceGetExternalIlluminance	OperationGetExternalIlluminance	Room temperature would increase if sunshine is directly passing through the
	ShutterPosition	ServiceShutterPosition	OperationGetShutterPosition	Room temperature would increase if shutters are open and windows are directly lit by the sunlight
2 Room Illuminance level	Bulb power	ServiceLightBulb	OperationSetLightBulbPower	Sets light bulbs powers
			OperationGetLightBulbType	Gets light source type
	ShutterPosition	ServiceShutters	OperationSetShutterPosition	sets shutters position

Executing this process accurately is essential. Not following the steps described previously might have a major impact on the subsequent stages and eventually on the final outcome. One should note that an error in this stage can result in unsatis-

fying data evaluation. Additionally, the person in charge of building information profiles must be aware of the scale of the DQ project and that complexity of the process will exponentially grow with the higher number of services and data subjects.

In summary, building information profiles does not end to with examining and stripping apart particular service but rather understanding the overall composition model, the dependencies of the services and the way these services handle the data. This stage will serve as an input for the assessment stage that ultimately will enable detecting the semantically inaccurate data.

4.6.2 Preparation Stage – Defining “Quality” of Data

According to the International Standards Organisation (ISO) the definition of ‘quality’ is “the totality of features and characteristics of an entity that bears on its ability to satisfy stated and implied needs” (ISO 8402-1986, Quality-Vocabulary). Quality, in industry terms, is generally accepted as a “conformance to valid requirements” (Y. Wang et al. 2001). Preparation stage is that part of the DQ process where actual ‘quality’ of data is defined. It includes activities that help DQ users to establish the levels of conformance. Before defining quality, however, few activities must be carried out:

- Determine who sets requirements
- Establish the levels of conformance that are needed
- Determine how the requirements are set

In section 2.4, raw information/data was described as a ‘product’ that has a description of what the consumer needs. When the product meets the consumer’s requirements, it is said to have achieved quality. Originating from this statement, it is the data consumer who ultimately determines the quality of information. However, data consumers could have a different view of the quality. For example, if considering a big enterprise business case; the business’ view of quality could significantly differ from one of its customers. Consequently, it is the business’ job to determine the quality on behalf of the customers and its responsibility to set the quality requirements. Most of the time there is not a single person responsible for specific sets of data. In such cases, the person responsible for the data in question

needs to be identified and that person's authority to make decisions concerning the quality requirements needs to be recognized. In large companies, the IT team collaborates to this process by ensuring that the business person making the decision is aware of the existing data quality deficiencies and the practical cost of overcoming them. If we take as an example the Smart meeting room case introduced earlier in this research, the person in charge of defining data quality would be the users of this room, possibly aided by the system administrators.

In order to elaborate further on the other two activities – “determine how the requirements are set” and “establish the levels of conformance”, a process was designed using BPMN specification as shown on Figure 4-13. It describes the actual sequence for defining the data quality in the SOA context and it consists of four steps namely 1) selecting data subject/entity, 2) extracting relevant information about selected data subject 3) constructing quality statements 4) linking quality statements and saving them into a repository. The following paragraphs explain each sub-step more in details.

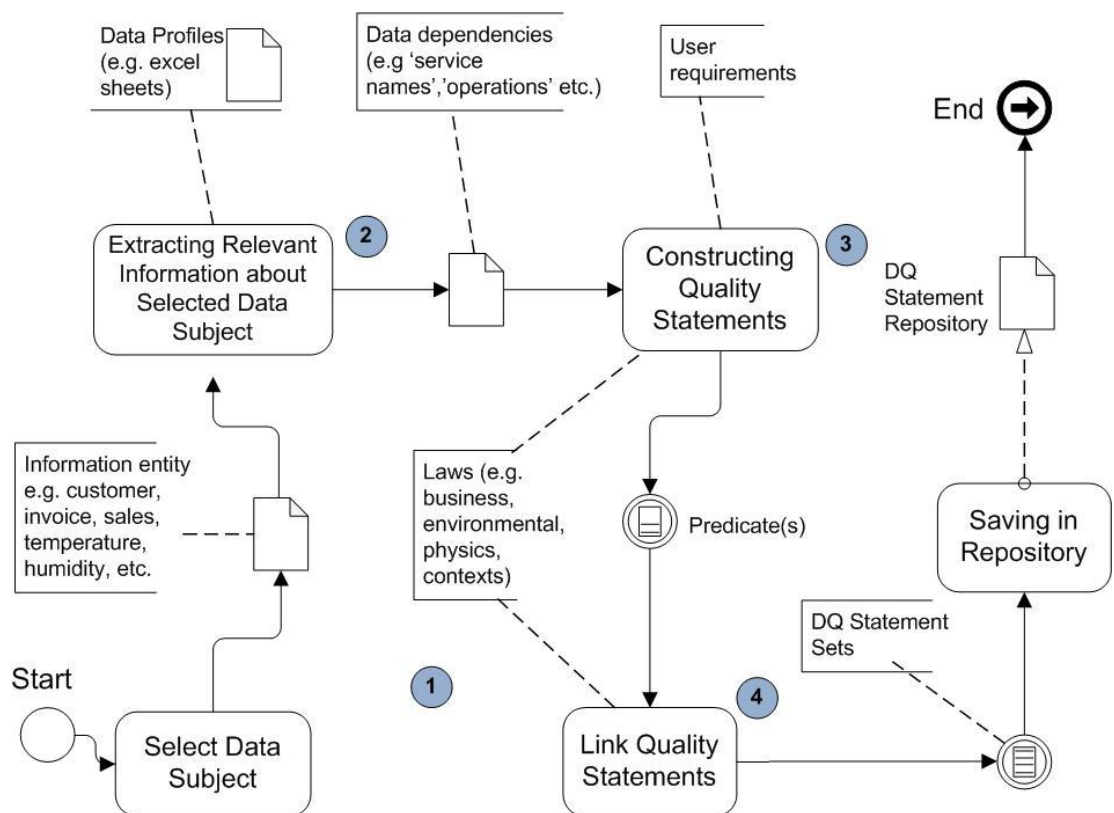


Figure 4-13: Process of Defining “Quality” in SOA

The process of preparation begins with careful selection of data subjects that are required to be quality scrutinized. This stage is tingly connected with the situation (e.g. the domain settings). A situational factor while selecting data subject could be its “relevance” e.g. how often they are referred to in the particular case. For example, examining the quality of “monthly sales” data subject is more important than the “annual sales” one. Other deterministic factors could be, for instance, the group of people that consumes data subjects, the devices that access this data, etc. Many times selection process requires prioritization of the data models. This activity requires the body in charge executing it to be a domain expert, or at least have solid knowledge in the area. Studies that propose approaches for identification, selection, and prioritization are carried out by Foley (2011); R. Wang, Storey, and Firth (1995); Y Wand and Wang (1996). Considering the case in section 4.3., variables that could be identified as important to monitor/assess are the temperature since it is of high significance for the guests’ comfort as well as the room illuminance since it influences the quality (e.g. brightness) of the projected presentation

Next task of the “quality” definition process is to extract information about targeted data (2). In this step, data consumer/analyst uses the information provided by the data profiling stage. The goal is to collect all necessary information (metadata) regarding observed data subject. To do so, information profile of regarded data must be fetched from the source e.g. excel sheet file and then thoroughly examined. As described in section 4.6.1, profiles of data subject must contain a comprehensive list or a map of dependencies in terms of services and its operations. The document also must include any external factors (e.g. data, functions, laws, people, environment, business rules, etc.) that alternate data. Examples of such documents were presented in the previous section (see Table 4-2 and Figure 4-7)

After profiles and data dependencies are examined the process of constructing data quality statements follows. In this stage, data user/analyst craft quality statements upon the information learned in data profiling stage. In section 4.4.1 quality predicates were referred as data constraints. Semantic constraints are meaningful statements about the data. In natural language, a predicate is the part of a sentence

or clause that contains a verb and states something about the subject. Predicate theory is based on propositional and predicate logic. Propositional logic is declarative and it assumes that the world contains facts. From this perspective, a predicate is a statement about *something* (e.g. data subject) and it can be either true or false. On the other hand, predicate logic defines predicates as relations (functions) over arguments. The role of the predicate here is either to assign a property to a single argument or more arguments to each other. It is also possible to compile multiple predicates.

To explain better the relationship between predicates, quality statements and Data quality, consider the following example which is based on scenario from the Smart Meeting Room case accentuated in Section 4.3:

***“Luminance is less than 4000 Lumens when
It is a Day time and
The Sky is Cloudy and
The shutter blinds are 100% open and
The room lights are off”***

The sentence above represents a natural language statement that defines the amount of light that enters the room under specific conditions. As seen in Table 4-5, this complex statement can be separated into two parts: 1) the main statement (S) which contains the variable under scrutiny; and 2) the contextual statements (CS1-CS4) which represent a collective declaration of conditions supporting the main statement. To ensure the quality of the Luminance variable (e.g. to meet the requirement its value to be under 4000 units), however, first all supporting statements need to return a positive (in this case) outcome, and second - the Quality of the Context statements (QoC) must be taken into consideration. To assess QoC of a particular contextual variable, it is suggested that data is collected from multiple different and independent sources. Alternatively, statements about the CS could be defined as to ensure the relationship between CS and other variables. In the case with the smart room, the QoC of the CS2 statement - “The sky is Cloudy” can be verified by obtaining weather conditions from at least three independent

weather services. Additionally, other services implicitly related with the CS2 (e.g. connected via third party service) can be explored to secure quality of the CS2.

Table 4-5: DQ Statement Decomposition

<i>“Luminance is less than 4000 Lumens when</i>		S
<i>It is a Day time and</i>	Context Instance	CS1
<i>The Sky is Cloudy and</i>		CS2
<i>The shutter blinds are 100% open and</i>		CS3
<i>The room lights are off”</i>		CS4

Considering predicate theory introduced earlier, each individual statement as defined in the table above can be represented in the following way:

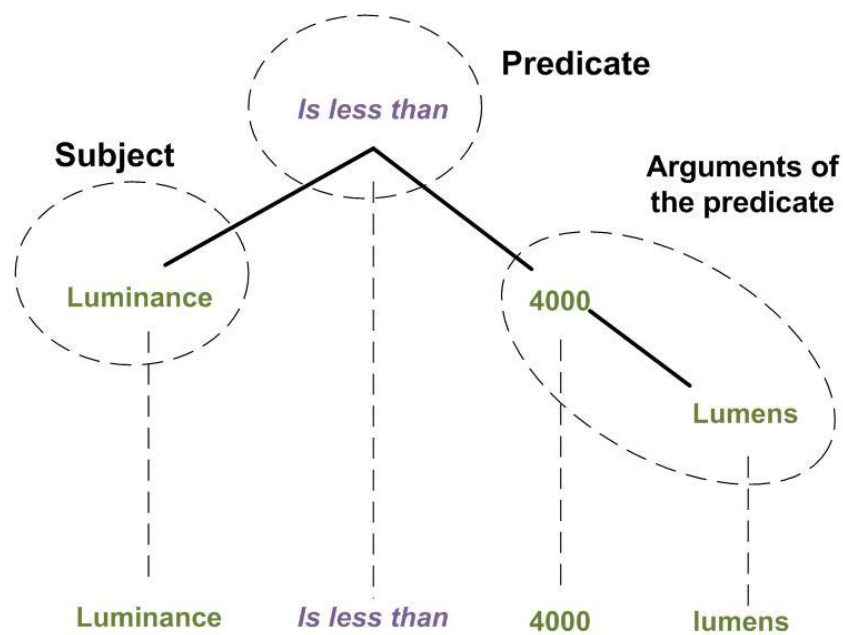


Figure 4-14: Example of a Predicate and its Arguments

As depicted in figure, the subject of this statement is “the luminance”, the predicate is “is less than” and the additional information about the subject are the arguments specified by the predicate – “4000 Lumens”. Considering this breakdown, aforementioned sentence can be expressed in the following machine executable manner:

```
is less than(Luminance, is(4000, Lumens));
```

where “is less than” defines the predicate function containing the arguments “luminance” and “4000 Lumens”. Note that the second argument is a complex argument (e.g. it is composed of few simple arguments). In this case, relationship within the complex argument is implied by the predicate function `is(4000, Lumens)`. However, in complex cases, relationships must be explicitly defined. This process is described in the next stage ‘Link Quality Statements’ of DQ process.

As it was mentioned earlier predicates are statements about data (including the contextual data) that could have a value of true or false. The goal of this sub-stage is to build a list of simple predicates (statements) that are going to be linked together in order to form a complex one. A simple predicate (contextual) statement is, for example, “*The shutters are 100% open.*” To build these predicates, data analyst must take into account users’ preferences about subject data. Then, he needs to identify service functions that retrieve subject’s data values. If functions are named according to General-purpose Programming Language (GPL) conventions, they will contain the prefix ‘**get**’, ‘**return**’, ‘**fetch**’ in their names (e.g. *ServiceBrightness.getLuminance()*). Ultimately, these functions will be used in the execution stage as described in section 4.6.3 to compare the *returned* and *required* (the one defined by the data user) values. After analyst identifies all ‘data fetch’ functions, he must link them with user’s requirements i.e. to construct predicates. To define quality or “the degree of excellence of something as measured against other similar things”, the predicates (the verbs) used in this process must contain comparative expression logic. Such predicates are ‘is’, ‘is not’, ‘is less’, ‘is more’, ‘is bigger’ ‘is smaller’ etc. From computer science perspective, these can be brought down to six logical operations: ‘<’(less), ‘>’(greater), ‘=’equal, ‘!=’(not equal), ‘<=’(less than or equal) ‘>=’(greater than or equal). As it was emphasised earlier in this section, the body in charge of definition process must also take into considerations any external factors that might “infect” subject data.

If DQ is a critical factor and SOA middleware does not feature any data quality mechanism, then it is the SOA architects and developers’ responsibility to deliver a feasible solution. The code snippet in Figure 4-15 represents a semi-structured

way of storing data quality statements in a repository using XML schema. The composition of this schema is inspired by the XML WSDL format.

```

<simpleDQ id="int" date="date" name="statement_name_string">
  <description> [statement description] </description>

  <serviceName> [service name] </serviceName>
  <servicePort> [service port] </ servicePort >
  <serviceInstance> [service id] </ serviceInstance >
  <serviceOperation>[service' getOperation]</serviceOperation>

  <predicate> [comparator: >,>=,<,<=, =, !=,] </predicate>
  <argument> [expectedvar:int,string] </argument >
</simpleDQ>

```

Figure 4-15: XML snippet code for storing simple DQ statements

As seen, this structure contains all the elements necessary to construct a primitive DQ statement. It features the subject data (described through its service and fetch function), the predicate and ultimately its argument. A simple DQ statement also must have a unique identifier and a date stamp (e.g. when the statement was created or updated). It is also a good practice that data users name statements accordingly to their intent to use. Example of a simple statement is presented in Figure 4-16.

```

<simpleDQ id="000012" date="094413022014" name="ShuttersPosIs0">
  <description>Check if shutters pos is zero</description>
  <serviceName>ServiceShutters</serviceName>
  <servicePort>http://1.0.0.19/doha/big</servicePort>
  <serviceInstance>0053</serviceInstance>
  <serviceOperation>OperationGetShutters</serviceOperation>

  <predicate>=</predicate>
  <argument>0</argument >
</simpleDQ>

```

Diagram illustrating the structure of the XML snippet in Figure 4-16:

- Statement meta data:** Includes `id="000012"`, `date="094413022014"`, and `name="ShuttersPosIs0"`.
- Service Metadata:** Includes `serviceName="ServiceShutters"`, `servicePort="http://1.0.0.19/doha/big"`, `serviceInstance="0053"`, and `serviceOperation="OperationGetShutters"`.
- DQ predicate, Expected output:** Includes `predicate="="` and `argument="0"`.

Figure 4-16: Example of Simple Statement in the Smart Room Case

The final step of the quality definition phase is constructing complex data quality statements and storing them in a suitable way for execution in the next phase. The word ‘suitable’ is used in terms of the capability of the software tool to automatically and successfully read semi-structured files during the Execution Stage. Data users can compose complex statements by linking two or more simple ones. ‘Links’ are relationships between simple statements. They can be obtained through data dependencies sheets or described by different laws e.g. natural, business, physics, etc. These relationships, on their own, also represent predicates – but complex ones, hence they must obey the predicted theory principles. Originating from this, predicates can be linked using logical connectives. Such connectives in the natural language are “AND” “OR” and “NOT” linking words. In IT and mathematics, logical expressiveness is levered by the Boolean algebra (Manning and Raghavan 2009). Boolean algebra is a branch of computer science and algebra in which the values of the variables are either true or false. Main operations of Boolean algebra are the conjunction AND denoted ‘ \wedge ’, the disjunction OR, denoted ‘ \vee ’, and the negation NOT, denoted ‘ \neg ’. A complex data quality statement must contain at least two simple statements and must not end with Boolean operator. Generalized format a complex statement is demonstrated in Figure 4-17.

```

complex_statement
{
    simple_statement_1 boolean_operand
    simple_statement_2 boolean_operand
    ...
    simple_statement N;
}
end of complex statement

```

Figure 4-17: Structure of a Complex Data Quality Statement

From a technical point of view, formats and procedures for storing data quality statements may differ in diverse situations. For example, a tree structure can be used to represent the sets of DQ statements where the simple statements are stored in its leaves and predicates links (in the form of Boolean operators) in their roots.

Another structure for storing predicates could be hash tables or relative database tables. Nevertheless, the main idea is that the method and structure chosen for storing the DQ statements are efficient enough to be executed in the next stage. This research suggests that predicates are stored in an external repository in a semi-structured way (e.g. XML, or Excel, format). This approach would reduce system dependencies, time to execute statements and would make it loosely coupled to other data services.

```
<complexStatement id="int" date="date" name="statement_name_string">
  <description> [complex statement description] </description>

  <simpleDQ id="int" />
  <booleanOperation>[boolean operation: AND OR NOT]<booleanOperation>
  <simpleDQ id="int" />
  <booleanOperation>[boolean operation: AND OR NOT]<booleanOperation>
  . . .
  <simpleDQ id="int" />
</complexStatement >
```

Figure 4-18: XML snippet of DQ Complex Statement

```
<listStatements id="int" date="date" name="statement_name_string">
  <simpleDQ id="int" />
  <complexStatement id="int" />
  <complexStatement id="int" />
  <simpleDQ id="int" />
  . . .
  <complexStatement id="int" />
</listStatements>
```

Figure 4-19: Semi-Structured Approach of storing DQ statements

Figure 4-18 defines the structure of a complex Data Quality statement and, as it can be observed, it reflects the model depicted on Figure 4-17. Similar to the simple DQ statements, complex ones must also have a unique identifier and a time stamp. An example representing the complex statement defined earlier in this section - in Table 4-5 is described by Figure 4-20. Ultimately, all statements are saved in batches as outlined in Figure 4-19. Batches are lists that contain not only

complex but also simple data quality statements. Generally, it is a good practice if the body in charge of this process always indicates his reasons of constructing particular DQ statement. He can do so by filling out the “description” field supported by the XML structure.

```
<complexStatement id="100021" date="140213022014" name="LuminanceCheck4000">
  <description>Check if Luminance is value is correct</description>
  <simpleDQ id="000010" />                                (Main statement)
  <booleanOperation>AND<booleanOperation>
    <simpleDQ id="000014" />                                (CS1)
    <booleanOperation>AND<booleanOperation>
      <simpleDQ id="000013" />                                (CS2)
      <booleanOperation>AND<booleanOperation>
        <simpleDQ id="000012" />                                (CS3)
        <booleanOperation>AND<booleanOperation>
          <simpleDQ id="000015" />                                (CS4)
    </booleanOperation>
  </booleanOperation>
</complexStatement>
```

Figure 4-20: Example of Complex Statement in the Smart Room Case

4.6.3 Execution Stage – Applying Data Quality Statements

Following the Data Quality Management cycle, the next phase of the DQ methodology is the Execution stage. This is the point where data quality statements, defined in Preparation stage, are executed through the service environment. However, as it was stated in before, SOA solutions, structure, and technologies may vary between different cases. This imposes two important requirements for the execution process. On the one hand, it has to be system independent – i.e. it should be flexible enough to be applied in as many situations as possible. On the other hand, it must be comprehensive enough (e.g. detailed enough) to aid SOA architects and developers with designing and integrating a tool for execution of DQ statements.

Taking under consideration aforementioned, I propose a process of execution DQ statements as illustrated in Figure 4-21 using the BPMN annotation. The Execu-

tion stage is comprised of three steps: (1) Reading the Data Quality statements from the repository; (2) Building and Executing service' queries based on the DQ statements; (3) Waiting services' response and aggregating results; and (4) Saving results from execution in a form of log files. The next few paragraphs describe how DQ statements are applied and executed through service oriented environment.

The process of execution begins with reading the quality statements, previously stored in the repository. After reading a DQ statement and before being executed through the composition, it is necessary that it undergo a series of condition's checks. Condition blocks are represented on Figure 4-21 - 1. The first condition – “if the DQ statements are up to date” - ensures that particular statement matches to the real world situation. For example, if some of the services or its retrieval operations that have been modified, the quality statement regarding data subject has to be updated. To check timeliness of the statement, services and processes operating with the subject must be taken into account. If the statement is out of date, the process suggests returning to Preparation stage where invalid DQ declarations are either updated/modified or deleted. If the statement is up to date, then next condition is followed – “if statement data is being used / modified by other services”. Applying this condition to the statements will later ensure its accurate execution, hence accurate detection of the poor data. In most cases, this block of the process is embedded into SOA process engine, however, if not, it needs to be taken into consideration and consequently implemented. Particular attention should be paid in regards to the long-running business transaction when data could be unavailable due to processes running outside the scope of accessibility. Generally, a data that is not available cannot be assessed. In cases of long-running transactions processes, a “time out” value is set by the DQ administrator as will be seen later in the Execution process. If requested transaction does not return a result within specified allowance, (e.g. the instance of) DQ process is terminated. In cases there are multiple services involved in performing the operation over data, then all services affecting this data are considered (listed) for the reporting stage.

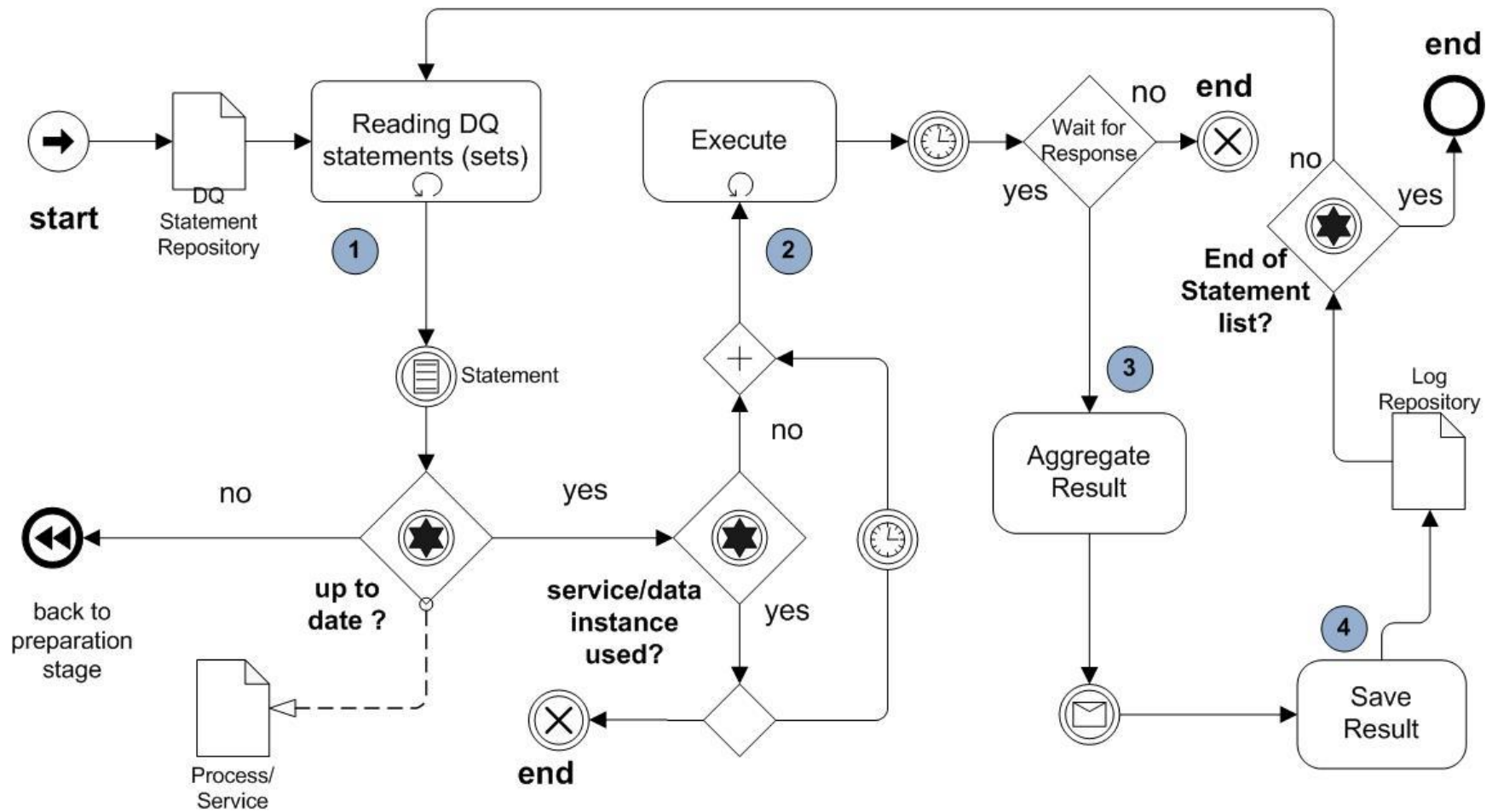


Figure 4-21: BPMN process of Execution Stage applied to the Service Environment

After the statement is being read and checked against the conditions stated above, the actual process of execution (Figure 4-21 - 2) is followed. Before actual execution, however, quality statement(s) need to be transformed in a compatible with particular SOA environment way for operation. Enquiry (-ies) is then composed and dispatched in the form of envelopes. In most of the cases, dispatching envelopes is done by message router block. The Router block is essential in every SOA and its function is to coordinate service' messages. Depending on the type of infrastructure (e.g. centralized, decentralized), standards and protocols envelopes can have a different format. The transformation from quality statement to enquiry is limited to the given case.

In case DQ statement is a complex one, multiple enquiries need to be placed. This on its own suggests cyclicity, however, considering the structure of complex DQ statements (e.g. tree structure), it is a good practice that the order of inquiries is executed in a recursive manner. This is a composite process and it requires that all services return a result within a given time frame. If any of the simple enquiries composing the complex one fail to deliver the result (e.g. due to unavailability or great delay), then the results of the complex statement composing it are considered invalid and this instance of the process is terminated.

The last step of this execution process is aggregating the results and saving them into log files. The Aggregator block (Figure 4-21 – 3) handles the collection of generated multiple enquiries. Its role is to wait for enquiries to be served and then, based on the Boolean operands stored in each complex DQ statement, to calculate the final result. The result is then saved in a log file (Figure 4-21 - 4). The format of this file may vary depending on the SOA case. However, as in Preparation stage, a semi-structured approach is suggested to store the results. This would make it as it convenient for analysing in later stages. Current approaches such as the Common Log Format and Combined Log Format of W3C (WC3 2015) are useful but not efficient enough to directly store captured data after execution. To address this issue, this research pro-

poses a practically oriented log file which will contain information such as id, name, instance; address etc. XML log structure is presented in Figure 4-22.

```
<Entry>
  <service_location> ... </service_location>
  <service_ID> ... </service_ID>
  <instance_ID> ... </instance_ID>
  <endPointName> ... </endPointName>
  <operationName> ... </operationName>
  <dataEntityName>... </dataEntityName>
  <data_Value>... </data_Value>
  <statement_id>... </statement_id>
</Entry>
```

Figure 4-22: XML Log Structure Format

The process described in this chapter defines the behaviour of the (SOA) middleware when executing of DQ statements. Correspondingly, this part of the DQ process contributes more to SOA Architects, System developers and Support than to Data users and analysts. As outlined in Section 4.7 – “ Method for Incorporating DQ Methodology within SOA”, architects and developers play important in SOA life cycle and without their aid any ideas and features will be impossible to implement. This implies that it is their responsibility to define specific requirements and ultimately develop or integrate a module (e.g. software tool) that will lever the process of execution of DQ statements. As mentioned several times in this document, a specific implementation depends on the use case and SOA settings. Later in this study, a DQ service has been developed and incorporated into a system for home automation which follows service-oriented principles.

From data analyst/user point of view, the process of Execution is automatic and does not involve any intervention. The only control he has is whether or not to start execution process. He also must have the opportunity to define execution time intervals – e.g. system runs the execution process every 5 hours. As it can be noticed from the process diagram shown on Figure 4-21 there are delays in execution caused by the

system and services which cannot be avoided. Data analyst must take this into account while analysing data outputs for timeliness.

4.6.4 Reporting Stage – Analysing Results

In this task, the results are compiled, annotated, and summarized for presentation and discussion with the business and project team. While the format will be tool-dependent, it will most often consist of summary tables and graphs from which it is possible to drill down into the supporting detail data. How and to whom the results are reported should be documented up front in the data quality analysis plan and indicated in the execution plan. Data analysts should ensure that consistent and standard annotations are created for results. Failure to do so may cause others to unnecessary review and potentially redo investigation tests and work. Where results fail to meet the identified requirements, or in cases where results conflict with preconceived expectations, it may be necessary to iterate the tests or to add supplementary tests to explain the results.

As with any project, these additional tests represent potential scope change and must be reviewed for risk and impact within the available timeline. The availability of automated tools at this point may allow for such scope expansion without significantly impacting schedules. Where significant problems are discovered, there are three options available to the SOA designer:

- Change the service implementation to mitigate the data problems.
- Request that a change is made outside of the SOA project team. This may be a catalyst for initiating an enterprise data quality initiative in the organization if one does not already exist.
- Choose not to expose the service on the grounds that it cannot meet business service level requirements.

Analysing is the phase where generated results from Execution stage are examined and investigated. This stage can be also referred to the quality monitoring and transformation stage in DQM. Previously a convenient services-oriented problem log file

has been produced after execution of DQ statements. In the analysing stage, summary reports based on the user preferences and data collected from the execution stage are generated. Depending on the business architect/user intentions and criteria chosen (e.g. “violated statement”, “service”, “process” etc.), different reports are generated. It is important to note that this stage does not present a merit of where exactly actual problem stems from, however, it aids the analyst investigation the source of bad data. It is possible multiple data quality violations may emerge and localizing the source of the error is data analyst’s job. Also, analysts must be aware of the possibilities of false trues and true fales results, hence it is strongly suggested experiments are repeated (e.g. re-executing statements). Additionally, the analysis of reported issues could be done base on real-time or delayed/scheduled data (statement) execution as defined in the beginning of this research. Real-time detection is characterised by the ability of the system to identify problems with poor data as soon as they occur whereas delayed or scheduled execution allow data quality to be checked only at particular moment of time or at certain intervals of time. Finally, it is very important for data analysts to know that the efficiency of the data quality program (e.g. how well poor data will be detected) is based on how well they (or data consumers) define “quality” – i.e. how well quality statements cover examined DQ subject variable.

4.7 Method for Incorporating DQ Methodology within SOA

In order to answer the research question “How can DQ methodology be incorporated within SOA?” and more specifically - “How to implement a DQ service/tool?” we first need to explore methods and strategies which indicate possible ways of developing products. One approach that might help to answer the inquiry above is to use and accommodate the mainstream SOA methodology for implementing SOA solutions. As part of SOA composition methodology, literature identifies two main strategies – the bottom-up and the top-down ones. Nevertheless, the definition of these strategies varies among different research studies. This indicates that there is no commonly accepted view among peers. Additionally, the purpose of top-down and bottom-up strategies is to support business people evaluating costs, estimating delivery times as

well as addressing key decision points related to weighing the attainable strategic benefits of SOA. In other words, these strategies describe the process of caring out SOA projects from the business perspective only. All these facts make aforementioned approaches not suitable starting points for practical incorporation of DQ methodology within SOA project.

After reviewing the literature, Reverse Engineering (RE) was identified as possibly suitable approach that would aid the development and incorporation of DQ module within SOA project by canonically detail every step. Before making any further argumentation why and what form of reversed engineering was accommodated, it is important to establish a background frame of references against which my decision can be positioned. It is important to note that definitions the techniques taken under consideration in this sections lies entirely within the domain of Information Systems but also tightly related with Software engineering domain. Reserve engineering of hardware is not discussed.

The idea of reverse engineering comes from software maintenance where, normally, the system's maintainers were not its designers, so they must use techniques to examine and learn about the system. In this context, reverse engineering is the part of the maintenance process that helps to understand the system so appropriate changes could be taken. The concept of service oriented architectures is multi-layered and often there is a distinctive separation of levels of operation e.g. service's (entity) managers and architects, SOA middleware developers and support, project managers, etc. Moreover, due to the dynamic nature of the service environment, different new services are being integrated and other - simply obsoleted. In the same time, new technologies emerge and current protocols and standards quickly become out of date. The same line of thoughts applies to the people involved into SOA project - often staff come and goes due to restructuring, reorganization or prequalification.

To comprehensively explain the notion of forward and reverse engineering approach, three dependent concepts must be first identified: the existence of a life-cycle model of SOA, the presence of a subject system, and the identification of abstraction levels.

The subject system may be a single program or code fragment, or it may be a complex set of interacting programs, job control instructions, and data entities. The concept of service orientation encapsulates nearly all of the aforementioned subjects. Consequently, the term subject system in this context refers to a service composite environment that includes different single services' entities as well as a middleware that integrates and interconnect these entities. Additionally, the subject system is the result of the design and later development process. It may not yet exist, or its existing components may not yet be united to form a system. In reverse engineering, the existing subject system is generally the starting point of the observation.

In a life-cycle model, the early stages deal with more general, implementation-independent concepts; later stages emphasize implementation details. The transition of increasing detail through the forward progress of the life cycle maps well to the concept of abstraction levels. Earlier stages of systems planning and requirements definition involve expressing higher level abstractions of the system being designed when compared to the implementation itself. It is important to distinguish between levels of abstraction, a concept that crosses conceptual stages of design, and degrees of abstraction within a single stage. Spanning life-cycle phases involve a transition from higher abstraction levels in early stages to lower abstraction levels in later stages. While you can represent information in any life-cycle stage in detailed form (lower degree of abstraction) or in more summarized or global forms (higher degree of abstraction), these definitions emphasize the concept of levels abstraction between life-cycle phases. The service oriented cycle depicted in Figure 4-23.

The life-cycle phases shown above in can be also divided into the following general groupings:

- **Predeployment.** During this phase, a business need is translated into a deployment scenario: a logical architecture and a set of quality of service requirements. The deployment scenario serves as a specification used to design deployment architecture.

- **Deployment.** In this phase, the deployment scenario is translated into deployment architecture. This architecture can be used as a basis for project approval and budgeting. The deployment architecture is also the basis for an implementation specification that provides the details needed to deploy (build, test, and roll out) a software solution in a production environment.
- **Postdeployment.** In the operations phase, the deployed solution is run under production conditions and monitored and optimized for performance. The deployed solution is also upgraded to include new functionality as necessary.

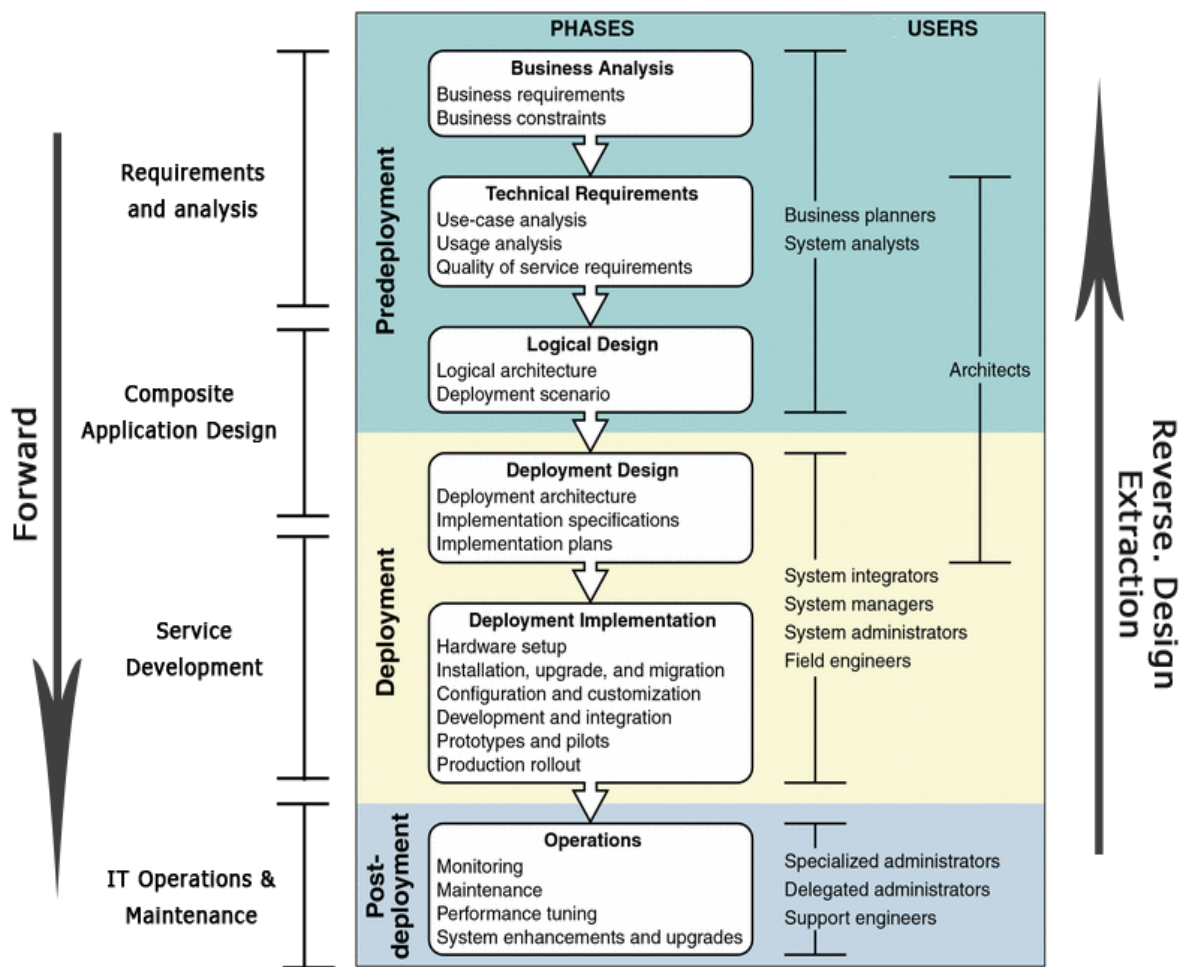


Figure 4-23: SOA Life Cycle and Reverse Engineering

Table 4-6 Role Groups in SOA life Cycle

User	Skills and Background	Phases
Business planner System analyst	<ul style="list-style-type: none"> • Has general rather than in-depth technical knowledge • Understands strategic direction of the enterprise • Knows business processes, objectives, and requirements 	Business analysis Technical Requirements Logical design
Architect	<ul style="list-style-type: none"> • Is highly technical • Has broad knowledge of deployment architectures • Is familiar with latest technologies • Understands business requirements and constraints 	Logical design Deployment design
System integrator Field engineer System administrator System manager	<ul style="list-style-type: none"> • Is highly technical • Is intimately familiar with information technology environments • Is experienced in implementing distributed software solutions • Knows network architecture, protocols, devices, and security • Knows scripting and programming languages 	Deployment design Deployment implementation
Specialized system administrator Delegated administrator Support engineer	<ul style="list-style-type: none"> • Has specialized technical or product knowledge • Is familiar with hardware, platforms, directories, and databases • Is skilled at monitoring, troubleshooting, and upgrading software • Knows system administration for operating system platforms 	Operations

Requirements and Analysis: This stage aims to identify and prioritise the business needs. Based on the identified priorities, non-technical staff works closely with business analysts to document the business process, rules, and requirements. High-level requirements include: Visually map business process starting from Level 0 downwards; Define each of the business processes; Identify business owners for each of the processes; maps input and output data elements. Table 4-7 describes the deliverables on every stage along the actors and their roles) and technologies used to derive these deliverables.

Table 4-7: Deliverable and Tools used in each stage of SOA cycle

Stage	Actors(users)	Tools used	Artifacts (Deliverables)
Requirements and Analysis	Business personnel (typically business operations from LOB) Project managers (business & IT) Business analysts Architects (optional)	Business requirements tools including office, business process modelling tool, requirements capturing tools Business process modelling tools including BPMN, Visio, and Pro*Activity Business rules tools including product rules engines and Word User Interface tools including portal simulation tools, Flash, Visual Studio, Eclipse, and JSP and HTML editors	Design models such as UML, BPM (business process models), data flow models Bindings such as JMS, RMI, IIOP, and HTTP(s)
Composite Application Design	Project manager (IT) Business analysts Enterprise architects Project architects Designers Technical leads or lead developer	Design: Rational, Together Architecture, Eclipse, and others	Design models: UML, SCA service assembly model, and others Bindings: JMS, RMI, IIOP, HTTP(s), and others
Service Development	Project manager (IT) Business analysts Enterprise architects Project architects Designers Technical leads or lead developer	IDE (for example, Visual Studio, Eclipse, JBuilder, JDeveloper, and BEA Workshop) Packaged application development tools Regression and performance testing tools Build tools (for example, Maven, Cruise Control, ANT, and shell scripts) Source control management systems	Source code Product specific metadata for configuration as well as service execution Java documents Release notes
IT Operations	Project manager Architects Release management Build teams IT operations	Tools used include product deployment tools such as Maven, Cruise Control, Ant, shell scripts, and IDE-based tools such as Eclipse, Visual Studio, and packaged application deployment tools.	Chief deliverables are the product domain model, packed application configuration, builds, and service dependencies, DQ initiatives.

Composite Application Design: This stage of the service lifecycle generates models that represent the system flow, data flow, enterprise data model (represented in the form of Entity Relationship Diagram(ERD)), application design (represented in UML), activity diagram, and sequence diagram. During this phase, the team also generates the high-level deployment model, identifying the servers, OS, middleware, databases, firewall, and load balancers.

Service Development: The artifacts produced during this stage include service definition, configuration, contract, and application configuration, as well as management, security and business policies at the service, application, department, and enterprise level. The application configuration also includes implementation configuration.

IT Operations: The primary output of this process is a product domain model which teams can leverage to assemble builds for each of the services (if required), assemble all metadata for products, and map URLs that are physical end-points for running the services. Teams develop the physical deployment model from the logical model. Once the team has assembled the services for deployment, it needs to create the network topology, server configuration, and server asset management. In addition, the team must also create runtime artifacts such as monitoring and event logs for correlation later downstream.

In an extensive study Chikofsky et al. (1990) defined Reverse Engineering (RE) as a process of analysing a subject system to:

- Identify the system's/environment's components and their interrelationships;
- Create representations of the system/environment at a higher level of abstraction;

An alternative definition given by another pioneer in the field of reverse engineering' is that: "Reverse Engineering is a process to support the analysis and understanding of data and processing in existing computerized systems'. It aims to extract the con-

tents, structure, and flow of data and processes contained within existing information systems in a form amenable to enquiry, analysis and documentation” Jones (1988). In other words, Reverse engineering is an approach of obtaining information about a subject system. RE generally involves extracting design artifacts and building or synthesizing abstractions that are less implementation-dependent. It is important to point out that reverse engineering does not involve changing the subject system or creating a new system based on the reverse-engineered subject system. It is a process of examination, not a process of change or replication. Following paragraphs points some aspects when RE is needed:

- Volume of Code. Computer Systems have been developed over many years and there is a huge code base needing analysis by some method.
- Inconsistent standards. Over the last 10 to 15 years many different styles have appeared: modular programing; flow charting; e.g. structured programing, information engineering, etc. Few companies are in the lucky position where all their systems correspond to one set of standards.
- Systems not streamlined. As systems have developed over the years, new areas have been bolted on, but due to the lack of understanding of existing code, these add-ons have often introduced redundancy into the structures.
- Lack or limited documentation. Normally caused by the pressures data processing departments to produce immediate results, system documentation is at best out of date – in many cases non-existent.
- Resulting of the aforementioned statement - lack of Knowledge / Expertise. Staff may have moved on and there are often few people with in-depth knowledge of how the systems are put together. Indeed in some cases, the code appears to have been written to avoid understanding.

There are many subareas (approaches) of reverse engineering. Chikofsky et.al, however, differentiate two main subareas that are commonly referred. These are documentation and design recovery.

Re-documentation is “the creation or revision of a semantically equivalent representation within the same relative abstraction level” Chikofsky et al. (1990). The resulting forms of representation are usually considered alternate views (for example, data-flow, data structure, and control flow) intended for a human audience. Re-documentation is the simplest and oldest form of reverse engineering, and many consider it to be non-intrusive, weak form of restructuring. The “re-” prefix implies that the intent is to recover documentation about the subject system that existed or should have existed. Some common tools used to perform re-documentation are code processors (which display a code listing in an improved form), diagram generators (which create diagrams directly from code, reflecting control flow or code structure), and cross-reference listing generators. A key goal of these tools is to provide easier ways to visualize relationships among program components so you can recognize and follow paths clearly.

The other approach that is apparently of bigger importance for this study is the Design recovery/extraction. Design recovery is a subset of reverse engineering in which the main knowledge, external information, and deduction are added to the observations of the subject system to identify meaningful higher level abstractions beyond those obtained directly by examining the system itself. According to Ted Biggerstaff, “Design recovery recreates design abstractions from a combination of code, existing design documentation (if available), personal experience, and general knowledge about problem and application domains... Design recovery must reproduce all of the information required for a person to fully understand what a program does, how it does it, why it does it, and so forth. Thus, it deals with a far wider range of information than found in conventional software engineering representations or code”. (Biggerstaff 1989) Based on discussion above process of incorporating DQ process in SOA as software tool using RE is defined and it is depicted in Figure 4-24.

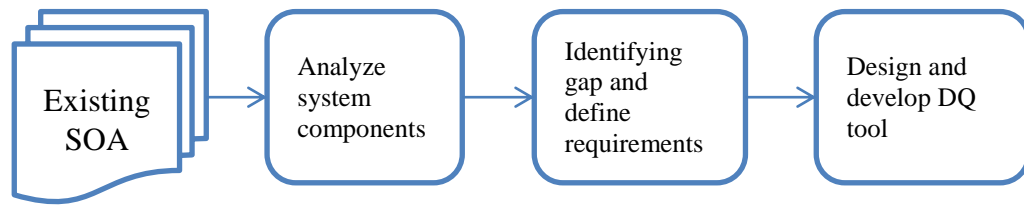


Figure 4-24: Incorporating new features into SOA using Reverse Engineering.

4.8 Summary

In this chapter, a framework for assessing data quality in SOA has been introduced. The solution consists of definitions, methods, step-by-step guidelines, as well as different factors and points that have to be taken into account while embarking on DQ project. More specifically, the data quality assessment framework is separated into three main sections.

- First section introduces the data quality to the service oriented environment. Then, the main concepts and terms of the DQ framework such as Total Data Quality Management, Contextual Data, and Data Constrains are discussed. The definition of above-mentioned elements is crucial since they act as foundations for this research solution.
- The second part of the DQ framework aimed to answer the main research question (RQ 2) in this study – “*How can be Data Quality measured in a SOA context?*” To do so it presents a process for detecting poor data quality in SOA. Particularly, it is divided into four sub-processes, each of which has its specific purpose and target.
 - The first stage of the process, namely “Building information profiles” presents a sequence of activities for identifying and gaining an understanding of the data to be assessed (e.g. subject data). This stage elaborates to answering RQ 2.1 – “*How to define ‘poor’ Data Quality within the context of SOA?*” by providing a process that involves in-

depth inspections of the subject data correlations and its dependencies in the environment.

- The second step of the DQ process is the “Preparation Stage”. At this stage data, consumers can explicitly define the quality of the data. This stage complements the answer to the RQ2.1 by providing a process that suggests series of specific steps (see Figure 4-13) to build data quality statements which are later used in the execution stage of the main DQ process. This sub-process also includes guidelines for SOA developers and architects in case middleware does not support DQ tool.
- The third step of the DQ process, namely “Execution Stage”, presents an algorithm for the execution of data quality statements within the SOA environment. The input of this stage is DQ statements defined in “Preparation Stage”. The main beneficiaries of this part of the DQ process are SOA developers and architects who are interested in the execution process and intend to implement it as a software tool.
- Finally, the last stage of the process is “Reporting stage”. This is the stage where generated reports, based on the data quality statements defined in the preparation stage, are analysed by data analyst or data consumer. This stage is also an anchor point for further data assessment and ultimately improvement. A summary containing the inputs, the outputs, the people’s roles and deliverables of each aforementioned process’s fragments are given in Table 4-8. The Execution Stage and Reporting stage answer to the research question RQ2.2, which in turn greatly contributes to main the RQ2 by explaining how to detect poor data in service oriented environment.
- The third part of the DQ framework answers to the question RQ 2.3 and RQ 2.4 – “*How to design and implement DQ assessment method within SOA context?*” The solution includes a method for incorporating the core DQ process (presented Section 4.5) into SOA which is based on the reversed engineering approach. It aims to deliver a systematic description of SOA environment and

present a way of integrating DQ as a software tool. This approach involves the re-examination and documentation (if necessary) of the SOA technologies and middleware prior to the development and integration of the DQ software tool.

Table 4-8: Summary of Data Quality Assessment process in SOA

Stage	Input	Output	Deliverables	Groups
Building Data Profiles	Data documentation, Relation diagrams, Any data supporting documentation	Structured Data Profiles. e.g. excel sheet documents	Provides a systematic way of profiling information in SOA. A starting point for Data Quality assessment process	Data Consumers Data Analysts SOA Developers, Architects and Managers
Preparation Stage	Data Profiles and Dependencies	Data Quality Statements	Provides a process for defining data quality. It supplies SOA developers and architects with guidelines for process implementation	Data Consumers Data Analysts SOA Developers, Architects
Execution Stage	Data Quality Statements	Data Reports	Provides developers and architects with sequence of activity for implementing the execution of data quality statements	SOA Developers, Architects
Reporting Stage	Data Reports	Decision Making	Process for analysing data quality reports	Data Consumers Data Analysts SOA Developers, Architects

Finally, Table 4-9 presents a summary of the research questions with references to the sections where particular question is addressed.

Table 4-9: Research Questions Addressed in Chapter 4

Research Questions Addressed in This Chapter		Section Reference
RQ 2	How can be Data Quality measured in a Service-oriented Architectures context?	Chapter 4
RQ 2.1	How to define “poor” Data Quality within the context of SOA?	Sections 4.4 , 4.5 , 4.6.1 and 4.6.2
RQ 2.2	How to detect poor data in service-oriented environment?	Sections 4.6.3 and 4.6.4
RQ 2.3	How to design Data Quality assessment method?	Sections 4.5 and 4.7
RQ 2.4	How to implement the Data Quality assessment method?	Section 4.7

Chapter 5 Incorporating Data Quality mechanism in Dynamic Home Automa- tion Environment

5.1 Introduction

By expanding the electronics and information technology field, today over four billion connected devices are distributed all over the world. Taking the average for the year 2013 every household in the United States possesses over 24 electronic devices reports the Consumer Electronics Association (TJ McCue 2013). Another study conducted by NDP Group from 2013 reports that Internet connected devices in the U.S. homes surpasses half a billion (Bogaty 2013). A lot of today's devices, such as routers, mobiles, LCD TVs, HiFi Receiver, DVD and BluRay player, phones, etc. come with networking capabilities. Even small, low-powered and battery driven devices, such as room thermostats to control heating and cooling systems, can be purchased with wireless transceivers to connect them in home automation systems. In general these devices use proprietary protocols for data transmissions and thus can only be used in conjunction with other devices from the same manufacturer. Interoperability with other devices is hardly possible.

By using the Internet Protocol (IP) as a common network protocol, interoperability (in some of their senses) can be achieved in the near future. Therefore, standardized application protocols on top of IP will be required. The Devices Profile for Web Services (DPWS), which includes W3C Web Services could be applied and offers several features, such as, eventing mechanisms, security and device discovery functionality. DPWS also brings an implementation of DPWS to very small, low-powered and battery driven devices.

The reminder of this chapter is structured as follows: next section begins with an in-depth discussion in the area of Ubiquitous Computing. More particularly, it presents fundamental concepts and standards embedded into UC as well as it discusses the relationship to SOA, UC and home automation systems. Subsequently, Dynamic Open Home Automation platform is presented. In this sub chapter underlying standards, protocols and technologies used in the platform are discussed. In order to implement data quality monitor, DOHA system and services are thoroughly examined. Following examination, collaboration was initiated with the industry partner and DQ monitoring service was developed. Additionally, to test the effectiveness of the tool, a home automation case was proposed. Eventually, this chapter concludes with summary, indicating the important points.

5.2 Ubiquitous Computing

Over the past decade, it has been a major increase of the electronic devices an average household uses. It is likely that in 2016 the amount of connected devices will increase to over 15 billion. The increasing number of connected devices claims for sufficient and efficient communication standards. Based on the idea of the Internet of Things (IoT) or Web of Things (WoT), the Internet Protocol (IP) is also drawing more and more attention in the area of embedded devices. Thus, the number of devices connected to the internet will increase rapidly as soon as IP becomes the core standard in fields of embedded devices like wireless sensor networks.

The Internet of Things, including the mass of resource-constrained devices, could benefit from the web service' architectures like today's web does. This is where the Devices Profile for Web Services (DPWS)(OASiS) comes into play. It identifies a core set of web service specifications for resource-constrained web service implementations and ensures interoperability with more flexible client implementations.

Ubiquitous Computing is a domain that encompasses various disciplines such as distributed computing, mobile computing, telecommunications, sensor networks, human-computer interaction, location sensing, etc. Lyytinen and Yoo define the concept of Ubiquitous computing as an integration of the characteristics of both mobile and

pervasive computing. However, they distinguish mobile computing from pervasive computing by highlighting two key property differences – embeddedness and mobility. Mobility encompasses the capability of computer services to physically move in space, while embeddedness characterises the capability of the computer to obtain the information from the environment in which it is. (Lyytinen and Yoo 2002). The relationship between aforementioned properties is presented on Figure 5-1.

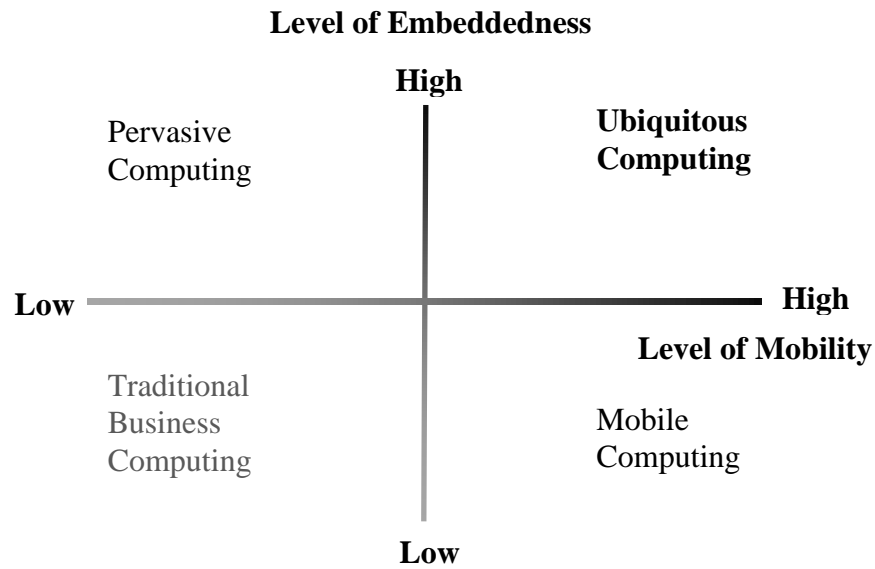


Figure 5-1: Dimensions of Ubiquitous Computing

Among IS community the concept of Ubiquitous Computing commonly relates to the term Ubiquitous Spaces. The idea of Ubiquitous Spaces envisions an environment that is populated with cooperating, interconnected devices which perform simple or complex tasks and are transparent (e.g. the way devices operates is hidden) to the user when achieving certain goal (Weiser 2002). The resources (e.g. devices, internet connection, etc.) of an ubiquitous system are dynamic by nature and they should be available to the users at any time, everywhere (Kutterer 2012). Widely accepted architectural style that supports the principles of the ubiquitous computing is SOA. As it was outlined in Chapter 2 – section 2.2.2, SOA characterizes with three distinctive properties: services are self-contain; services are platform independent; and services the can be dynamically discovered, invoked and recombined to form new services.

Stemming from SOA characteristics, functionally provided by devices in ubiquitous computing can be abstracted and encapsulated in terms of services, which may be combined with other services (devices) in order to create more complex applications (Stojanović and Dahanayake 2005)

Home automation is an emerging application domain that comprises both - the principles of UC, for instance, mobility and embeddedness, and the principles of Service-orientation – e.g. interoperability, loosely-coupling, transparency, etc. However, with emerging of home automated systems, problems concerning these principles also appeared. The heterogeneity of devices and network protocols, as well as diversity of hardware platforms and software frameworks, have led to the development of multiple solutions and commercial products; middleware such as Home RF and Konnex; software technologies such as Osgi, HAVI, etc. These events have arisen few important problems. Firstly, the availability of large number of middleware and commercial products does not ease the eventual user in his selection of a home automation solution. Counter wise, the presence of multiple solutions creates confusion among users as well as compatibility issues. To avoid such issues users are typically forced and limited to used devices and technologies offered by middleware/product provider. Secondly, issues concerning the lack of security and monitoring, high cost of installation, deployment, maintenance and security of home automation systems have restricted the full acceptance of such systems by the final users and building promoters.

Generally speaking, pervasive spaces can be observed as abstract logical environments where devices provide simple or composite functionality that other devices (in the same or different environment) or users directly consume. In order to address the issues mentioned in previous paragraph, solutions that target building home automation systems should characterize the following features:

- Common/unified standards - standards which allow uniform connectivity and messaging between diverse/heterogeneous devices.

- Scalability – the ability of the system to easily expand the functionality by adding new or replacing old services (devices) with minimal effort.
- Devices with limited resources – supporting devices with low memory, limited processing power and power consumption. System should be able to minimize resource consumption managing by managing and redistributing tasks when necessary.
- Mobile devices.
- Device (Service) abstraction – internal functionality of a device (service) is transparent and invisible to the rest of devices (services) interacting with it.
- Information monitoring and security – information should be accessible and in the same time secured. Monitoring of information is essential to ensure faultless operation.

Some of the requirements for Ubiquitous spaces, such as remote communication, remote information access, high availability, fault tolerance and distributed security have been addressed in the Distributed computing domain. Likewise, features such as mobile networking, power consumption control, adaptive applications and location aware sensitivity have been adopted in the Mobile computing domain (Satyanarayanan 2001). In summary, Ubiquitous Computing requires the support of scalability, heterogeneity, integration, invisibility, while in the same time favours devices with limited resources and high mobility (Weiser 2002).

5.3 Dynamic Open Home Automation (DOHA)

Service-oriented Platform

As discussed in the previous section, ubiquitous computing concept involves systems that are both embedded and mobile. High mobility and embeddedness allow such systems and its services to be demanded at any time from everywhere (see 5.2 Ubiquitous Computing). These systems are composed of devices and technologies. Essential blocks of UC includes various computing devices such as mobile phones, tables, microcontrollers, sensors, embedded computers and technologies such as Internet, middleware, operating systems and location and positioning services. An emerging

system that follows the ubiquitous paradigm is Dynamic Open Home Automation (DOHA). DOHA is a SOA-based middleware platform for the access, control and management of home automated systems which are composed of a set of lightweight and independent services as shown on Figure 5-2. Essentially, DOHA enables development and communication of decentralized and independent applications, where new services can be added without knowing the implementation or operations of the rest of services already running on the underlying platform. Being SOA based, DOHA embodies all SOA principles such as interoperability, loosely-coupling, service autonomy and abstraction, reusability, discoverability, etc. The process of investigation of DOHA platform revealed that it is built upon the Device Profile for Web Services (DPWS) technology (also mentioned earlier). DPWS is a bundle stack offers libraries and protocols for building home automation solutions while providing the advantages of the web services. Figure 5-2 presents the overall situation in DOHA. As it can be seen, there are three main layers that builds DOHA – physical (this includes actual devices – sensors and actuators), virtual (this includes the machine code implementing functionalities of sensors and actuators), and service (and abstract level that describes functionalities on a high level of sensor and actuators but combines functionalities with other nonphysical services such as web services). As mentioned DPWS leverages the use of web services in conjunction with various devices. Next section elaborates about technologies related with DOHA and DPWS.

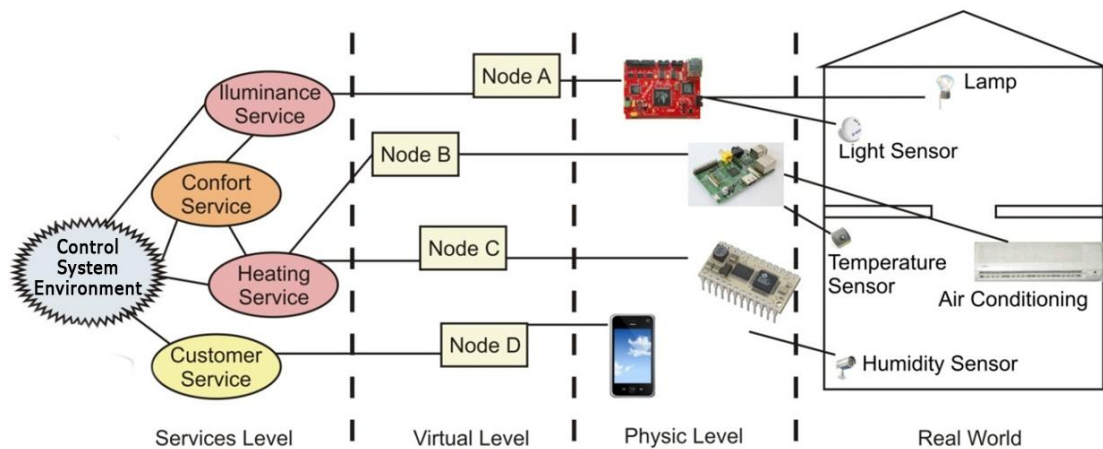


Figure 5-2: Dynamic Open Home Automation

5.3.1 Device Profile for Web Services

As it was mentioned in previous section, Dynamic Open Home Automation platform is built upon DPWS (Device Profile for Web Services) stack. Main advantage of this technology bundle is that it provides a solid foundation for building interoperable and flexible home automation solutions. Another benefit of adopting the DPWS set is its suitability for realizing a P2P-like infrastructures for smart home systems (Parra, Hossain, and Uribarren 2009).

The advantages of P2P-like infrastructure before the widespread client-server, centralized infrastructure for realizing smart home automation systems are several: First and foremost it favours distribution of application workflow logic among different peers. Peers are any nodes or networked devices that implements (runs) one or more pieces of the DPWS stack. Secondly, implementing P2P simplifies the application workflow at a peer level, meaning that peer will be responsible to perform its own task, operating independently and asynchronously from the rest of the peers. Last but not least, having a true P2P-compatible architecture enables the addition or removal of services without affecting the overall operations. Other reasons why peer-to-peer-like infrastructure is more suitable for Home Automation is given in section 2.2.2 SOA Design Patterns.

Ubiquitous environment are active spaces where new devices can be integrated without requiring complicated installation or multifaceted update process (preferably, full plug & play solutions.) As it was discussed, DPWS and P2P-like infrastructure work together to fulfil latter requirement. Despite the fact DPWS and its standards have been comprehensively explained in (Jammes, Mensch, and Smit 2005), the next few paragraphs will outline an overview of the DPWS stack and some of its key functionalities. Figure 5-3 depicts fundamental elements of the DPWS block.

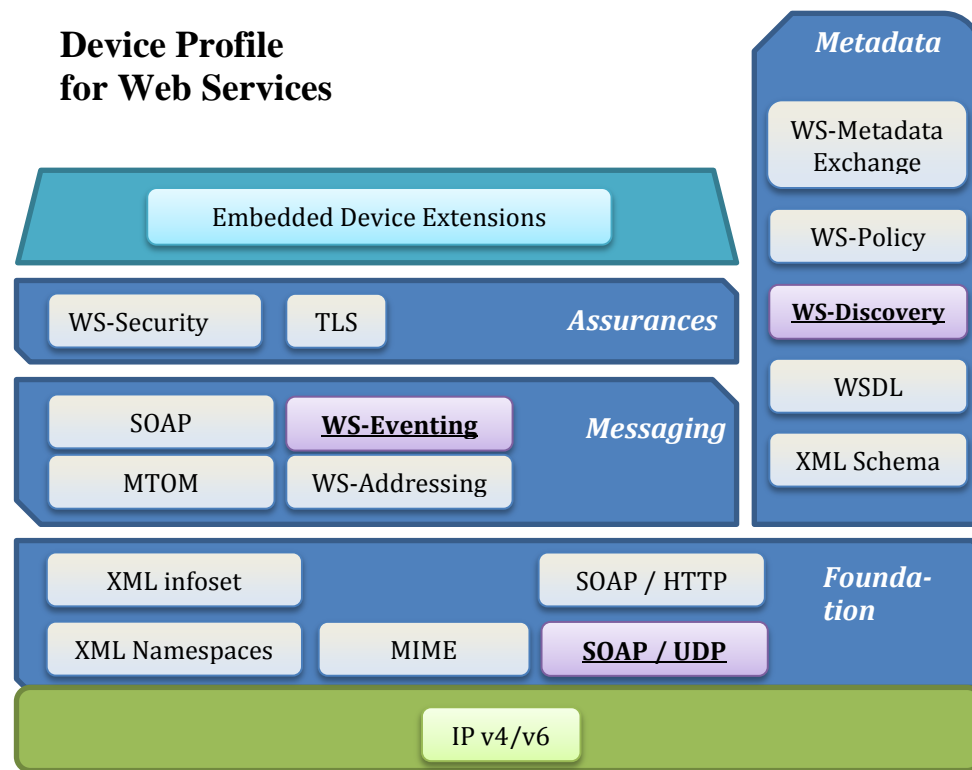


Figure 5-3: Device Profile for Web Services Stack.

These fundamental elements are grouped in to four layers: foundation, messaging, assurance and metadata layers. At the foundation, the Internet Protocol (IPv4/v6) provides low-level communication channel for the following standards: XML infoset, XML Namespaces, MIME, SOAP over HTTP, and SOAP over UDP. The last of these standards, SOAP over UDP, is vital in Home Automated systems since it provides robustness and high connectivity before the SOAP over HTTP standard. It is important one to note that DPWS stack supports both – SOAP over HTTP and SOAP over UDP, however, DOHA system uses only the SOAP over UDP to communicate services.

The role of **the messaging layer** is to define and address the messages in the service environment. At this level DPWS stack includes protocols such as Message Transmission Optimization Mechanism (MTOM), SOAP, WS-Eventing and WS-Addressing. Definition of the Simple Object Access Protocol (SOAP) and its purpos-

es were already discussed in **2.2** – “Fundamental Concepts of Service Oriented Architectures”.

At this layer, of a critical importance is the WS-Eventing protocol. Web Service protocol provides an instrument that defines the way of how clients (Subscribers) are registered to events (Subscriptions) of a Web Services (Event Sources). An important feature of this protocol is that it allows the delivery of events to be executed using simple asynchronous messaging. In a classical client-server architecture, request originates from the client and are addressed to the server (i.e. invocation of a service method is performed by an initiation request message sent by the client). The previous statement implies that Servers are typically passive, meaning that they will not respond or send any messages until they are asked by the client. On the other hand, the WS-Evening mechanism allows servers to send messages to already subscribed clients without any request, and this way become active (e.g. they become active peers). When a change in the server occurs, the server initiates a new communication by sending a message to the clients (Subscribers). To improve robustness, a subscription may be leased by an event source to a subscriber, and the subscription expires over time. The subscription manager provides the ability for the subscriber to renew or cancel the subscription before it expires (W3C, 2006). Table 5-1 summarizes steps involved in the subscription and notification process.

The Metadata layer of the DPWS block encapsulates and standards and protocols concerning the physical and logical representation of the services within the smart home environment. Implementation sets that carry out such functionalities are the WS-Policy, WSDL and the XML. Some of those standards had been already discussed in **2.2**. Other important technologies of the Metadata layer that provide mechanisms for Discovery and metadata exchange are the WS-Metadata Exchange and WS-Discovery standards.

Table 5-1: System messages defined in WS-Eventing protocol

Message Type	Sender	Description
Subscribe	Subscriber	Send message to the Event Source to create a Subscription
SubscriptionResponse	Event Source	Reply to a Subscribe message sent if the subscription is accepted, stating the expiration date and time of the subscription
Renew	Subscriber	Sent to an Event Source in order to update the expiration time of a subscription
RenewResponse	Event Source	Reply to a Renew message sent if the subscription renewal s accepted, with the new expiration date and time
GetStatus	Subscriber	Sent to Event Source to request the expiration time of subscription
GetStatus Response	Event source	Reply to a GetStatus message, with the current expiration time of the subscription
Unsubscribe	Subscriber	Sent to an Event Source with the urpose of cancelling a subscripton and stop receiving notifications
Unsubscribe Response	Event Source	Reply to a Unsubscribe message, confirming the subscription cancellation
SubscriptonEnd	Event Source	Sent by an event source when it no longer can send notifications to subscribers
Notification	Event Source	Message sent to subscribers with event data

An important protocol that enables the Ad-Hoc service discovery feature in DOHA is the WS-Discovery. This specification defines a discovery protocol to locate services.

The most common scenario for discovery is a client searching for one or more target services (OASIS, 2009.). Clients are the endpoints that search for specific services known as Target Services. Clients send a request using multicast (public) messages to all devices in the network. If any Target Services satisfy the request, they reply back directly to the client through unicast (personal) messages. Clients can search for Target Services using criteria such as “service name”, “service type” or “service scope”. Unlike the traditional Web Service repository (UDDI), the WS-Discovery protocol does not require a central service directory. To prevent scalability problems (e.g. large number of endpoints), the protocol supports multicast suppression behaviour. This protocol is not designed to support internet-scale discovery but is suitable for home automation, where the number of devices is not intended to be extremely high, and therefore, scalability is not a real issue. Table 5-2 gives A description of simple message types supported by the WS-Discovery protocol.

Table 5-2: System messages defined in WS-Discovery protocol

Message Type	Sender	Description
Probe	Client	Multicast message sent to search Target Services by type or within a scope
ProbeMatch	Target Service	Unicast response to the sending client when the Target Service matches the Probe message.
Resolve	Client	Multicast message sent to search Target Services by name
ResolveMatch	Target Service	Unicast response to the sending client when the Target Services matches the Resolved message
Hello	Target Service	Multicast message sent when service is joining the network containing descriptive information
Bye	Target Service	Multicast Message sent when leaving the network

The Sequence diagram on Figure 5-4 presents a typical case scenario of printing a document by a printer device, offering printing service. The diagram shows a Client that is probing (multicast searching) for a printing service on the network (1).

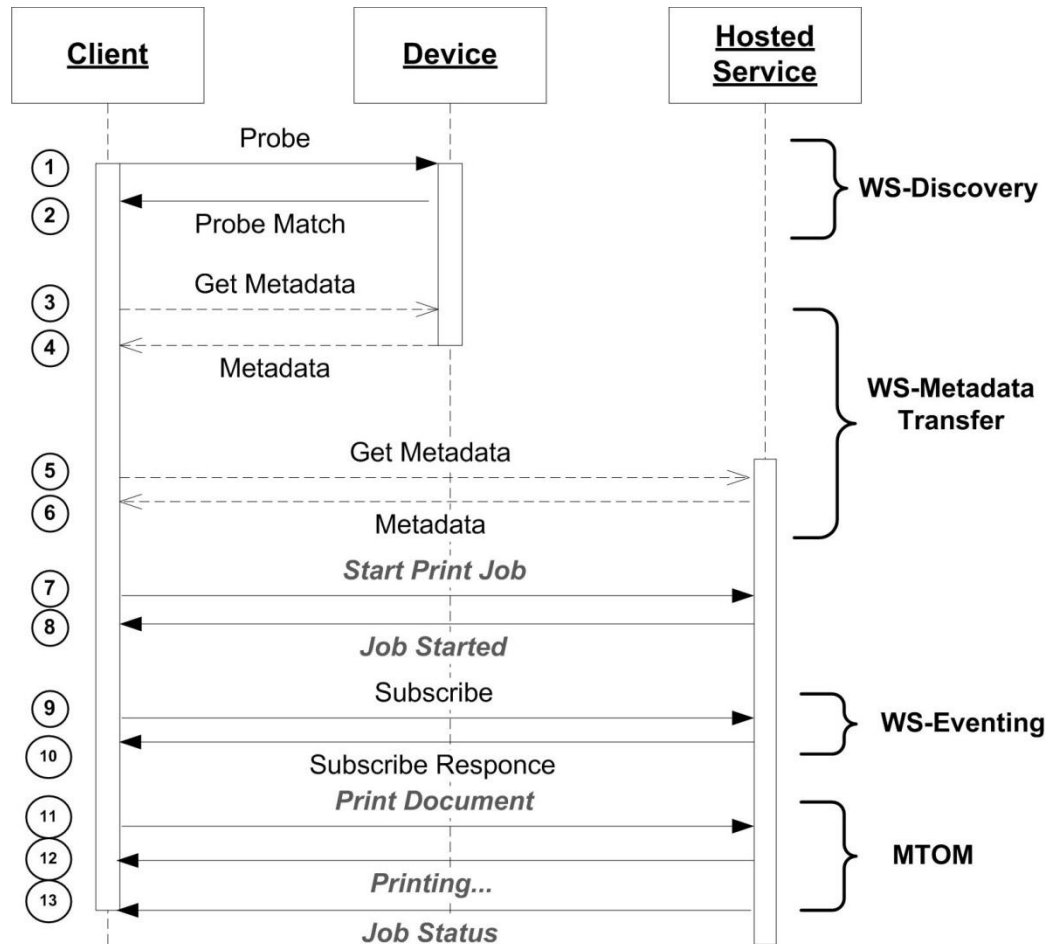


Figure 5-4: Sequence Diagram of a ‘printing’ Process Using the DPWS Protocols

If criteria are met by any service/device, an acknowledgment unicast (private) message is sent to the client (2). This sub-process is facilitated by the WS-Discovery Protocol. After a service is found, process of exchanging metadata (3 – 6) starts. The protocol enabling these activities is the WS-Metadata Transfer standard. An important point in this example is that physical Device and hosted services are separated - meaning that metadata for both device and services must be exchanged prior any other activity. After any necessary initial data is exchanged, the actual business pro-

cess (the one requested by the Client) 7,8,11-13 begins. The commands in this case are sent using binary SOAP's Message Transmission Optimization Mechanism (MTOM). As it can be observed from the sequence diagram, client communicates with the hosted Service directly, hiding (making transparent) the activities of the physical device. Activities number 9 and 10, represent some of the fundamental aspects the home automating systems; a client subscribing for a service. The subscription process is asynchronous meaning that subscription may occur at any time. The subscription mechanism is implemented by the WS-Eventing protocol, a short overview of which was given above.

5.3.2 DOHA Service

A service in the context of DOHA is an autonomous self-contained component capable of performing specific activities or functions independently. It can accept one or more requests and returns one or more responses through a well-defined, standard interface. There are two special types of services in DOHA: the Device Service and the Virtual Service. The Device Service can interact with other physical devices of the environment and it provides physical device control. The Virtual Service does not provide direct access to the physical services; it rather provides a logical representation of the physical object. Virtual services could also be end users applications, such as mobile applications, graphical user interfaces, etc.

Each service of DOHA is characterized by a multilayer architecture (see Figure 5-5). The multilayer structure consists of several design layers and it decouples the service into components. This facilitates the implementation and deployment of services, allowing components to control the state of the service when a request is accepted, to interact with other services, etc. The **Interface Layer** guarantees the access to services from any other element of the system. The **Application Layer** abstracts the functionality of a service. Finally, the **Interaction Layer** contains the settings needed to allow communication and collaboration with other services.

The deployment, start-up, and execution of the service require knowing additional information that is managed by the service during its life-cycle at runtime. This in-

formation is enclosed into three descriptive documents which form the base of the service specification. These are the **Service Contract**, the **Service Configuration** and the **Service Composition Map**. Each of them abstracts a fundamental aspect of the service within the platform and contextualizes its collaborative behaviour with other services. An overview is presented on Figure 5-5.

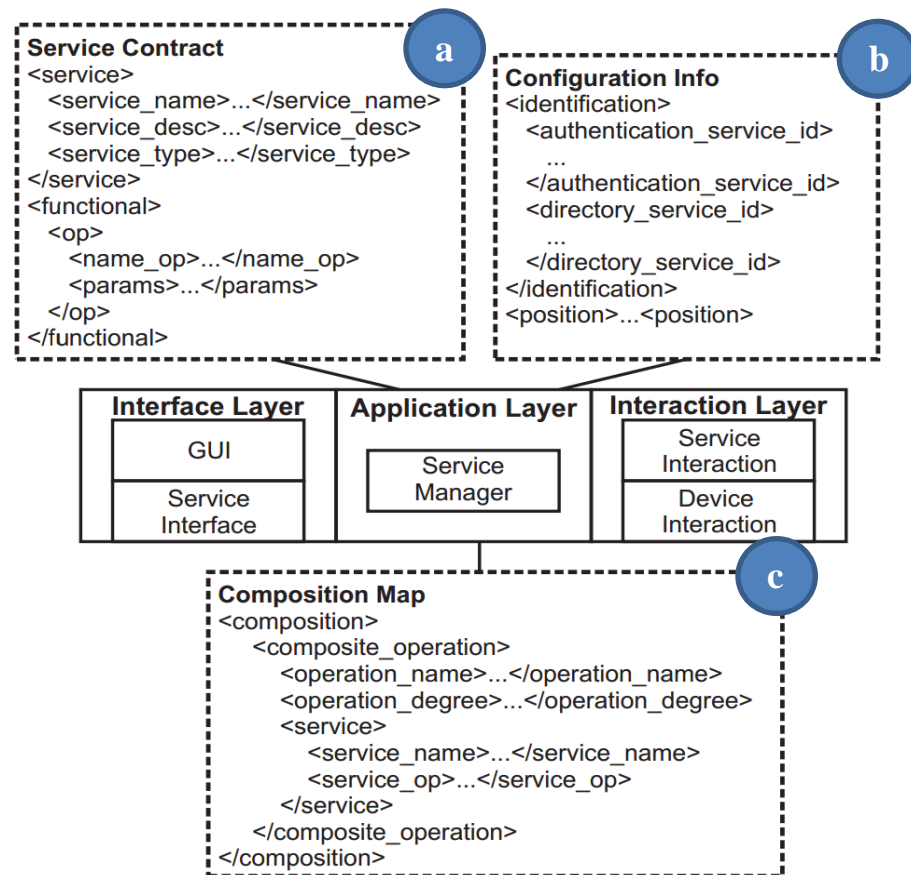


Figure 5-5: Structure of a DOHA service

The **Service Contract** (Figure 5-5 - a) is a fundamental part of any service composition. It represents an agreement between at least two parties. Usually, one of the parties is called ‘provider’ and the other – ‘consumer’. The provider is the service that provides some functionality, and the consumer is the service that takes benefit of this functionality. In DOHA, the Service Contract is a public exchangeable resource between services, containing a description of the requirements, restrictions, and functionality of a specific service and it is realized through WSDL standard.

The **Service Configuration** (Figure 5-5 - **b**) specifies the parameters required to execute the service in the proper manner. Configuration files contain information about the physical location of the node where the service is running, the type of connectivity, the service lifetime, etc. Additionally, these files include an identification of the service' instance that can be queried by the service consumers. Consequently, different instances of the same service could have different service's configuration. Basically, the configuration files contain information that is needed for a service to be initiated before communicating with the rest. In some cases, the service configuration can consist of non-typed information such as an image or a *.pdf document that can be downloaded through a URL.

The **Composition Map** (Figure 5-5 – **c**) is probably the most important part of the service anatomy. It defines the links of services involved in the composite operation. Each service has its own composition map. This feature promotes decentralization of the nodes and levers the autonomy of the composition. Further details are given in following section 5.3.2.1

All of the service specification documents, i.e. contract, composition map and configuration are stored in XML (Extensible Markup Language) files according to an XSD (XML Schema Definition) schema. Services involved into composition first acquire the metadata from XSD in order to establish common “language”. Specifically, in DOHA case, the service contract is described by the WSDL language included in the DPWS stack.

5.3.2.1 Service Composition Model

While examining the DOHA system, deliberate attention is paid to the composition model. This model represents the way services interact between each other; e.g. activities that particular service should perform to collaborate with other services in order to complete requested operation. My examination of the DOHA documentation revealed that the service composition model is based on the orchestration (and not choreography) principles (see Chapter 2 - 2.2.2). This means that execution flow of messages and transactions required in the collaboration control is always a responsi-

bility to at least one of the parties involved. Each service involved in the orchestration performs either simple or composite operation. A simple operation is a single transaction that service can perform on its own, i.e. the service has all resources necessary to carry it out the operation, and it does not require any additional interactions with other services. On the other hand, a composite operation includes invocation of at least one or more operations from one or more services. Service that executes a composite operation must interact with all necessary services included in its composition model. A service that only has simple operations is called “basic service”, whereas a service that implements composite operation(s) – “composite service”. The composite operations are the foundation of the collaboration model and are listed in Service Composition Map. Figure 5-6 represents an example of a simple Service Composition Map.

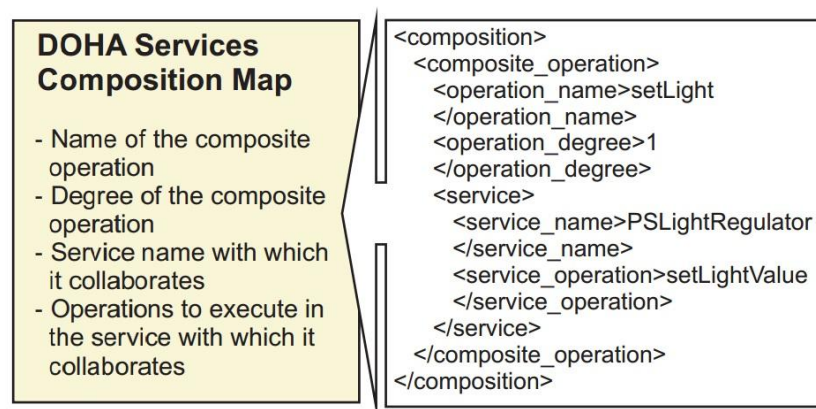


Figure 5-6: Structure of Service Composition Map (Holgado-Terriza and Rodr'iguez-Valenzuela 2011)

The XML code above describes the structure of a DOHA Service Composition Map. It lists services (using meta-data enclosed into <service name> tags) and corresponding operations required for the execution of a composite operation available on the service. From this example, we also can deduct that composite service is only aware of the services directly involved in the composite operation; and not of the full set of operation/services involved in the entire composition model. In other words, consumer service is not aware of how his provider executes its operations. This leverages the sustainability and autonomy of the service.

Further on, DOHA documentation discloses that service composition is modelled through graph theory (Bondy and Murty 1976). The composition map of a service can be seen as a graph. It contains the number services involved into composite operations, as well as their relations and roles in the process. Each composite operation op of a service S can be defined as a directed graph. Following definition (1) elaborates further:

$$\mathbf{G}_{opS} = (\mathbf{o}_{ops}, \mathbf{V}(G), \mathbf{L}(G), \mathbf{E}(G)) \quad (1) \text{ where}$$

- \mathbf{o}_{ops} is the main vertex of the graph and corresponds to the origin service S of the composite operation ops .
- $\mathbf{V}(G)$ is the set of vertices of the graph where each vertex represents a service which operation is invoked from the composite operation ops .
- $\mathbf{L}(G)$ is the set of labels where each label embodies a requested operation in a required service $\mathbf{V}(G)$.
- $\mathbf{E}(G)$ is the set of edges related with two vertices where the origin vertex is \mathbf{o}_{ops} , the destination vertex is an element of $\mathbf{V}(G)$ and the label of the line is an element of $\mathbf{L}(G)$

The composite graph is directed because the arcs (operations) between vertices (services) always have a sender and a receiver. The former is the service that invokes (sends requests), and the latter is the requested service (the provider of the operation). The graph of a given composite service may contain calls to several other services that are also composite. Hence, in order to perform required composite operation, all sub (composite) operations need to be executed first. This means that all operations are performed in a sequential fashion, and not nested one.

Additional investigation about composition model of DOHA showed that composition graphs denote *complexity degrees* values. These parameters define the degree of complexity of composite operations; e.g. the depth of the graph. A simple operation has degree of complexity **zero** - $complexDeg(op) = 0$. The degree of complexity of a composite operation \mathbf{G}_{ops} is specified by Definition 2:

$$\text{complexDeg}(G_{ops}) = \max(\text{complexDeg}(op_i)) + 1 \quad \forall op_i \in L(G) \quad (2)$$

Above stated definition is interpreted in the following way: the degree of complexity of a service G_{ops} is the maximum degree of all composite operations of its composition map plus one. The idea of the complexity degree values for each composite service is to ensure proper execution and to prevent cyclicity. Cyclicity is a state where a composite service can invoke itself. To ensure acyclicity, all composite operations must finish with a simple operation; or a composite operation can only invoke another composite operation with lower complexity degree.

The degree of complexity of operation provides useful information about the collaboration capabilities of each service. It also helps to determine the time spent for the execution of service's operation. Composite operations with a higher degree of complexity will take longer to execute because they must add the execution times of the requested operations to their own execution time. A graphical example of orchestration graph with few composite services and different complexity degrees is represented on Figure 5-7.

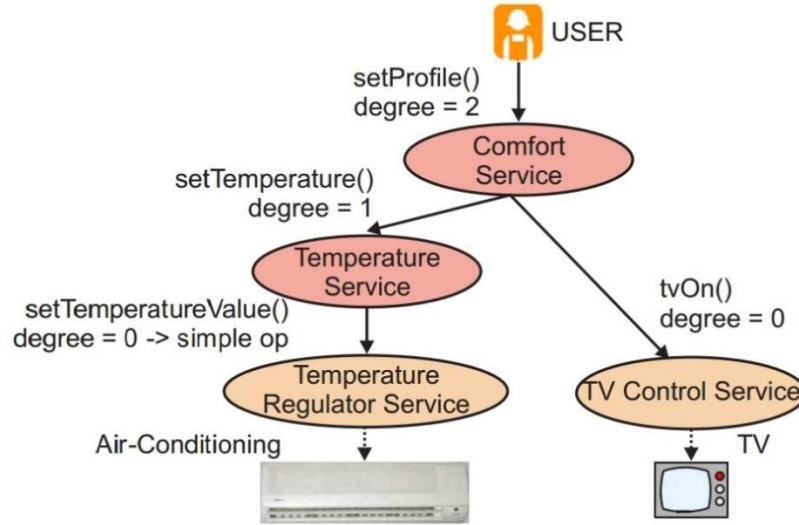


Figure 5-7: Service-orchestration presented by Directional Graph

5.4 DOHA DQ Service

This section provides discussion of how DQ framework was incorporated in DOHA. This process requires collaboration with DOHA stakeholders. After stakeholders were identified, a strategy must be created to engage them in all stages of the developing and evaluation process. Ideally, this engagement takes place from the beginning of the project or program or, at least, the beginning of the development cycle. The stakeholders should also know that they are an important part of the evaluation of the DQ process (e.g. validation) and will be consulted on an on-going basis throughout its development and implementation. The relationship between the stakeholders and the evaluators should involve two-way communication, and stakeholders should be comfortable initiating ideas and suggestions. Evaluation during program implementation could be used to inform mid-course corrections to program implementation (formative evaluation) or to shed light on implementation processes (process evaluation). This research supports both programs.

In chapter 3.3.1, Action Research was defined as an approach that “combines theory and practice (and researchers and practitioners) through change and reflection in an immediate problematic situation within a mutually acceptable ethical framework. In this study, I have considered University of Granada (UGr) as an academic partner who conduct practical research in SOA (DOHA). The fact that academia was chosen to partnership promotes some advantages before particular company. First, the research is more open and less biased, as described be AR communities, and second issues related intellectual property are easier to resolve between two academic institutions. Thirdly, the partnership with the University of Granada presented a great opportunity to incorporate developed DQ process within their service environment.

Through collaboration, this research study benefits in a few ways. Firstly, working together with UGr, allowed me to revise and further refine my DQ process. More specifically, this was done through organizing series of in-depth discussions which focused on the application context of the DQ process and its benefits for DOHA study. During these meetings, partners from UGr also presented their SOA research

(also as described earlier in this chapter) and defined their research objectives. Their prime objective was to improve DOHA system by making it pro-active, meaning that it must become intelligent – e.g. to be able to make decisions automatically based on certain conditions and events. To make the system more reliable, it has to support a mechanism that ensures the quality of the data – e.g. to secure the data running between services is as flawless as possible and complies requirements (statements) for data quality (see Figure 5-8). At the time of conducting this study, DOHA system did not feature any DQ tool, neither it supported any mechanism for governing user data/requirements. At the same time, this offered a great opportunity for my approach to be tested and validated.

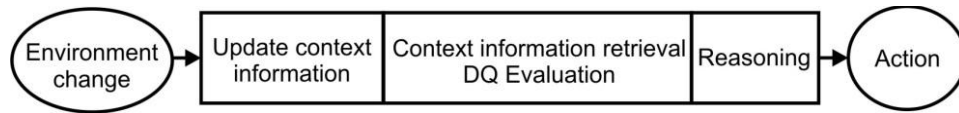


Figure 5-8: DOHA Proactive Behaviour Concept

As a result of all meetings, a decision was taken to develop a data quality measuring tool which will be the foundation of the intelligent DOHA system.

To develop and test a DQ assessing tool, several factors must be taken under consideration – i.e. “How to incorporate DQ mechanism into DOHA middleware?” and “How to test the functionality of developed tool?” Section 4.7 presented a method for incorporating data quality feature within service oriented context based on reverse engineering, and more particularly on design recovery and extraction. The goal is to obtain as detailed information about SOA middleware, standards and protocols as possible which will be later utilized when constructing the tool. On the other hand, to test such middleware add-on/service, domain specific case must be studied / developed. The case has to embody sets of services that models/manages data and executes particular (business) processes in a specific domain. Reflecting the method described in Section 4.7, Figure Figure 5-9 presents a detailed description of the process that answers aforementioned two questions. The process is built using BPMN specification to comply the consistency for process representation and improved readability of this document. The process was delivered after conducting an extensive workshop

with DOHA developers and architects. The main objectives were to conceive execution plan for developing and evaluating DQ tool within DOHA middleware as well as to refine and provide feedback about the DQ framework introduced in this research project.

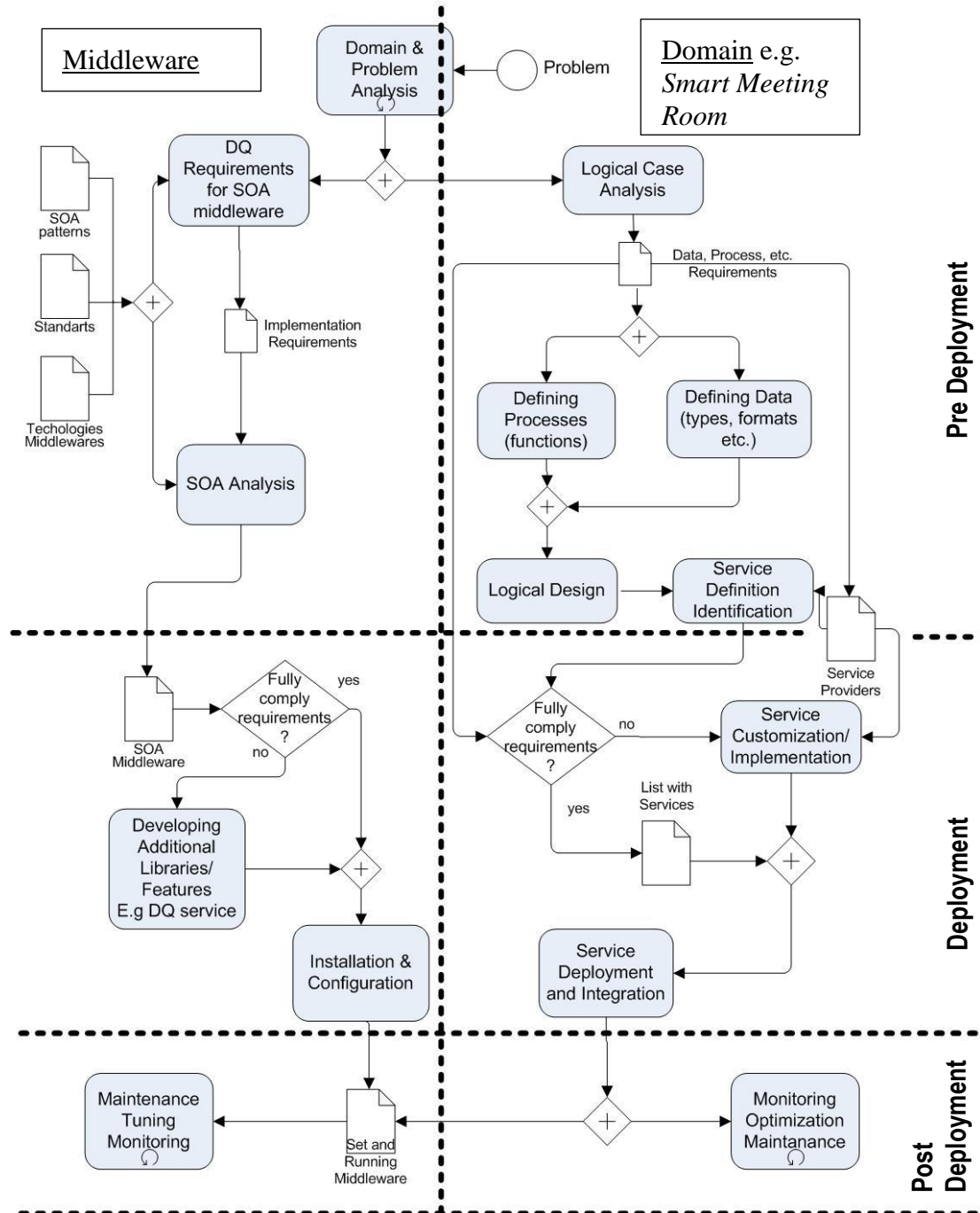


Figure 5-9: DQ Tool and SOA Domain Developing Plan

The process presented in Figure 5-9 was inspired by the SOA lifecycle diagram as shown in Figure 4-23. As seen, process scopes two areas and their development - the DOHA middleware and the application domain. Prior incorporation any DQ strategy in DOHA, several aspects has to be clarified – e.g. what are the specific DQ requirements for the DOHA middleware. Conducted workshop at UGr aided this process by aligning their objective with the objective of this research study – e.g. to validate DQ process and to increase awareness of the DOHA system. Based on the presented by researcher DQ process, the following system DQ specific requirements were defined in conjunction with UGr partners:

1. It must be able describe the environment
2. It must be able to save user preferences
3. Action mechanism based on user preferences
4. Describes the service/data dependencies

Based on these requirements an objective was set to develop a model e.g. class diagrams of DQ/reasoning tool as well as to determine what technologies and data structures are going to be used.

The output of the workshop is presented in Figure 5-10 in a form of a class diagram. The picture presents two main packages of the system – the DOHA package and *SemanticLayer* package. Note that for brevity, only some parts of the DOHA classes are presented e.g. the main *Sevice*, *Operation* and *Param* classes. The highlight, however, is on the *SemanticLayer* package. It consists of *Context*, *ContextElements*, *Profile*, *Constraint*, *Action*, *JenaReasoner*, *Ontology*, and the linking *SemanticManager* class. Firstly, *Context* class is used to describe the elements of the environments. It is based on the predefined *ontology* (e.g. see Appendix C) fetched via *JenaReasoner*. Jena Reasoner is an add-on to the Apache Jena java framework. It presents set of interfaces to support the use of different OWL based ontology languages (Oracle 2015).

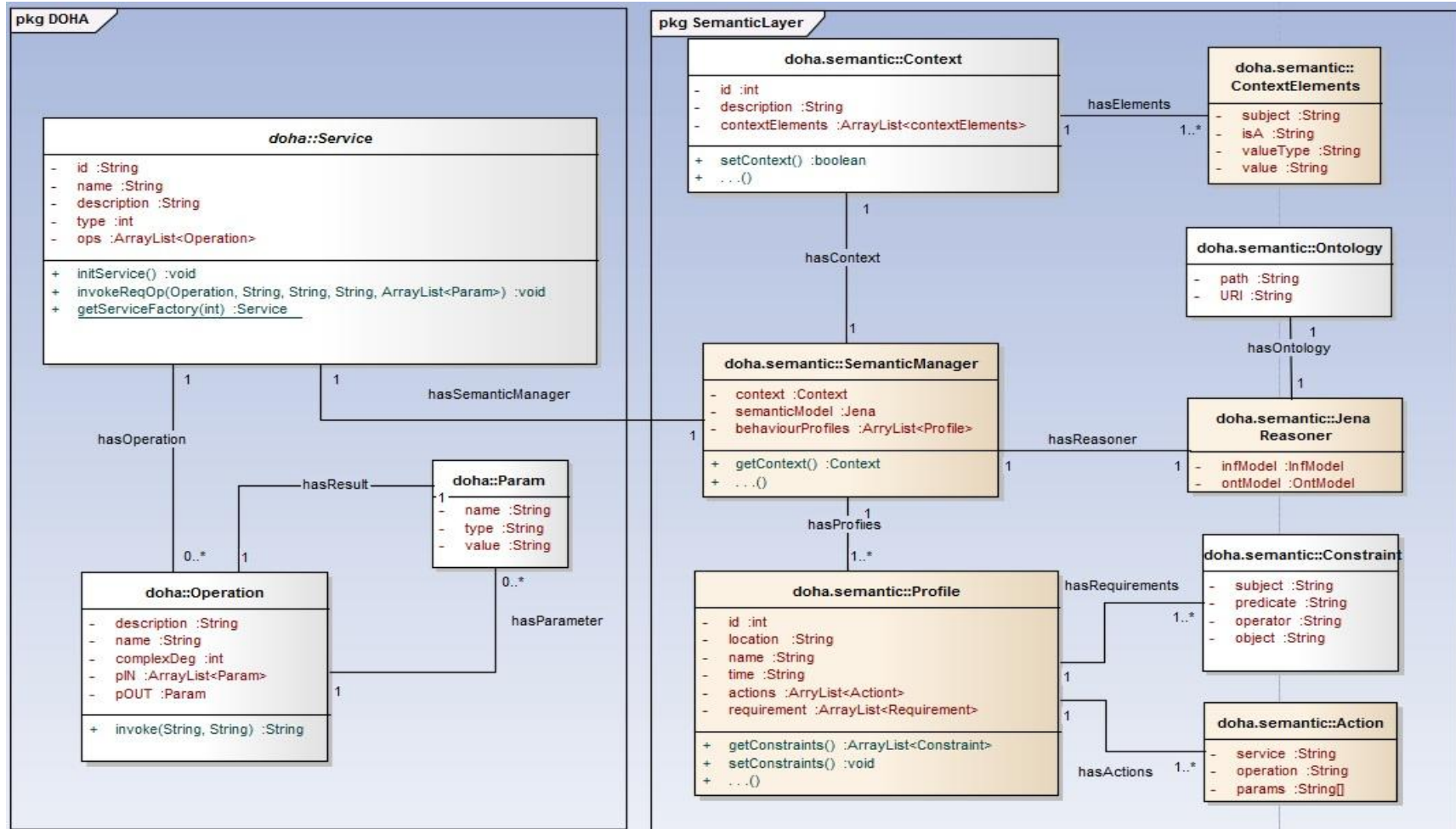


Figure 5-10: DOHA Semantic Layer

During the workshop, CoDAMoS (Preuveneers and Bergh 2004) was selected as main ontology since it offers a comprehensive description of smart spaces. As seen in the figure, Context can contain custom elements to describe the base ontology file. For example, if ontology contains the elements humidity temperature, etc., the *ContextElements* class puts these elements into context e.g. humidity *is relative*, the temperature *is in Degree Celsius*, the temperature *is in Degree Fahrenheit* etc.

The *Profile* class relates together two classes - the *Constraint* and the *Action* one. The *constraint* class implements the statements about data in the system. The model used to build this class is based on the one that was proposed in 4.6.2. As outlined in the same section, users are allowed to compose lists DQ statements. As seen in the class diagram, the link between *Profile* and *Constrain* is one-to-many. The *Action* class comprises information needed to execute specific operation after constraint is met. As in the case of the *Constraint* class, one profile can have multiple actions. Finally, the *Semantic Manager* class links all classes of the package

In summary, classes *Context*, *ContextElements*, *Ontology*, *JenaReasoner* fulfills requirement one; classes *Profile* and *Constraint* – requirement two; classes *Profile* and *Action* – three; and *SemanticManager* – requirement number four.

Figure 5-11 presents the architectural layers of the DQ/semantic tool. The communication between the data users/administrators/animists of the DQ service is executed through series of web interfaces. Web technologies were chosen to build the presentation layer since they allow greater flexibility, technology independencies, application support, maintenance and adaptability is easier to make, and last but not least, DQ can be accessed from any device that supports web connectivity. In this case, the interface was build using the combination of HTML, Cascade Style Sheets (CSS) and Java Server Pages (JSP). JSP is a technology for controlling the content or appearance of Web pages through the use of servlets, small programs that are specified in the Web page and run on the Web server to modify the Web page before it is sent to the user who requested it. To store and execute JSP, Apache Tomcat web application container was used. The reason behind using the combination of the JSP + Apache

Tomcat was that they allow generation of dynamic web content and most importantly they are based on Java EE, meaning that all libraries written and in java can be used when building JSP pages. This is particularly important, since DOHA middleware, and all its services, including DQ/Semantic are built upon Java EE. Examples of user interfaces that use JSP technologies are presented in Chapter 6, i.e. Figure 6-4 and Figure 6-5. After user requests are translated by the JSP they are sent to the DQ / Semantic module, does several operations. First, it uses parsers e.g. Apache Jena parser to define the context of the inquiry and to fetch needed information from CoDAMoS ontology repository. Then, it calls another parser to read and translate user's defined data requirements (described in XML documents as suggested in section 4.6.2) to service executable queries. Next, the DQ module communicates with (invoke) different services using the DOHA infrastructure and technologies embedded into the DPWS stack described in section 5.3.1. The results of this process are saved into log repository. The DQ/Semantic manager is also in charge for parsing and passing already stored log data to the appropriate JSP page - in this case (*../reporting.jsp*) (see also Figure 6-6) which in turn generates and visualise reports based on the users' queries. JSP allows the generation multiple views depending on the context and requests, this is yet another reason why it was chosen as a presenting technology. The class diagram of the DQ/Semantic manager/layer was shown earlier in Figure 5-10. The reasoning function is not discussed since it is out of the scope of this research.

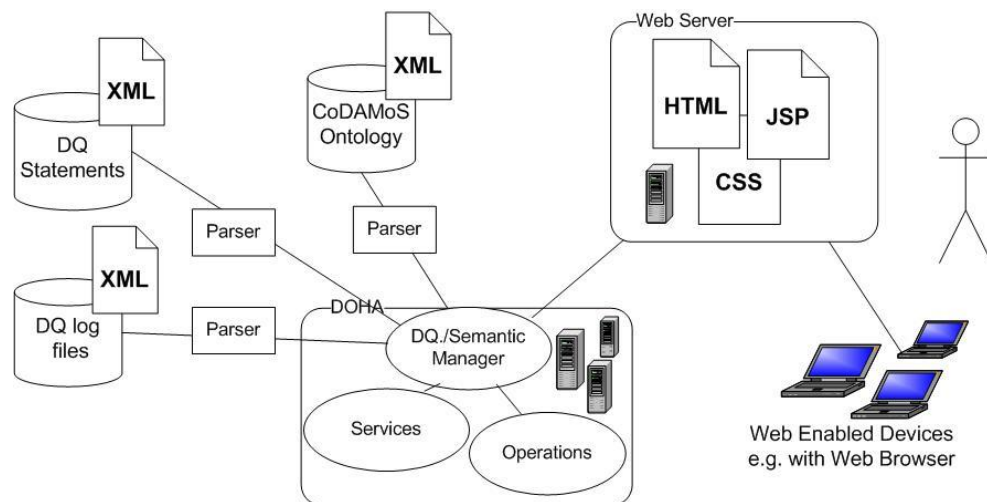


Figure 5-11: DQ Service Layers

After DQ tool was built, next step was to apply it within a real setting system. For this reason a specific domain has to be chosen - in this case, it was the Smart meeting room (introduced in section 4.3). This decision was made based on a few factors. First BIG group situated at Dublin City University had the necessity of automating the process of their research meetings as well as improving the comfort of the participants. And secondly, the environment involved a number of sensors, actuators and services governing them which predisposed to a great opportunity for spreading mis-handled data across the system. Based on these factors, the researcher developed a total number of eight services which were deployed on several Systems on a chip (SoC) and mini PC devices as it can be seen in Figure 4-3: Smart Room Service Composition of section 4.3. The services were built upon DOHA libraries and the DPWS stack as described in section 5.3.1. Each service contained a package of classes with different operations and its main (service initialising) function as seen in Figure 5-12. Partial code of a typical DOHA service is also listed in Appendix D. The case was developed following the process model presented in Figure 5-9 (right side).

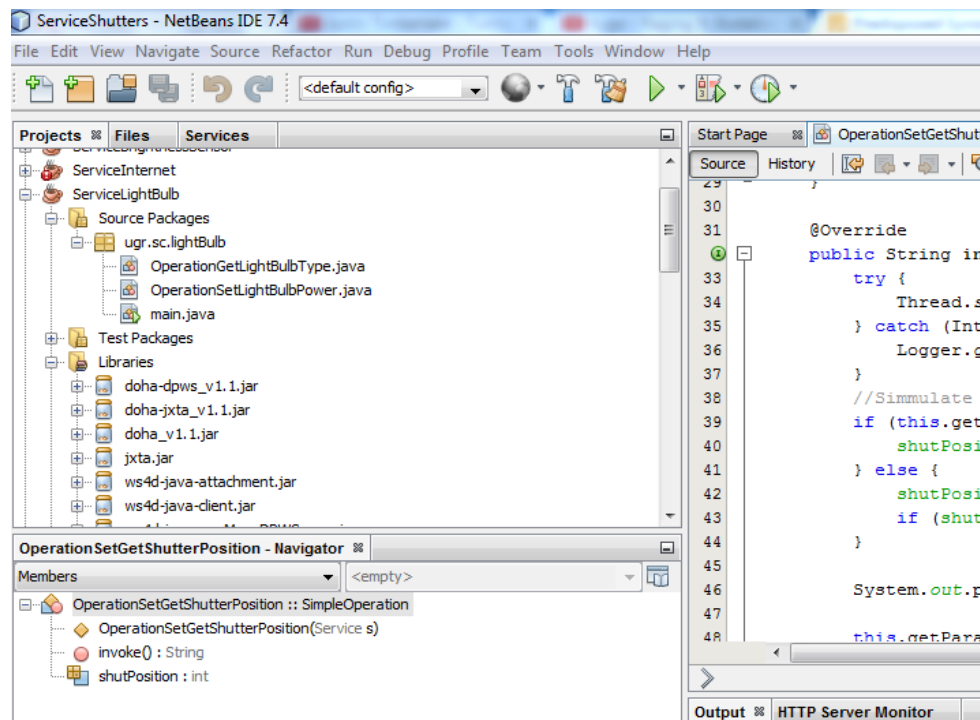


Figure 5-12: Service Development of Smart Meeting Room Case

5.5 Summary

This chapter aimed to complement answering the main research question “*How can be Data Quality measured in a Service-oriented Architectures context?*”- RQ2 and its sub questions RQ2.1, RQ2.2, RQ2.3 and RQ2.4, by instantiating DQ process for a particular case of SOA namely Dynamic Open Home Automation system. DOHA is an SOA-based middleware platform for access, control and management of home and office -automated systems which are composed of sets of lightweight and independent services. It leverages the communication of decentralized and independent applications, where new services can be added without knowing the implementation or operations of the rest of services already running on the underlying platform. DOHA was chosen as a suitable system as it leverages the main principle of SOA such as service autonomy, reusability, loosely coupling etc. DOHA was also favoured due to the fact that it did not support any DQ detecting mechanisms while in the same time it was open-sourced which allowed access to technical documentation and code. Later in this chapter, an in-depth discussion was provided about specific standards, approaches and designed principles that are embedded into DOHA. Moreover, collaboration with University of Granada allowed refining artifact in term of further detailing of DQ assessment process. Following the DSR collaborative model and the method for incorporating DQ process proposed in Chapter 4 – section 4.6, a prototypical software service that realizes and aids the process of capturing poor data (e.g. addressing and instantiating RQ2.1 and RQ2.2) within Home/Office automated environments was developed. This chapter concludes with a specific case of DOHA, namely BIG Smart meeting room. It was also described in section 4.3 - Motivational Case – Dynamic Open Home Automation (DOHA) Smart Meeting Room. Finally, Table 5-3 presents a summary of the research questions with references to the sections where a particular question was addressed along with contributions.

Table 5-3: Research Questions addressed in Chapter 5

Research Questions Addressed in This Chapter		Section Reference	Summary and Contributions
RQ 2	How can be Data Quality measured in a Service-oriented Architectures context?	Chapter 5, Sections 5.1-5.5	<ul style="list-style-type: none">- introduced specific SOA context- DOHA architecture, service, composition model, communication standards and protocols explained- incorporating DQ process within DOHA case
RQ 2.1 and 2.2	How to define and detect “poor” Data Quality within the context of SOA?	Section 5.4	<ul style="list-style-type: none">- developed DOHA Data Quality mechanism using DQ process proposed in section 4.6- developed a class diagram- explained overall architecture
RQ 2.3 and 2.4	How to design and implement Data Quality assessment method?	Section 5.4	<ul style="list-style-type: none">- presented method for designing and implementing data quality software tool and services for Smart Meeting room (also introduced in section 4.3)

Chapter 6 Artifact Evaluation

6.1 Introduction

In Chapter Four I have presented the design of DQ framework that consists of conceptual descriptions, DQ assessment artifact in form of BPMN process, and a method for incorporating the DQ artifact into SOA. Consequently, based on aforementioned method, in Chapter Five I have outlined the construction of the DQ software tool delivered in conjunction with DOHA research team. While developing the tool, an iterative approach was employed. This approach allowed me to further refine the DQ assessment artifact through frequent meetings and discussions with the partners of University of Granada. Moreover, a preliminary evaluation, in terms of concept evaluation of the DQ process was achieved through feedback collected from DOHA architects and developers. The development and incorporation of the DQ tool within home automation middleware indicated the applicability and feasibility of my DQ framework.

The purpose of this chapter is to evaluate utility of the instantiated artifact from two perspectives:

- The DQ assessment process.
- DQ software tool.

To do so, the researcher of this study organises a workshop in which he aims to demonstrate the DQ process, tool as well as collect feedback through survey, and short interviews/discussions. The rest of this chapter is organized as follows: First, an introduction of the evaluation protocol is presented. Then, in Section 6.3 evaluation criteria are examined. The following Section 6.4 considers different influential factors that might affect the evaluation results, followed by presentation of the design of the workshop – Section 6.5. Workshop includes home automation use case that is

first introduced and then executed by participants – Sections 6.6 and 6.7. At the end researcher collects their feedback and summarizes his findings considering evaluation criteria and different influential factors. This is carried out in Section. This chapter is then finalised with summary and conclusion.

6.2 Evaluation Process

Evaluation process is a crucial part of any research study. It is extensively defined as “a systematic, rigorous, and meticulous application of scientific methods to assess the design, implementation, improvement, or outcomes of a program. It is a resource-intensive process, frequently requiring resources, such as, evaluate expertise, labour, time, and a sizable budget” (Ross, Ellipse, and Freeman 2004). The primary purpose of the evaluation process is to "determine the quality of a program by formulating a judgment" (Hurteau, Houle, and Mongiat 2009). Researchers have identified two main types of evaluation according to its purpose: “formative” which provides input for improving a product or artifact; and “summative” – one that provide information of short-term effectiveness or long-term impact of a product or process to determine its adoption (Bloom 1971). Nonetheless literature does not agree on a single definition and purpose of the evaluation process. Moreover, evaluation methods, techniques and sequence of activities are strictly dependent to the particular research project and on its purpose and requirements. For this reason, an individual evaluation plan and process was developed to assess the artifact of this study, as shown on Figure 6-1. The purpose of this evaluation process is primary formative e.g. to provide information for improving artifact, but it also contains summative purposes such as to provide justification about the quality of the artifact and its prospect for further use or adoption.

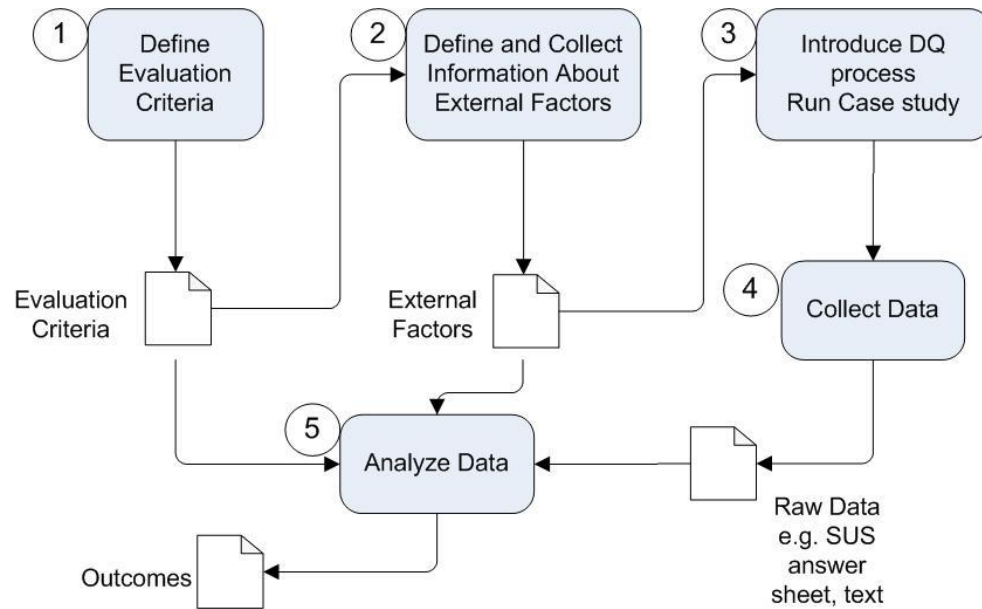


Figure 6-1: Evaluation Process

The first step (Figure 6-1-1) of the evaluation process is to define exactly what criteria would be used to determine the quality of the artifact under consideration. The identification and definition of the evaluation criteria is based on the main research question asked in this study – e.g. “how to assess and analyse data quality within Service-oriented context” and research output – a data quality process that is incorporated within specific SOA system. Selected criteria are described in the following section 6.3. The output of this stage is a list of well-defined criteria that are used to determine artifact’s value.

As mentioned in previous section 6.1, researcher have made decision to conduct a workshop in order to present and evaluate developed DQ artifact by collecting data in forms of survey and discussions. The rationale for this choice is based on the fact that workshops provide training sessions while at the same time they provide a great opportunity for collaboration and discussions among participants and researchers which can result in artifact evaluation. Audience of workshops is typically small – e.g. between 5 and 15 participants, however obtaining information about them could be crucial for the effectiveness of the outcomes. For this reason, during this step, factors such as participant’s background, expertise and prior knowledge in the area has to be

collected. Section 6.4 elaborates further about selected factors and how they would affect results.

The third activity of the evaluation process of this research study is to conduct the workshop (Figure 6-1 - 3). Normally, workshops are single, short (2-4 hours) educational events designed to teach or introduce to participants practical skills, techniques, or ideas which they can then use in their work or their daily lives. One objective of organising workshop is to familiarize participants with concepts of Data Quality. The second (and main) purpose is to obtain feedback about the DQ process and tool through survey and discussions. The process goes as follows: attendees are first thoroughly informed about various terms, concepts, and the contexts used in the study. Then they are invited to take part in case study prepared by the researcher (see section 4.3 for details about the case). During this step, they are assigned tasks (e.g. scenarios) to assess data quality. More detailed information about the workshop e.g. layout, purposes, participants, the sequence of activities, etc. is described in section 6.6.

At the end of workshop, attendants are invited to provide their comments and feedback against criteria identified in step one of evaluation process. This step (Figure 6-1 - 4) is referred to collection of participant's data. For evaluation purposes of this research study, a mix of qualitative and quantitative approaches (with focus on the qualitative one) has been used. Detailed discussion on motivation and reasoning for selecting this method is presented in section 3.4.3.1. During this stage, quantifiable data is collected through adoption of the System Usability Scale (SUS) survey tool. The SUS was selected as evaluator tool as it provides a mechanism for measuring the usability (Sauro 2011). It consists of a 10-item questionnaire with five response options for respondents; from *strongly agree* to *strongly disagree*. In addition, qualitative data is collected from participants that aim to provide information about the reasoning behind participants' choice when using the SUS tool. The collection of data is managed by the researcher of the study. The output of this stage is in form of raw data – e.g. quantifiable SUS answer sheets and open-ended responses in form of text about each of the SUS questions.

The final step of the evaluation process is the analysis of raw data and transforming it into sensible outcome. The goal of analysis is to extract meaningful information that aims to understand issue or particular situation by investigating the perspectives and behaviour of the people in these situations and the context within they act (Kaplan and Duchon 1988). Analysis also aims to understand not only *what* participants respond - e.g. about the “usefulness” of the DQ process, but also *why* – i.e. to understand how people think or feel about something and why they feel in this way (in this research case, why participants consider DQ process and tool useful; or perhaps why not?) To find out what participants felt about the DQ process and tool, the SUS tool is used. The analysis of the SUS sheets is based on the protocol described in “Measuring usability with system usability scale” (Sauro 2011) and results are presented in section 6.8. On the other hand, to observe why participants made their choice and how they feel about the presented artifact, open-ended questions (see Appendix B – open question) are asked. These questions target each of the items in the SUS questionnaire. Then answers were provided by participants in form of free text, which was assigned to the particular SUS question. Then the analysis of participant’s responses is conducted in the following order: Once participant’s notes are collected, researcher reviews each document line by line. The objective of this procedure is to extract any key notations (codes) which can be then referred to the criteria defined during step one of the evaluation process (see Figure 6-1-1). For example, if in a sentence there is a piece of information that relates with any of definitions of the criteria, this piece is coded (e.g. categorised). Then after all key notions are categorised they are joined into groups and interpret. The interpretation is performed by the researcher of this study, however, each participant’s answer is carefully analysed and genuinely interpret considering their positive and negative standpoints. Then responses that are similar were combined. During this process more emphasis is assigned to the statements that are repeated by more than one participant (there is an agreement or disagreement about a construct e.g. interface design). Another important point researcher have to do while interpreting results is to take all external factors (e.g. participants demographics, background, etc) under consideration. Finally, responses that do not relate with any of the listed criteria but are considered as important are also extracted

and presented as additional comments – e.g. comments for improvement of DQ software tool. The outcome of this step is presented in chapter 6.8 – “Results and Discussion”.

6.3 Evaluation Criteria

The output of this research study is a DQ assessment process, part of which was implemented as software tool. Following the design science “build-evaluate” cycle, it is essential that a rigorous evaluation of the artifact is provided. As part of this, definition of appropriate criteria that effectively evaluate the artifact is needed.

Design science research often relates the evaluation process of the designed artifact with two main concepts - fitness and utility (Gill and Hevner 2013). Fitness is defined by the DS community as the ability of organism (in this case DQ assessment process) to be reproduced — completely or in part — and evolve over successive generations. Fitness, however, is presumed to exist only in the abstract - it cannot be measured directly and its true value unfolds only over time (Gill and Hevner 2013). Analogously to fitness, the term utility is used in a number of ways. When considering the utility of a tool, DS researchers normally referring to its usefulness. As currently used in the context of DSR, Hevner, et al. state: —”The utility, quality, and efficacy of a design artifact must be rigorously demonstrated via well-executed evaluation methods” (Hevner et al. 2004). This implies that utility is a characteristic of the design and its intended application context. To distinguish between the two usages of the term utility, researcher often refer to the first as usefulness. Other factors included in the meaning of usefulness are efficacy in performing the task (including performance), range of task cases performed, ease of use, ease of learning, and cost-benefit in the performance of a task.

It is useful to consider what kinds of qualities for evaluation are discussed in the literature. Hevner et al. further state that “artifacts can be evaluated in terms of functionality, completeness, consistency, accuracy, performance, reliability, usability, fit with the organization, and other relevant quality attributes” (p. 85)(Hevner et al. 2004). Greiffenberg et al. points out that in order to appraise the applicability of the

method it must describe its conditions and intended scope of application. To do so, they recommend three criteria to evaluate methods: appropriateness, consistency, and completeness (Greiffenberg 2004).

All mentioned criteria are applicable for evaluation processes and can be useful when evaluating my DQ assessment process. However, in order to evaluate the utility of instantiated artifact – e.g. the DQ software tool, a criterion that measures the interaction between user and the process/system is needed. Such criterion is the User Experience. According to the International Standard Organization (ISO) with subdivision “ergonomics of human system interaction” - ISO 9241-210, User Experience (UX) is “a person’s perceptions and responses that result from the use or anticipated use of a product, system or service” (ISO 2015). UX includes three main defining characteristics:

- A user is involved.
- That user is interacting with a product, system or really anything with an interface.
- The users’ experience is of interest, and observable or measurable.

Some researchers distinguish between the terms usability and user experience. “*Usability is usually considered the ability of the user to use the thing to carry out a task successfully, whereas user experience takes a broader view, looking at the individual’s entire interaction with the thing, as well as the thoughts, feelings and perceptions that result from the interaction*” (p.5)(Albert and Tullis 2013). Literature further elaborates that almost any product or system that has some type of interface between the user and the system could be studied from the UX perspective (Carlsson and Henningsson 2011). Finally, User experience is defined as prime model in human–computer interaction which makes it very applicable in socio-technical studies - such as this one.

Taking into account discussion above, I have defined criteria that consider the evaluation of overall DQ process as well as the DQ software tool as presented on Figure 6-2.

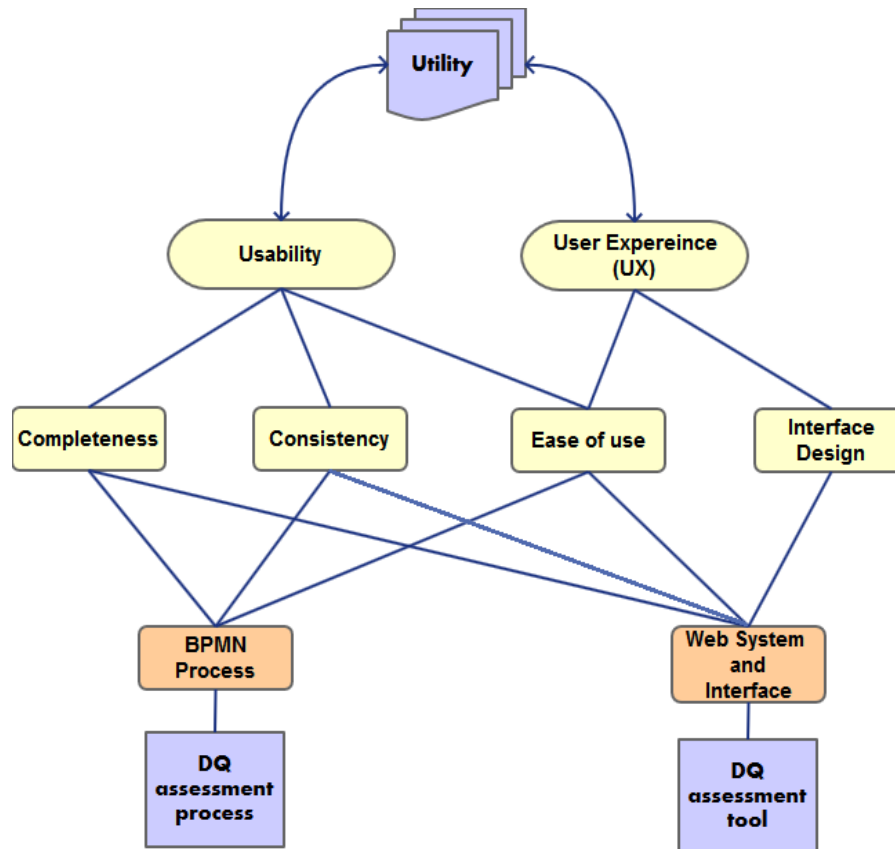


Figure 6-2: Evaluation Criteria

As it can be observed from the figure, both - *usability* and *user experience* criteria was used to determine the *utility* of the artifacts. More specifically, usability criterion was divided into three sub criteria – e.g. *Completeness*, *Consistency* and *Ease of use* that better identifies its meaning. *Usability* criterion is used to describe the utility of the overall DQ assessment process, and the *user experience* to determine the utility of the DQ software tool, since it involves interaction. Nonetheless, the *usability* and *user experience* overlaps in their characteristics. This is observable from Figure 6-2 as they are branching down to the same criteria. In general, it is impossible to specify the usability of a system (i.e., its fitness for purpose) without first defining who are the intended users of the system, the tasks those users will perform with it, and the characteristics of the physical, organizational and social environment in which it will be used.

Since usability may vary in different contexts, it follows that measures of usability must themselves be dependent on the way in which usability is defined for the particular case. Table 6-1 provides compiled description about selected criteria as well as mapping to each of the artifacts.

Table 6-1: Evaluation Criteria Description

Artifact Criteria	DQ assessment process	DQ software tool
Completeness	Whether the DQ assessment methodology comprises all necessary information and functionality it was designed for. E.g. to detect poor semantic data.	Whether the software tool have all necessary functionality to support DQ assessment process
Consistency	Whether all parts of the DQ process are mutually compatible and work without contradiction.	Whether or not DQ tool is consistent with the DQ process.
Ease of use	Time needed to understand the process.	Time needed to understand the how application work.
Interface Design	n/a	Whether or not interface elements are accessible and easy to find.

In the table above, a few remarks could be noted in relation with ease of use. The ease of use is tightly related with the level of understandability of the DQ process and how software tool works. Two factors are relevant in regard with that – 1) the *model factors* such as structure, number and complexity of the tasks, notations, etc. and 2) *personal factors*, such as training experience, skill, and personal viewpoint. Ease of use is also affected by the way information is presented. Also, when speaking of software tool, the ease of use is highly dependent on the Interface Design. For example, if there are too many elements or text user may find it hard to comprehend and consequently it would take him longer to complete a task.

As indicated above there are many factors that can influence the evaluation of my DQ process. To address that, variety of factors influencing user experience such as user's previous experience, system properties, and the usage context (situation) must be taken into account (Hassenzahl and Tractinsky 2006). During evaluation process, considerable amount of efforts were made in order to keep these factors stable. Next sections further elaborate to aforementioned statements.

6.4 Factors Affecting Evaluation Process

Performing an evaluation with relatively small number of subjects is a difficult task. This is due to the fact that there are certain factors that might influence subjects' feedback and hence evaluation scores. These factors, also known as 'exogenous', have critical impact on the quality of research evaluation (Kumar 1996). They are called in such way since they are not related with DQ process under investigation but rather deal with the environment and the people testing/evaluating DQ process. Literature defines many factors that may influence decision quality, such as information overload (Eppler and Mengis 2003), decision aids (Neumann and Hadass, 1980), decision models (Janis and Mann 1977), decision strategy (Payne et al. 1988), task complexity (Campbell 1988), expertise (Fisher et al 2003), decision time (Svenson et al. 1990), decision environment (Shankaranarayan 2003), interaction (Burleson et al. 1984), and information presentation (Remus 1984). Ge (2009) explored ten exogenous factors which he applied in his research evaluation phase. These are 'information presentation', 'decision strategy', 'expertise', 'time', 'task complexity', 'environment', 'decision models', 'decision aids', 'information overload' and 'interaction'(Ge 2009). In this study, only specific factors were considered while caring out the evaluation process. Selected exogenous' factors are presented in Figure 6-3. The rationale behind this particular selection was based on analysis of literature mentioned above by extracting only factors that relates to the nature of DQ artifact. For example, factors such as 'decision strategy' decision models' and 'decision aids' were not relevant for my research study. Hence, following discussion focuses only on selected factors for this work.

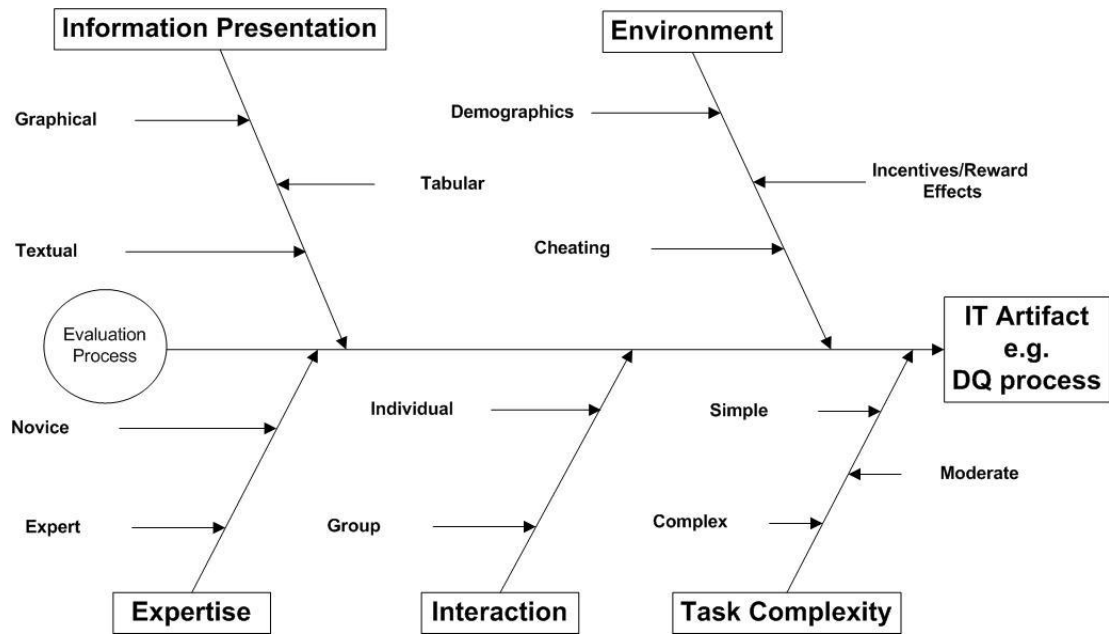


Figure 6-3: Exogenous Factors Influencing the Evaluation Process (adapted from Ge (2009))

One perspective influential factors can be seen during evaluation is that they can act as independent (or fixed) variables. In this research, an attempt to control the influencing factors and thus evaluate more effectively the process of assessing data quality was made. In following sections a review of some of the ways to control these influential factors is presented.

Task Complexity. The complexity of tasks can greatly affect the result of the evaluation; hence an effective method to set appropriate complexity for home automation use case is needed. Different kinds of tasks could achieve complexity through different paths. In problem tasks, complexity is a function of the number of potential paths to the desired outcome (Taylor 1981). Literature observes task complexity from two perspectives. On the one hand, task complexity could be examined by participant's experience, familiarity and interests of the area (Shaw 1971). On the other hand, task complexity could be examined by task alternatives, task constraints, and information interrelationships (Schroder, Driver, and Streufert 1967). In order to define complexity for this study's evaluation process objectively, the number of elements in the system and the degree of the interactions among these elements is considered. For the

purpose of this evaluation, I divide task complexity into three groups: simple (e.g. between 2-5 elements/interactions), moderate (6-10 elements/interactions) and complex (10 and above). Based on the brief discussion above, a decision was made to limit task complexity up to moderate. The smart meeting room case presented in 4.3 includes not more than 10 logical services. Also steps/interactions required to complete a DQ assessment process by the users is not more than 10.

Expertise. Expertise is defined by the degree of knowledge and confidence participants possess in particular area. Literature accentuate that user's expertise is an important variable when conducting evaluations involving people (Fisher et al. 2003). Researchers suggested that expertise can have both positive and negative impact on the evaluation process (Fisher 2003; Buckland and Florian 1991). For example, experts may be better at using relevant information, whereas novice may be more sensitive to new information, hence assessing it more objectively. As mentioned before task complexity can be greatly affected by the expertise level. For instance, if user's expertise is relevant to the performed task, his perception of complexity may be much lower than novice's one. Considering above mentioned, a summary can be drawn that expertise is tightly related with working experience and domain specific knowledge. In order to simplify the evaluation process, two levels of expertise are used while conducting the workshop and use case study: expert and novice.

Interaction. The level interaction can be defined as the way of communication between participants while conducting the evaluation. This level can be individual or group. The participant as individual is undoubtedly important for evaluating things; however studies suggest that working together as a group has more benefits (Wetmore and Summers, 2003). Group possesses substantially effective advantages for improving evaluation quality. For example, groups are more successful than individuals in understanding problem, detecting alternative flaws and collecting broad relevant knowledge (Almeida and Marreiros 2005). Nevertheless, there are also some disadvantages while evaluating in groups, e.g. group members consume more time to come up with consensus feedback. Literature also suggests that interactive groups produce better quality feedback than non-interactive ones (Burlinson 1984). Taking

into account aspects mentioned above, in this process evaluation, interactive groups are considered before individuals.

Information presentation. One of the most important factors affecting evaluation process is the way in which the information is presented. Generally information presentation can be divided into three groups – graphical, tabular and textual. Unfortunately, there has been inconsistency in the literature with regard to whether graphical, tabular or textual ways of presenting information are more effective. For example, while Lusk and Kersnick (1979) reported that graphical displays are harder to use than tabular and textual ones, Ives (1982) found that graphics can have positive effects on information understanding. Conclusively, based on different situations, information presentations may impact the quality of the evaluation results. Hence during evaluation stage, all types of information presentation were provided.

Environment. Environmental aspect involves three parameters, namely ‘demographics’, ‘incentive/reward effects’ and ‘cheating’, that can influence my IT artifact evaluation. Following bullet points elaborate further:

- **Demographics:** Demographics factor includes the participants’ characteristics such as age, gender, educational level, occupation, etc. A number of studies (Buckland and Florian 1991; Fisher 2003) emphasized that demographical variation could generate different evaluation outcomes. During the evaluation process aforementioned demographic factors were considered.
- **Incentives/Reward Effects:** Incentives are typically introduced to guarantee that all participants involved into evaluation process will seriously consider the relevant information. Incentives such as cash or other rewards can be used to motivate participants to achieve high-result evaluation results.
- **Cheating:** Cheating is characterized by creating an unfair advantage in one’s own interest. During the evaluation participants may employ unfair approaches such as looking at neighbouring screens to complete the task. Hence, the leader and instructors of the initiative must guide and supervise the participants to effectively accomplish their tasks.

6.5 Workshop Layout

This section provides an overview of the workshop I have organized in order to evaluate the DQ process and its practicality of this research. Participants were invited to take part in a home automation use case. This case involved the execution of a few scenarios in smart office environment where they had to assess quality of information fed by sensors. Participants had to use the specially developed DQ software tool to practically execute the DQ process proposed in this study. After the use case scenarios were carried out, attendants were invited to provide their comments and feedback. Beside evaluation purposes, designed workshop had also the goal to teach and show the importance of data quality, and help participants make decisions while assessing ambient data quality in future.

The workshop was held in the University of Granada, Spain on the 23rd of March 2014. Participants were invited from both academia and industry. A total of 15 people attended the event. The guests were mainly postgraduate students from computer science and academic researchers who are interested in information system field. The attendees were mainly from University of Granada and Dublin City University.

The participants were expected to understand the impact of data quality on their everyday life. Throughout workshops' use case they were also assumed to learn how to define, measure, and analyse data quality which aligns with the objectives of the workshop.

The workshop began with introduction of the concept of data quality. To make it more understandable, a few cases that demonstrate the impact of poor data quality were presented. These cases covered data quality issues emerging from daily life such as inaccurate and misspelled information in different documents and web pages to large scale ones such as loss of large amount of money and property damage. An introduction and application of data quality in smart spaces and home was provided. Following on the case studies, researcher introduced the definition of data quality and presented different angles data quality can be considered. The host of the workshop

specifically stressed on clarifying the semantic meaning of data by providing real world examples.

After the concept of poor semantic DQ and smart spaces was clarified researcher of this study presented the DQ assessment process, followed by the DQ tool. The information was given form of PowerPoint presentation. Additionally, every participant received a stack of papers containing graphical and textual description of the key elements as well as quick user guide of the software tool. After all workshop attendees familiarized with the DQ process and tool, they were invited to get involved in a few home automation use case scenarios. The use case study aimed to put the participants in the role of data quality managers where they had to work out the quality of data feeding from variety of sensors in a smart meeting room. The quality of the data was ultimately determined by the participants. Further details about the case are presented in section 4.3. After they got familiar with the case they were separated into groups of 3 people. The number of people in each group was decided not to be exceptionally large due time constraints – e.g. large groups take longer time to take decisions and to cooperate than smaller. Each group was then assigned to a computer with running DQ software tool. On completion of their tasks, every participant received a survey containing questions about their understandability and experience about the system. A discussion about the overall DQ assessment process and possible improvements was also carried out with each group.

6.6 Workshop Planning and Settings

This section presents a discussion about workshop planning, external factor settings described in section 6.4 that might reflect the evaluation process. Following paragraphs deliver discussion about summary presented in Table 6-2 about workshop planning and settings.

Attendees were recruited mainly through circulating invitation emails within universities. Based on the artifact nature and research topic, target group for workshop were people who are familiar with the topic of data quality or at least have experience in information processing – data analysing etc.

Table 6-2: Workshop planning settings

Workshop goal	Recruitment method	Planned duration
<ul style="list-style-type: none"> Evaluate and Extract feedback Educate 	University Email system	4 to 5 h
Workshop size	Participants group level	Breaks
Up to 16 participants	<ul style="list-style-type: none"> Graduates Participants with knowledge in DQ and information processing area 	Yes, 2-3 breaks of 15 to 30 min each

Participants were expected to have college or university degree in areas of computer sciences, engineering, or economic studies. In the same time a quota of maximum 16 participants was set in order to enhance efficiency of the workshop results (e.g. each attendee have equal to opportunity and time to express its opinion). In addition the length of the workshop was decided not to exceed 4-5 hours. This was determined based on the workshops goals e.g. to educate and extract feedback from participants about DQ process, participants, level (e.g. how much they know in the area), their availability as well as on the planned tasks for execution and their complexity. Couple of short breaks (e.g. 15-30 min) were planned to allow participants to relax and network. The following paragraph discusses the external factors that were taken into account during the workshop. Table 6-3 provides overview settings about external factors and their parameters assigned during the evaluation process.

A total number of 15 participants took part of this research workshop. Demographic was the first information element that was collected. Attendees were at age of between 25 and 32. 73% of participants were male whereas 27% were female. All members had undergraduate degree. 93% of them were doing postgraduate course. Although most of participants were familiar and had some experience with information processing, none of them was able described himself as an expert in DQ field.

Table 6-3: Evaluation process settings

Expertise		Information Presentation			Interaction	
Expert	Novice	Graphical	Tabular	Textual	Group	Individual
No	Yes	Yes	Yes	Yes	Yes	No
Environment				Task Complexity		
Demographics	Cheating	Incentives	Simple	Moderate	Complex	
Yes	Yes	Yes	Yes	Yes/no	No	

As a consequence, their expertise level was defined as to be novice to moderate. In addition, the Interaction level was set to be in form of groups. Groups were chosen before individuals as research suggests that groups tend to perform tasks more efficiently - e.g. faster (Gladstein 1984), and since time for conducting the workshop was limited. Participants were allowed to interact within the group but not between groups. Communication with other groups was considered to be cheating. As a result, interacting behaviour of participants was monitored by the researcher. In section 6.3, task complexity was defined as the users' experience, the number of the elements in the system, and the degree of the interactions among these elements. From Section 6.5 it can be observed that system has between 5 and 10 elements and about the same number of interactions. Taking into account the participants' degree of expertise, the task complexity was set simple to moderate. Information through workshop process

was presented different forms e.g. textual, tabular and graphical. On completion of their tasks attendees were asked series of questions. After answering, workshop guests were rewarded with small tokens.

6.7 Conducting the Workshop

After participants became familiar with case settings and environment (see section 4.3), they were given the role of BIG room's administrators and data quality managers/analysts. Their task was to assess the quality of the room environmental data, including data subjects such as temperature, luminance, etc. Next four sections describe the process of assessing DQ in smart meeting room performed by participants.

6.7.1 Service/Information Profiling

Prior to start operating with the DQ tool, participants had to gather as much information as possible about services and data they were assessing. They were required to build information profiles structured in tables based upon the model described in chapter 4.6.1. Table 6-4 presents an example of information profiles submitted by one of the groups. During this process, participants were encouraged to use all kind of supporting documents describing the service composition and data such as composition maps such as UML diagrams, activity and sequence diagrams, etc. The researcher of this study also was responsible to help and answer any enquiries participants had in regard with service composition model. After groups were able to collect and structure information about service composition, they proceeded to the next step – composing DQ statements using the web based DQ application.

6.7.2 Composing DQ Statements

During this part of the process participants were required to build DQ statements according to their level of conformance about assessed data. To do so, attendees were provisionally logged into the DQ web system. Example screenshot is presented in Figure 6-4. The system led users straight to the DQ management screen. The interface of the application allowed users to 1) compose data quality statements; and 2) manage already existing statements.

Table 6-4: Information Profiling in Smart Meeting Room

<i>Who</i>				<i>How</i>		<i>What</i>					
Id (End point)	Service Name	Level		Operation Name	Operation Description	Param Name	Param Type	IN/OUT	Data Scale	Data Description	Remarks
0001	ServiceTemperatureSensor	0	Simple	OperationGetTemperatureSensor	Get measured temperature	tempOut	float	OUT	Degrees Celsius	Room Temperature	returned temperature value
0051	ServiceSetTemperature	0	Simple	OperationSetTemperature	Set Desired Temperature	air_set_in	float	IN	Degrees Celsius	Room Temperature	Airconditioner/Heater requested temp
				OperationGetSetTemperature	Return Set	air_set_out	float	OUT	Degrees	Room Temperature	Airconditioner/Heater
0002	ServiceBrightnessSensor	0	Simple	OperationGetBrightness	Return Room Brightness	brightOut	int	OUT	Lumens	Room Luminance	Sensor LT433
0052	ServiceLightBulb	0	Simple	OperationSetLightBulbPower	Set Light Power	bulbIn	int	IN	Watts	Power	set the value
				OperationGetLightBulbType	Return Type of the Light Source	bulbOut	int	OUT	Watts		return set value
0053	ServiceShutters	0	Simple	OperationSetShutterPosition	Set Shutters Position	posIn	int	IN	percentile	Shutter Position	
				OperationGetSetShutterPosition	Return Set Position	posOut	int	OUT	percentile	Shutter Position	returned set shutter position
1001	ServiceControlTemperature	1	Composite	OperationTemperatureControl	Control Ambient Temperature	tIn	float	IN	Degrees Celsius	Room Temperature	air_set_in = tIn
						tResponse	float	OUT	Degrees Celsius	Room Temperature	ambient temperature
1002	ServiceAmbientLightControl	1	Composite	OperationAmbientLightControl	Control Brightness	lumensOut	int	OUT	Lumens	Room Luminance	brightness
						lumensIn	int	IN	Lumens	Room Luminance	returned ambient
2001	ServiceAmbientComfort	2	Composite	OperationSetAmbientComfort	Set Ambient Control	tempIn	float	IN	Degrees Celsius	Room Temperature	set temp value
						brightIn	int	IN	Lumens	Room Luminance	set brightness value
				OperationAmbientComfortStat	Get Ambient Conditions	tempOut	float	OUT	Degrees Celsius	Room Temperature	get ambient temp
						brightOut	int	OUT	Lumens	Room Luminance	get ambient brightness

Prior composing any statements, participants first had to decide what data needs to be assessed. Then, they were required to refer that data to its profile – e.g. previously built during service profiling stage, then system aids them extract corresponding fetch/get function. After they allocate appropriate functions, they proceeded with actual building process. To compose a DQ statement, users first needed to input desirable name. As mentioned in 4.6.2, it is a good practice to conceive names in a way so they indicate the purpose of the DQ statements (e.g. “*assess temperature sensorsT1,T2*”). Then users were prompted to select fetch function from the drop down list containing service’s functions, following by selection of comparison operator and desired value (see Figure 6-4). At this stage, users were also able to compose complex statements by pressing the “+” symbol. When building composite statements, they were suggested to pick logical operators that link every simple statement. In addition to the composing functions, users were also able search, view, edit, and delete already existing DQ statements through the DQ statement explorer and viewer.

The screenshot displays the DQ Statement Composer, Explorer, and Viewer interface. The top navigation bar includes a clock showing 12:23 PM, the application logo 'DQ @ Home', version 'V. 0.3.1', and a 'Log Out' button. Below the navigation bar are three main sections: 1. Manage DQ Statements, 2. Execute, and 3. Report & Analyze.

The **DQ Statement Composer** section is active, showing a form to create a new statement. It includes a 'Statement name' field with the placeholder 'Enter statement name here'. The first step shows the function 'getTemperatureT1()' selected from a dropdown, followed by the comparison operator 'GREATER' and the value '25'. A yellow callout bubble points to the 'Add DQ Statement' button. The second step shows a logical operator 'AND' selected from a dropdown. Below this, a 'Select Service Operation' dropdown is open, showing options: 'getTemperatureT1' (ServiceTemperature), 'getTemperatureT2' (ServiceTemperature), and 'getTemperatureDHT1' (ServiceTemperature). The 'Operator' dropdown is also open, showing 'Enter Value Here'. The 'Save' and 'Cancel' buttons are at the bottom right.

The **DQ Statement Explorer** section shows a table of existing statements. The table has columns: ID, NAME, DATE, and ACTION. The data is as follows:

ID	NAME	DATE	ACTION
3123	Check T1 Sensor	12/03/2015	[Edit] [Search] [Delete]
3124	Check T2 Sensor	12/03/2015	[Edit] [Search] [Delete]
3125	Check T1 + T2 Sensor	12/03/2015	[Edit] [Search] [Delete]
3133	Check DHT11 Service	12/03/2015	[Edit] [Search] [Delete]
3134	Check DHT11 + T2 Service	12/03/2015	[Edit] [Search] [Delete]

The **Statement Viewer** section shows the details of a selected statement, 'Check T1+T2 Sensor'. The statement is displayed as a code block:

```
getTemperatureT1->temperature_c > 25
AND
getTemperatureT2->temperature_c > 25
```

The footer of the interface includes logos for DCU, HGT Universidad de Granada, Plamen Petkov, and a copyright notice for 2015.

Figure 6-4: Manage DQ Statements

6.7.3 Execution of DQ Statements

By clicking on the “Execute” (2) tab workshop participants were introduced to the execution screen as shown in Figure 6-5. It consists of two main panels – the DQ explorer pane and the DQ execution pane. From the DQ pane, users were able to search for particular DQ statement, e.g. by keyword. Then on their request, they were able to execute selected DQ statement. Executions can also be multiple. Additionally, users were allowed to commit delayed executions (e.g. at 12 o’clock noon time every day) or at certain intervals of time (i.e. every 20 minutes). From DQ explorer participants could also observe when the last execution for particular statement was. The DQ execution console offers information about the status of execution process, such as progress bar and status messaging.

The screenshot displays the DQ Statement Execution Console interface. At the top, there is a navigation bar with three tabs: "1 Manage DQ Statements", "2 Execute" (which is highlighted), and "3 Report & Analyze". The "Execute" tab is further divided into "DQ Statement Explorer" and "DQ Statement Execution Console".

The "DQ Statement Explorer" panel contains a table with the following data:

SELECT	ID	NAME	DATE	LAST EXECUTION
<input type="checkbox"/>	3123	Check T1 Sensor	12/03/2015	14/03/2015
<input type="checkbox"/>	3124	Check T2 Sensor	12/03/2015	14/03/2015
<input checked="" type="checkbox"/>	3125	Check T1 + T2Sensor	12/03/2015	n/a
<input type="checkbox"/>	3133	Check DHT11 Service	12/03/2015	12/03/2015
<input checked="" type="checkbox"/>	3134	Check DHT11 + T2 Service	12/03/2015	n/a

Below the table, there is a "Delayed Execution" section with a "Execute Selected" button and an "Apply" button. The "Apply" button is accompanied by a dropdown menu for frequency (every 30 min) and a time selector (09:30 AM every DAY).

The "DQ Statement Execution Console" panel shows the execution progress and status. It includes a progress bar and a "Clear" button. The console output shows the following messages:

```
Executing... "Check_T1+T2_Sensor" ...  
... "Check_T1_Sensor" ... OK!  
... "Check_T2_Sensor" ... OK!  
... OK! Report "120320151620.xml" created.  
  
Executing... "Check_DHT11+T2_Sensor" ...  
... "Check_DHT11_Sensor" ... FAIL!  
... "Check_T2_Sensor" ... OK!  
... FAIL!
```

Figure 6-5: Execution of DQ statements

6.7.4 DQ Analysing and Reporting Stage

The final part of the DQ assessment process is the problem reporting and analysing stage. Users were able to access this option by clicking on to the third tab (3) as shown in Figure 6-6. This part of the application consists of two main windows – DQ Report Explorer and DQ Report viewer. The DQ report explorer contains a list of reports generated during DQ Execution process. At the time of writing this dissertation, DQ tool offered user the option screen reports only by date. Participant then could view or delete reports they were created. The DQ Report viewer displays the result from each report. More specifically, it presents a table containing data subject's violated values and a link to related DQ statement. Viewer pane also includes information about any services and function that are related with the violated data subject. The example on Figure 6-6 shows the violated value of the *brightOut* data subject operated by the *BrighnesSensor* service.

DQ Report Explorer

ID	REPORT NAME	DATE	ACTION
125	120320151610.xml	12/03/2015 @ 16:10	View report
126	120320151615.xml	12/03/2015 @ 16:15	View report
128	120320151719.xml	12/03/2015 @ 17:19	View report
129	120320151742.xml	12/03/2015 @ 17:42	View report
130	120320151745.xml	12/03/2015 @ 17:45	View report

DQ Report Viewer

120320151719.xml

Service Location	Service ID	Service Name	Operation Name	Data Subject	Value	Ref. Value	Statement ID
http://1.0.0.19/doha/big/serviceBrightness/operation/GetBrightness	002	ServiceBrightnessSensor	operationGetBrightness	brightOut	323	500	3163

▼ Service Dependencies

Service Location	Service ID	Service Name	Operation Name
http://1.0.0.19/doha/big/serviceLightBulb/operation/SetLightBulbPower	052	ServiceLightBulb	operationSetLightBulbPower
http://1.0.0.19/doha/big/serviceShutters/operation/SetShutterPosition	053	ServiceShutters	operationSetShutterPosition
http://1.0.0.19/doha/big/serviceAmbientLightControl/operation/CompositeLightControl	1002	ServiceAmbientLightControl	operationAmbientLightControl

DCU | Universidad de Granada | Plamen Petkov | copyright 2015

Figure 6-6: Report and Analyse Screen

As seen from the screenshot, there were three services/operations that would affect the *BrightSensor* service. In this case these were *ServiceLightBulb*, *ServiceShutters* and *ServiceAmbientLightControl* services. Participants were asked to execute the whole DQ process several times. In order to check effectiveness of the DQ software tool, organiser (the researcher of this study) had to manually introduce poor data. For example, the ambient light sensor was covered with a piece of cloth for 10 min which resulted in faulty behaviour of the services related with it. Fortunately, participants were able to capture and locate the source of erroneous data from nearly all the case scenarios they were offered to execute.

6.8 Results and Discussion

To rigorously evaluate the utility (also referred as the usability) of the system and user experience, participants were ask to complete a questionnaire. As mentioned earlier the questionnaire was built upon Systems Usability Scale (SUS) model. It consisted of 10 questions. Each question had five response options - from (1) “Strongly agree” to (5) “Strongly disagree”. The SUS was employed since it allowed evaluating a wide variety of products and services, including hardware, software, mobile devices, websites and applications. It is also suitable for this research since it can be used on small size samples with reliable results (Brooke 1996).

At the end of the workshop, I have used the SUS scoring to calculate the average result. The participant’s scores for each question are converted to a new number, added together and then multiplied by 2.5 to convert the original scores of 0-40 to 0-100. Though the scores are 0-100, these are not percentages and should be considered only in terms of their percentile ranking. Based on research of over 500 studies (Sauro 2011), a SUS score above 68 would be considered above average and anything below 68 - below average. According to Sauro, the best way to interpret this score is to view it as a percentile as shown in Figure 6-7: SUS Curve . The average score during workshop was 72 which correlates to an above average percentile of around 61%. This means that 72 in the SUS scale represent a score higher than 61% of the 500 studies done. Although SUS does not provide objective evaluation, SUS data score could be used for a reference, if further evaluation is conducted.

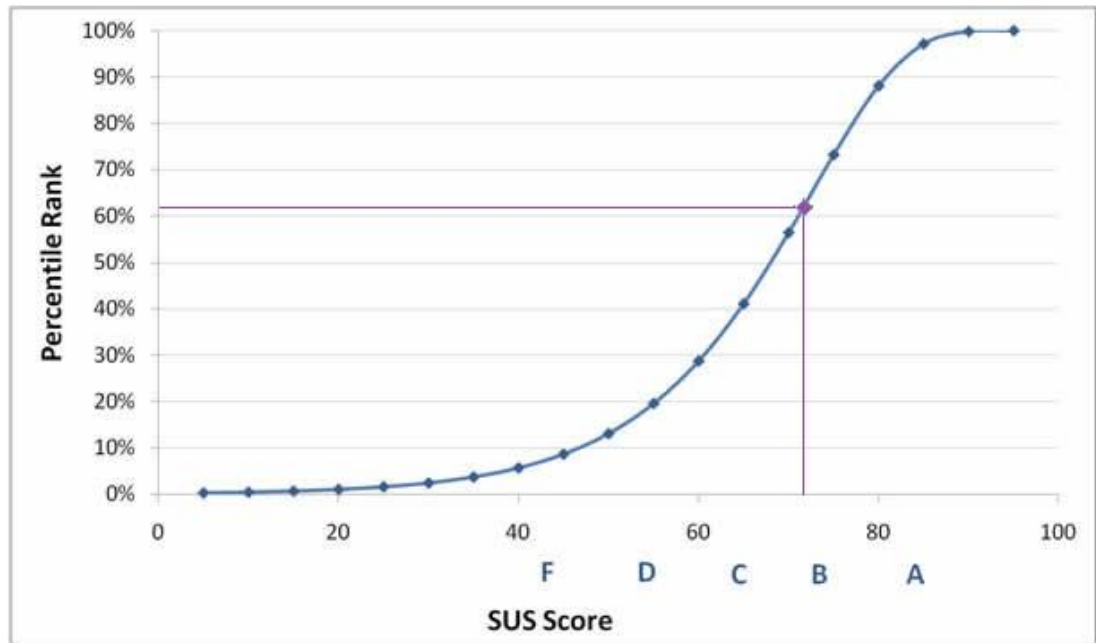


Figure 6-7: SUS Curve (Sauro 2011)

As mentioned in chapter 3.4.3.1 and 6.2, in addition to quantitative conclusion that were drawn above, I have employed an open-question approach along the SUS questions to gauge the evaluation criteria discussed in 6.3. Reflecting to the research criteria, here were four main areas of concern; *completeness*, *consistency*, *ease of use* and *interface design*. A mapping between SUS statements and evaluation criteria is introduced and it was based on the core definitions of the criteria itself and interpretation of the SUS constructs (e.g. statements). This is presented in Appendix B. Next paragraphs discuss the outputs that were drawn upon analysing the open-ended question regarding each criterion.

6.8.1 Completeness

In section 6.2, the “completeness” was defined from two perspectives – from ‘DQ process’ and ‘software tool’ perspectives. The tool represents physical implementation of the process. It is realized through web interface to communicate the process and its main steps with the user. From DQ process perspective the completeness is a qualitative criterion that describes whether the DQ assessment methodology comprises all necessary information and functionality it was designed for – e.g. to detect poor semantic data. On the other hand, from ‘software tool’ (e.g. the physical representa-

tion of the process under evaluation) point of view, completeness was described as to whether or not it has all necessary functionality to support DQ assessment process.

With regard to the first aspect of completeness, participants from the workshop were not able to provide an explicit and consistent answer due to the theoretical nature of the artifact, and fact that participants were not experts in data quality management theory. Nonetheless, the measurement of completeness was easier through evaluation the completeness of the software tool. Generally speaking, the software tool is considered 'complete' if it strictly follows the DQ process, where DQ process is considered complete if it allows the detection of poor semantic data. When asked to evaluate the completeness of the DQ process most of the participants were not consistent in their responses. One participant said that: *"the data quality process wasn't very straightforward from the initial presentation, but it only get to become clearer when it was put into the Home Automation domain and when I started using the tool."* This statement was interpreted objectively since, as stated above, participants did not identified themselves as experts in data quality domain but only had some previous experience with data processing. Another supporting statement was: *"I found the theoretical part hard to grasp but with the aid of the web enabled portal it was easy to understand the process and I was able to start using the tool with very little help from the organizer."* From the latter statements it can be deducted that participants had some difficulties understanding the theoretical process. Possible reason for this could be the information overload.

Overall, the participants were predominantly positive in their responses to the question relating to the completeness of the system. The main statement that reflected into evaluation analysis was the extent to which participants perceived functionalities were well integrated. Most of them agreed that the functions of the system were well integrated. For example one participant stated: *"Generally, the tool complies the data quality process introduced in the beginning of the workshop"*. Many found it all the necessary components for usability. However, there were a few coding statements that had negatively value to the completeness criterion. For example, a few participants gave suggestions for improvement stating that they would like to see a unit of measure added when composing the DQ statement. Another recommendation given

was to add a few more viewing options in the service dependencies in order to better analyse any potential issues.

6.8.2 Consistency

Consistency criterion was the second of all four denoted in section 6.2. It was identified as one of the sub-categories to measure qualitatively the usability of the DQ process and Web System and interface. Likewise completeness, consistency definition has two facets - 1) DQ process and 2) DQ software tool.

The Consistency of the DQ process comprises the notion of whether all parts of DQ process are mutually compatible and work without contradicting. For example *“Does the end of one stage of the DQ process conclude with element compatible with the beginning of the next stage?”* Or is *“DQ process described in a consistent manner?”* Analysis from the survey reported that all of the participants agreed that there were little to no inconsistencies in of the DQ process. For example one subject reported that *“consistency of the DQ process was achieved by adopting the BPMN standard”*. Other stated that *“activities blocks at the end of each DQ sub stage ended in consistent way so, next stage can benefit from it”* which indicated consistency of the process

The other perspective of consistency dictates whether or not DQ tool is consistent with the DQ process – e.g. *“Does it reflect the DQ process?”* When asked participants if they believed there were too many inconsistencies they greatly disagreed. Attendees reported that the system was consistent with the DQ process presented to them prior using the software. Nevertheless, one withdraw was identified when analysing the user’s data - all of the participants replied that ‘information profile’ stage was missing from Web screens: *“The web tool’s three main functions e.g. ‘Managing DQ Statements’, ‘Execute’ and ‘Report & Analyze’ consistently reflected the Data quality process described in beginning of the workshop. The only missing phase was the profiling.”*. This was noted down and could be considered in future version of the DQ tool. From user interface perspective consistency was rated with top marks. Attendees noted that each web screen was fitting with the ones following or prior. Also they reported that all interface elements such as buttons, fonts, shapes and tables were very consistent: *“The User interface was well structured and every each block re-*

ferred to a particular action. Screens during different stages of the process were consistent e.g. colours fonts, pictograms, etc. Overall the tool was reflecting the DQ process except profiling.”

6.8.3 Ease of use

Similarly to *completeness* and *consistency*, the *ease of use* criterion consists of two definitions – each of them target different aspect of the research artifact. The first one measure the effort needed to understand the DQ process, whereas the second definition takes into account the time that participants needed to understand how software tool works. It is important to note that, external factors such as ‘user expertise’ and ‘graphical presentation’ were taken into account while processing results.

When asked to evaluate the ease of use of the DQ process, most of the respondents identified the process as hard to understand from provided presentation. For example one response was: *“...it wasn’t very straightforward from the initial presentation, but only it get to become clearer when data quality assessment process was applied into the smart meeting room and when I started to use the tool.”*; other complemented: *“I found the theoretical part hard to grasp but with the aid of the web enabled portal it was easy to understand the process and I was able to start using the tool with very little help from the organizer.”* Such statement was identified as objective since the respondents who provided it had little experience with process modelling. Nevertheless, there were a few that were confident in understanding the DQ process from the very beginning and reported its use to be very easy and clear. These participants had prior knowledge in process modelling and the BPMN specification.

On the other hand, when asked to assess the ease of use of the DQ software tool, many responded they found it easy after some additional explanation from the researcher/user manual. They discovered with a little training they would be able to understand and use the functions. – *“The DQ process was easy to follow through the navigation bar on the top, in the web page. For first-time users, the numbered labeling presented a great view of the execution sequence of the process”* The participants were also asked if they believed most people would be able to quickly learn this system and the response was positive with a few stating it would require some previous knowledge of DQ systems.

6.8.4 Interface Design

The design of the user interface has a great impact on the ease of use and the overall user experience and its evaluation was made based on the “feel” of the user when using the web interface. In section 6.2, the criteria for evaluation the interface design was stated as *“Whether or not interface elements are accessible and easy to find.”*

Overall the participants were very pleased by the design of the interface. Several stating they found it easy to follow. When asked in the SUS survey if they thought the system was cumbersome, majority disagreed stating it was well structured with the layout of each page being clear and the colour scheme simple. The design of interface was also said to *“be sufficient in appropriately represent the functionality described in the DQ process”*. Also important elements such button and input fields were positioned in accessible space – e.g. one statement was *“The interface of the application was simple and intuitive. Elements such as input fields, drop down, tables, buttons, colour themes, etc. were designed in a consistent manner, which made use of the app feel good”*.

On the other side, there were a few respondents who believed there were some areas for improvements. For example, one suggested that the service dependencies section in the report analysis needed some restructuring for better usability. Another recommended more textual guiding/prompting to make the system more comprehensible to non-technical people. Lastly, it was stated that the font size was too small and was sometimes difficult to read.

6.9 Summary

Aim of this chapter was to answer the third research question RQ3 – *“How to evaluate the utility of the developed Data Quality assessment method?”* Particularly, the focus was to evaluate the utility of the artifact from two perspectives – 1) DQ assessment process and 2) the designed data quality software tool for DOHA home automation system. In order to answer the RQ3, questions RQ3.1 and RQ3.2 had to be answered in beforehand.

Firstly, RQ3.1 asked: “*How to define and evaluate effectiveness of the Data Quality assessment method within the boundaries of this research?*” To answer this question two main topics had to be considered – one is how to ‘define’ effectiveness in terms of utility of artifact, and the other is how to draw justification against evaluation criteria. Section 6.3 contributes to the first topic by specifying utility criterion and decomposing it into several other research criteria, namely *completeness, consistency, ease of use, and user experience* which cover different aspects of assessment process and software tool. In this section, definitions of each criterion were given. With reference to the second topic, section 6.2 presents a process for the evaluation of the artifact against the criteria scoped earlier. The artifact evaluation was aided by conducting a workshop where participants were introduced into the area of DQ and home automation, and were offered to execute the DQ process using the developed DQ tool. Participants were given a few execution scenarios to play with in a typical smart meeting room setting.

To support RQ3, a second sub question RQ3.2 was defined – i.e. “*Does the designed assessment method meet the requirements for its research purpose?*” To answer this question, at the end of the workshop (described in sections 6.5 and 6.6), participants were requested to give their feedback with regard evaluation criteria defined in response to RQ3.1. The evaluation process described in section 6.2 and 3.4.3.1 suggested that collection of participant’s feedback is done through the use of the System Usability Scale (SUS) tool along with qualitative analysis. The final section of this chapter presents a summary of users’ experience with DQ process and tool. Results from SUS survey and qualitative data analysis showed that participants’ opinions were predominantly positive with regard presentation of the DQ process and usability of the DQ software tool. They were in consensus that newly introduced DOHA extension implements the theoretically described DQ assessment process, and that they would benefit from using in to ensure the quality of data their automated homes. Overall users’ opinions imply that the answer of RQ3.2 is a definite ‘yes’. However, one must note that this conclusion is made only in the context of DOHA SOA system and Smart Meeting Room case. Stemming from this, it would be an interesting direction for future work to apply DQ in different SOA context to achieve greater generalisation of the evaluation of the DQ process.

Finally, Table 6-5 presents a summary of the research questions and contributions along with references to the sections where were addressed.

Table 6-5 Research Questions addressed in Chapter 6

Research Questions Addressed in this Chapter	Section Reference	Summary and Contributions
RQ 3 How to evaluate the utility of the developed Data Quality assessment method?	Chapter 6, Sections 6.1-6.8	<ul style="list-style-type: none"> - Provides evaluation of DQ process in Smart Meeting Room case applying developed DOHA DQ software tool. - Evaluates the overall DQ process
RQ 3.1 How to define and evaluate effectiveness of the Data Quality assessment method within the boundaries of this research?	Sections 6.2-6.5	<ul style="list-style-type: none"> - Definition and description of artifact evaluation criteria such as <i>completeness, consistency, ease of use, and user experience</i> was given - Evaluation plan was proposed - Conducted extensive workshop to provide DQ training and artifact evaluation in against criteria
RQ 3.2 Does the designed assessment method meet the requirements for its research purpose?	Sections 6.6 and 6.7	<ul style="list-style-type: none"> - System Usability Scale tool and Qualitative analysis were used to collect and process evaluators feedback. - Results indicated that designed DQ process and software tool meet the research requirements (e.g. to be able to assess DQ in DOHA SOA system and to present a transparent the DQ process execution in smart home automated systems)

Chapter 7 Conclusion

7.1 Summary and Contributions

Individuals and companies rely highly on a variety of information systems to perform daily business tasks. Service Oriented Architectures is an extensive and comprehensive approach that serves as a means to deal with the diverse, complex and numerous fast evolving technologies and computer application of today. SOA offers a set of principles and standards that allows the development of complex applications by combining together a number of simple ones. A key benefit that SOA brings is the delivery of new function combinations rapidly. On the other hand, however, the higher complexity of such systems increases the risks of developing poor information quality, in terms of propagating inconsistent and inaccurate data. Observation in DQ literature and SOA practice indicated the lack of methodological and systematic solutions that can provide transparency when assessing the quality of information running through the SOA environment. As a consequence the objective to design and build a solution that tackles the aforementioned problem was set as central to this study. Consequently, to scope this research, RQ1 which asks “*What are the requirements needed for applicable DQ assessment in SOA context?*” had to be answer. To elaborate answering RQ1 breaks down into two sub questions – RQ1.1 – “*What are main criteria for DQ methodology and how different DQ methodologies compare against them?*” and RQ1.2 – “*What are the key principles embedded into Service-oriented Architectures that affect Data Quality?*”

In order to reach answer of RQ1.1, a systematic literature review in the area of DQ was conducted. The output of this review was to identify the most relevant DQ frameworks such as TDQM (R. Wang et al. 1998), DWQ (Jeusfeld, Quix, and Jarke 1998), AMIQ (Lee et al. 2002) TIQM (English 1999) IQM (Eppler and Muenzenmayer 2002) (see Table 7-1: Data Quality Methodologies, pp. 33) as well as criteria such as *type of data, types of IS, phases and steps* that comprises DQ methodology, *techniques and strategies* and finally *dimensions* that are used in the method-

ologies to compare them. As a consequence from the analysis, a gap in the DQ literature was identified. A solution for assessing data quality effectively is needed in organizations and initiatives that follows SOA paradigm. Emerging from literature review of data quality, a few further questions can be found significant for developing DQ assessment process, that is: *What sort of particular DQ problem is tackled?*, *What dimension(s) has to be considered?*, and *What step and phases are required in order to overcome stated DQ problems?* For example, research scope defines the need of a DQ solution that tackles *poor semantic data problems*. The dimension related with this type of problems is *accuracy*; and the steps and phases required to overcome such problems as mainly three – DQ definition, DQ measurement, and DQ improvement. The scope of this research excludes the DQ improvement stage where strategies for improving the quality of information are proposed.

To answer the RQ1.2, the relationship between data quality application and SOA had to be investigated. An overview of SOA research provided numerous key principles such as service *composability*, service *reusability*, *statelessness*, *autonomy*, etc. that are needed for an effective service composition. However, a few SOA principles were found to be related with the quality of information. For example, service *abstraction* and *autonomy* suggest that data and process should be abstracted e.g. services hide logic and physical structure of data from the outside world. If any poor quality information moves in and out of such “transparent” services, its detection could become extremely challenging. More thorough discussion about key factors affecting DQ was presented in section 2.2.2.1. Another concept that also influences DQ in SOA was the Quality of Service, and it was discussed in 2.5.2.2. An example for such QoS is the *Timeliness* category which contains dimensions relating to the end-to-end delay of the service data flow. Such delay depends upon several aspects of the SOA system and service capabilities e.g. network delays, physical service execution time, data load, etc. Also delays like that could potentially lower the quality of information at the consumers end. On the other hand, literature confirmed there is a tight link between the DQ and the context they are applied in. This was discussed in section 2.5.1. Therefore, considering aforementioned perspective and SOA principles that affect DQ, a method for incorporating DQ assessment process into particular SOA environment had to be delivered. RQ2 elaborates on this matter.

The main research question – e.g. RQ2 – “*How to assess Data Quality in Service-oriented architectures’ context?*” marks the main part of this thesis. In order to resolve this question, a few sub-questions had to be answered first: e.g. “*How to define ‘poor’ Data Quality within the context of SOA?*”(RQ2.1), “*How to detect poor data in the service-oriented environment?*”(RQ2.2), “*How to design Data Quality assessment method?*”(RQ2.3), and “*How to implement the Data Quality assessment method?*”(RQ2.4) To address the first sub-question, literature of DQ that focuses on theories concerning the expression of information was investigated. Semiotic theory was identified as an approach that concerns the process of analysing and understanding symbols and data. Other constructs that are found in literature, and which elaborate on the definition of “good/poor” information was the concept of contextual data (e.g. environmental data – time and space) and user’ constraint data (e.g. users defined value restrictions about particular data). The answer to the second sub-question - how to detect DQ - requires the accommodation of methodology that takes into account mentioned theoretical constructs (contextual data and user data definitions), and the specifications of the SOA approach. To achieve this, Total Data Quality Methodology was used as a backbone construction of assessment method because it provides an overarching approach and philosophy that takes into account all aspects of the DQ lifecycle. This and the developed DQ approach were described in detail in Chapter 4. To reflect on the third sub-question – how to design DQ assessment method in SOA - I turned to the Design Science research approach as it specializes in the design and construction of various IT artifacts. Chapter 3 and Chapter 4 provided in depth discussion of why and how DQ assessment process artifact was designed. The answer to the final sub-question requires knowing the particular SOA environment from technical point of view. This question is optional and is applicable in cases where SOA middleware do not offer any data assessment functionalities. For such cases, I have proposed a method for incorporating DQ approach into different SOA initiatives by following reverse engineering approach which offers unambiguous way of investigating already functioning system with the intentions to improve it.

The third research question (RQ3) was set to query the evaluation of the designed artifact. More specifically, the intention was question the utility of the DQ process in practice. Design Science research (Peffer and Tuunanen 2007; Gill and Hevner

2013) requires that upon construction of the artifact evaluation must be clearly demonstrated. The validation of my method is performed through a verification of its implementation. A particular case of service oriented environment called Dynamic Open Home Automation system (Holgado-Terriza and Rodriguez-Valenzuela 2011) was examined. The DOHA case study provided an excellent opportunity to incorporate the DQ assessment process developed in this study with respect to semantic data quality. It also acted as development and improvement ground for the process steps and models. This involved prototyping, and discussions, and was iterative in nature with many of the participants (e.g. architects and developers) being central to the process. This allowed us to apply and revisit my DQ process fragments regularly until we refined each of them. To fully address RQ3 two sub-questions were asked: RQ3.1 – *“How to define and evaluate effectiveness of the Data Quality assessment method within the boundaries of this research?”* and RQ3.2 – *“Does the designed assessment method meet the requirements for its research purpose?”*

To answer RQ3.1 two main topics had to be considered – one is how to ‘define’ effectiveness in terms of utility, and the other is how to ‘evaluate’ the artefact against specified criteria. Section 6.3 contributes to the first topic by defining utility criterion and decomposing it into several research criteria, namely *completeness, consistency, ease of use, and user experience*. In that section, definitions for each criterion were given. With reference to the second topic, section 6.2 presents a process for the evaluation of the artifact against the criteria indicated previously. The artifact evaluation was aided by conducting a workshop where participants were introduced into the area of DQ and home automation, and where they were offered to execute the DQ process using the developed DQ tool. Participants were also assigned a few scenarios to execute in a typical smart meeting room setting.

To answer RQ3.2 – i.e. *“Does the designed assessment method meet the requirements for its research purpose?”*, at the end of the workshop (described in sections 6.6 and 6.8), participants were requested to give their feedback with regard evaluation criteria defined in response to RQ3.1. The evaluation process described in section 6.2 and 3.4.3.1 suggested that collection of participant’s feedback is done through the use of the System Usability Scale (SUS) tool along with qualitative anal-

ysis. Chapter 6 of this dissertation presents a summary of users' experience with DQ process and tool. Results from SUS survey and the qualitative data collected. Analysis showed that participants' opinions were mainly positive with regard presentation of the DQ process and usability of the DQ software tool. They were in consensus that newly introduced DOHA extension implements the theoretically described DQ assessment process, and that they would benefit from using it to ensure the quality of data their (future) automated homes. Overall users' opinions imply that the answer of RQ3.2 is a definite 'yes'. However, one must note that this conclusion is made only for context of DOHA SOA system and Smart Meeting Room case. It would be an interesting direction for future work to apply DQ in different SOA context to achieve greater generalisation of the evaluation of the DQ process.

This study contributes to the academia and industry by addressing the importance of the quality of information in the service oriented context in the following ways:

- **Process oriented artifact.** In light with Gregor's classification of theoretical contributions, my artifact falls under type 5 – design and action theory. This type of theory explains how to do something and gives explicit prescriptions (i.e. methods, techniques, and principles in form of functions) for construing artifacts needed for achieving specified goals (Gregor 2006). Data Quality literature and research provides numerous frameworks, criteria and methodologies to guide enterprises in the assessment, analysis, and improvement of DQ (Y Wand and Wang 1996; English 1999; Lee et al. 2002; C Batini and Scannapieco 2006). However, none of them present a process to measure and assess data quality for emerging and diverse service oriented architectures. I contend that my approach takes into account SOA specifications and standards and provides a methodological step-by-step process that aims to assess the accuracy of the information running through SOA environments based on data consumer' requirements.
- **Theory based.** From a theoretical point of view, I contend that my approach extends general data quality management theories, such as TDQM, by adding knowledge for assessing information within SOA context. Additionally, my approach contributes to the DQ research by addressing the significance of the

data quality and providing researchers with foundations for further investigation in the area of DQ, particularly, in improving quality of information and decision making in SOA. Furthermore, I contribute to DSR by presenting a research process that is centred on the building and evaluation IT artifacts.

- **Practically driven artifact.** The main beneficiaries of this research are the practitioners. The proposed DQ assessment approach is flexible enough to be accommodated in fast-changing and diverse SOA environments. This approach is modular as it comprises four steps which have defined points of integration. Consequently, as summarized in Table 4-8 – section 4.8, my artifact contributes to IS/DQ managers, architects, administrators, and software engineers by suggesting practical methods and techniques for detecting inaccurate semantic data. Additionally, this research provides the industry with a useful guide for better understanding data quality in the SOA context. Finally, the instantiation of the artifact within Dynamic Home Automation system would serve as a basis for research in intelligent and context-aware systems where the quality of information is a crucial factor in making such systems proactive – e.g. where timely decisions are needed.

7.2 Limitations and Critical Discussion

From a critical point of view, this project does not deliver a one-size-fits-all DQ assessment process, even though on a higher level the DSR abstraction principle is fulfilled. Domain specific requirements imposed by different SOA settings and environments might change the design of IT artifacts or exchange one artifact with another serving the same purpose. The question whether my approach is applicable to other SOA environments, or disciplines other than SOA, can be positively answered. However, current approach needs to be adapted for each individual SOA case or organization in order to deliver a value.

Another critical aspect is that the DQ assessment process proposed in Chapter 4 is fairly broad from a conceptual standpoint. Nevertheless, in order to accommodate my research requirements, I need to provide a flexible and comprehensive method while preserving as much simplicity as possible. Additionally, I am acquainted with the fact that my proof of concept and artifact is not as evolved as to deliver a complete solu-

tion to the DQ problem. Despite this fact, I may collaborate with a particular industry and use my DQ oriented guidelines as a starting point to help refine and deliver specific solution to the problem.

In accordance with the DSR, ‘efficiency’ is an evaluation factor that considers resources such as money, equipment, and people’s time. It suggests that to work efficiently, the artifact should apply resource constraints or minimize their consumption. Venable et al. identify that evaluating the artifact must be carried out by comparing it to other solution technologies. “A new artifact should provide greater or relative efficiency than existing artifacts that can be used to achieve the same purpose” (Venable, Pries-Heje, and Baskerville 2012)(p.5). Unfortunately, through this research course, it was not possible to evaluate overall efficiency of the DQ process (e.g. time and cost to implement). This limitation stemmed from the fact that currently available artifacts – e.g. DQ methodologies and tools are not directly comparable with the artifact developed in this study. More specifically, there is no other solution that is designed to have exactly the same or similar purpose as the one presented in this work. However, parameters such as ‘implementation time’ – i.e. time spent for deriving DQ process and implementing software tool, and ‘resource costs’ were recorded. Consequently, these parameters can be considered as a comparative basis for future studies that aim to evaluate the artifact efficiency – e.g. after it is improved or similar artifact is developed. Other reasons for this drawback include limited research scope and time constraints.

7.3 Future Work

Gill and Hevner stated that IT artifacts are divided between test instances - for which returning to the design cycle is intentionally left open as a possibility - and release use instances, for which further redesign is not anticipated (Gill and Hevner 2013). In this study, I have conducted a workshop and a use case test to evaluate the artifact leaving it open for future work. As stated in chapter 5 - section 5.4, my DQ approach was incorporated into DOHA with the intention to provide context awareness. DQ software that is “context aware” is worthy of further exploration and could prove to be research rich. The widespread use of wearable and mobile devices and deployment of small scale home and office applications with a very specific purpose is a scenario

that opens potential opportunities for DQ assessment. The development of applications for specific purposes has the potential to make DQ more pervasive. Figure 5-8 and Figure 5-10 in Chapter 5 included a possible configuration for context aware DQ applications that incorporates users' "fit for use" requirements and environmental situation.

The possibility of my work's application to other SOA situation over a long time period has the potential to further refine my findings. In-depth analysis of every SOA environment and consequently rigorous incorporation of my DQ process into each SOA would greatly enhance this study. The employment of design science techniques in my research facilitated construction of my method. Refinements of DQ and design science approaches could therefore have potential for a more widespread adaptation of this study in the DQ domain.

As DQ is becoming more and more important for today's fast evolving IT, the need for systematically and methodically constructed DQ artifacts will become imperative. The developed DQ assessment process, along with the future usability of specially designed DQ software tool in home automation domain, provides a mean for enhancing DQ research and practice. The scale of current information systems and its application in all aspects of society will demand for the highest quality of information.

Publication List

- Petkov, P.** and Helfert, M. 2014. “A Methodology For Analyzing and Measuring Semantic Data Quality in Service-Oriented Architectures.” In *CompSysTech '14 Proceedings of the 14th International Conference on Computer Systems and Technologies*, 201–8. <http://dl.acm.org/citation.cfm?id=2516782>.
- Petkov, P.** and Helfert, M. 2013. “Developing a Data Quality Methodology in Service Oriented Context Using Design Science Approach.” In *Perspectives in Business Informatics Research*, 158:254–66. Springer. doi:10.1007/978-3-642-40823-6_20.
- Petkov, P.**, Ł Ostrowski and Helfert, M. 2013. “Process for Assessment Data Quality in Complex Service Oriented Architectures Using Design Science Approach.” In *European Design Science Symposium, EDSS 2012*, 76–87. doi:978-3-319-04090-5.
- Petkov, P.**, Rodriguez-Valenzuela, S., Helfert, M. 2013. “Modeling Context-Awareness in a Pervasive Computing Middleware Using Ontologies and Data Quality Profiles.” In *Evolving Ambient Intelligence. Communications in Computer and Information Science Volume*, 271–82. Springer.
- Petkov, P.**, Helfert, M. and Pham, T. 2012. “Discovering Data Quality Issues in Service-Oriented Architectures: A Framework and Case Study.” In *BMSD 2012 - Proceedings of the 2nd International Symposium on Business Modeling and Software Design*, 72–80.
- Petkov, P.** and Helfert, M. 2012. “Data Oriented Challenges of Service Architectures. A Data Quality Perspective.” In *Proceedings of the 13th International Conference on Computer Systems and Technologies CompSysTech '12*, 163–70. Ruse: ACM. doi:10.1145/2383276.2383302.

References

- Abiteboul, S., P. Buneman, and D. Suciu. 2000. "Data on the Web: From Relations to Semistructured Data and XML", Morgan Kaufmann Publishers Inc. San Francisco, CA,
- Adelman, S., L.T. Moss, and M. Abai. 2005. *Data Strategy*. Addison Wesley.
http://www.informationbuilders.com/new/magazine/v16-1/PDF/21_IBIw06.pdf.
- Aladwani, A.M., and P.C. Palvia. 2002. "Developing and Validating an Instrument for Measuring User-Perceived Web Quality." *Information & Management*. Volume 39 Issue 6, pp. 467 - 476
- Albert, W., and T. Tullis. 2013. "Measuring the User Experience: Collecting, Analyzing, and Presenting Usability Metrics." 2nd Edition, Morgan Kaufmann
- Almeida, A., and G. Marreiros. 2005. "A Scheduling Model Based on Group Decision Support." In *IEEE International Conference on Systems, Man and Cybernetics*, 1180–87. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1571306.
- Austvold, E., and K. Carter. 2005. "Service-Oriented Architectures: Survey Findings on Deployment and Plans for the Future." *AMR Research*.
- Ballou, D., R. Wang, H. Pazer, and G.K. Tayi. 1998. "Modeling Information Manufacturing Systems to Determine Information Product Quality." *Management Science*. Volume 44 Issue 4, pp 462-484
- Baresi, L., and S. Guinea. 2009. "Dynamo + Astro: An Integrated Approach for BPEL Monitoring." In *International Conference on Web Services, ICWS '09*, 230–37. Los Angeles, CA. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5175828.
- Baskerville, R, J Pries-Heje, and J. Venable. 2009. "Soft Design Science Methodology." In *Design Science Research in Information Systems and Technology DESRIST '09*. <http://dl.acm.org/citation.cfm?id=1555631>.
- Batini, C., and M. Scannapieco. 2006. "Data Quality: Concepts, Methodologies and Techniques." *Journal ACM Computing Surveys (CSUR)* 41: Article no.16. Volume 41 Issue 3.
- Batini, Carlo, Cinzia Cappiello, Chiara Francalanci, and Andrea Maurino. 2009. "Methodologies for Data Quality Assessment and Improvement." *ACM Comput. Surv.* 41 (3). New York, NY, USA: ACM: 16:1–16:52. doi:10.1145/1541880.1541883.
- Batini, Carlo, and Monica Scannapieco. 1998. *Data Quality: Concepts, Methodologies and Techniques (Data-Centric Systems and Application)*. Springer Berlin, Germany.
- Bauchot, F., and H. Lalanne. 2015. "Data Integration in Service Oriented Architectures." In *Business Machines Corporation*. Armonk, NY, US.
- Bean, James. 2009. *SOA and Web Services Interface Design: Principles, Techniques, and Standards*. 1st Editio. Morgan Kaufmann publications.

- Bennett, S., T. Erl, C. Gee, R. Laird, A. Manes, and R. Schneider. 2011. *SOA Governance: Governing Shared Services On-Premise and in the Cloud*. 1st ed. Prentice Hall/PearsonPTR.
- Bertossi, Leopoldo, Flavio Rizzolo, and Lei Jiang. 2008. "Data Quality Is Context Dependent." *Enabling Real-Time Business Intelligence* 84: 52–67. doi:10.1007/978-3-642-22970-1_5.
- Bianco, Phil, and Rick Kotermanski. 2007. *Evaluating a Service-Oriented Architecture*. Report, CMU/SEI-2007-TR-015. Software Engineering Institute.
- Biggerstaff, T.J. 1989. "Design Recovery for Maintenance and Reuse." *Computer* 22 (7): 36–49. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=30731.
- Bloom, B.S. 1971. *Handbook on Formative and Summative Evaluation of Student Learning*. New York, NY, USA: McGraw-Hill Book Company. <http://eric.ed.gov/?id=ED049304>.
- Bogaty, Sarah. 2013. "Internet Connected Devices Surpass Half a Billion in U.S. Homes, According to The NPD Group." <https://www.npd.com/wps/portal/npd/us/news/press-releases/internet-connected-devices-surpass-half-a-billion-in-u-s-homes-according-to-the-npd-group/>.
- Bondy, J.A., and U.S.R. Murty. 1976. *Graph Theory with Applications*. New York, NY, USA: Elsevier Science Publishing.
- Booth, A., D. Papaioannou, and A Sutton. 2011. "Systematic Approaches to a Successful Literature Review." SAGE.
- Braun, C., F. Wortmann, M. Hafner, and R. Winter. 2005. "Method Construction - a Core Approach to Organizational Engineering." *Proceedings of the 2005 ACM Symposium on Applied Computing*, 1295–99. <http://dl.acm.org/citation.cfm?id=1066971>.
- Brinkkemper, S. 1996. "Method Engineering: Engineering of Information Systems Development Methods and Tools." *Information and Software Technology* 38 (4): 275–80. <http://www.sciencedirect.com/science/article/pii/0950584995010599>.
- Brooke, J. 1996. "SUS-A Quick and Dirty Usability Scale." In *Usability Evaluation in Industry*, 189–94.
- Buckland, M.K., and D. Florian. 1991. "Expertise, Task Complexity, and The Role of Intelligent Information Systems." *Journal of the American Society for Information Science* 42 (9): 635–43. <http://people.ischool.berkeley.edu/~buckland/expertis.pdf>.
- Burleson, B.R. 1984. "Decision-Making Procedure and Decision Quality." *Human Communication Research* 10: 557–74.
- Byrne, B., J. Kling, D. McCarty, and G. Sauter. 2008. "The Value of Applying the Data Quality Analysis Pattern in SOA." *IBM (April 17, 2008)*.
- Calero, C., A. Caro, and M. Piattini. 2008. "An Applicable Data Quality Model for Web Portal Data Consumers." *World Wide Web*. <http://link.springer.com/article/10.1007/s11280-008-0048-y>.

- Cappiello, C., C. Francalanci, and B. Pernici. 2003. "Preserving Web Sites: A Data Quality Approach." *Proceedings of the Eighth International Conference on Information Quality (ICIQ-03) Boston, MA*.
- Carlsson, S.A., and S. Henningsson. 2011. "Socio-Technical IS Design Science Research: Developing Design Theory for IS Integration Management." *Information Systems and E-Business Management* 9 (1): 109–31.
- Carminati, B., E. Ferrari, and P.C.K. Hung. 2005. "Web Service Composition: A Security Perspective." In *Web Information Retrieval and Integration WIR '05*, 248–53. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1553021.
- . 2006. "Security Conscious Web Service Composition." In *International Conference on Web Services ICWS' 06*, 489–96. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4032061.
- Cassell, C., and P. Johnson. 2006. "Action Research: Explaining the Diversity." *Human Relations* 59 (6): 783–814. <http://hum.sagepub.com/content/59/6/783.short>.
- Chandler, D., and R Munday. 2003. *Information Technology. A Dictionary of Media and Communication*. Oxford University Press.
- Channabasavaiah, K, K Holley, and E Tuggle. 2004. *Migrating to a Service-Oriented Architecture. IBM DeveloperWorks*. ftp://129.35.224.15/software/info/openenvironment/G224-7298-00_Final.pdf.
- Comerio, M., and H.L. Truong. 2010. "Service-Oriented Data Quality Engineering and Data Publishing in the Cloud." In *International Conference on Service-Oriented Computing and Applications (SOCA'10)*, 1–6. Perth, WA: IEEE. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5707184.
- World Wide Web Consortium. 2006. "Extensible Markup Language (XML) 1.1." <http://travesia.mcu.es/portaInb/jspui/handle/10421/2507>.
- Coy, W. 2004. "Between the Disciplines." *ACM Special Interest Group on Computer Science Education (SIGCSE) Bulletin*. <http://dl.acm.org/citation.cfm?id=1024340>.
- Debevoise, T. 2007. *Business Process Management with a Business Rules Approach: Implementing the Service Oriented Architecture*. 1st ed. BookSurge Publishing.
- Ejigu, D., M. Scuturici, and L Brunie. 2007. "An Ontology-Based Approach to Context Modeling and Reasoning in Pervasive Computing." *Pervasive Computing and ...*. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4144786.
- English, L. 1999. *Improving Data Warehouse and Business Information Quality: Methods of Reducing Costs and Increasing Profits*. Edited by Inc. John Wiley & Sons. John Wiley & Sons, Inc. New York.
- Eppler, M.J., and P. Muenzenmayer. 2002. "Measuring Information Quality in the Web Context: A Survey of State-of-the-Art Instruments and an Application Methodology." *Seventh International Conference on Information Quality IQ*, Boston, MA, pp.187–96.

- Erl, T. 2005. *Service-Oriented Architecture: Concepts, Technology, and Design*. Prentice Hall, Upper Saddle River, NJ, USA.
- Fisher, CW. 2003. "The Impact of Experience and Time on the Use of Data Quality Information in Decision Making." *Information Systems Research* 14 (2): 170–88..
- Fishman, NA. 2009. *Viral Data in SOA: An Enterprise Pandemic*. IBM Press.
- Foley, Owen. 2011. "Information Quality and Diverse Information Systems Situations," Phd Dissertation, Dublin City University.
- Gackowski, Z.J. 2006. "Redefining Information Quality and Its Measuring: The Operations Management Approach." In *International Conference on Information Quality (ICIQ)*.
- Ge, Mouzhi. 2009. "Information Quality Assessment and Effects on Inventory Decision-Making." Phd Disertation, Dublin City University.
- Geiger, Jonathan G. 2004. *Data Quality Management: The Most Critical Initiative You Can Implement. Data Warehousing, Management and Quality, Paper 098-29*.
- Gill, T.G., and A.R. Hevner. 2013. "A Fitness-Utility Model for Design Science Research." *ACM Transactions on Management Information Systems* 4 (2).
<http://dl.acm.org/citation.cfm?id=2499963>.
- Gladstein, DL. 1984. "Groups in Context: A Model of Task Group Effectiveness." *Administrative Science Quarterly* 29 (4): 499.
- Goel, A. 2006. "Enterprise Integration EAI vs. SOA vs. ESB." *Infosys Technologies White Paper*. http://ggatz.com/images/Enterprise_20Integration_20_20SOA_20vs_20EAI_20vs_20ESB.pdf.
- Goh, Cheng Hian, Stéphane Bressan, Stuart Madnick, and Michael Siegel. 1999. "Context Interchange: New Features and Formalisms for the Intelligent Integration of Information." *ACM Transactions on Information Systems* 17 (3): 270–93.
doi:10.1145/314516.314520.
- Gregor, S., and D. Jones. 2007. "The Anatomy of a Design Theory." *Journal of the Association for Information Systems (JAIS)*. Vol. 8 (5), paper 1
- Gregor, S. 2006. "The Nature of Theory." *Management Information Systems Quarterly* ,30 (3): 611–42.
- Greiffenberg, S. 2004. "Method Engineering in Business and Government." *Dr. Kovac, Hamburg*.
- Hart, C. 2001. "Doing a Literature Search: A Comprehensive Guide for the Social Sciences." SAGE Publications Ltd
- Hassenzahl, M., and N. Tractinsky. 2006. "User Experience - A Research Agenda." *Behaviour & Information Technology* 25 (2): 91–97.

- Helfert, M. 2001. "Managing and Measuring Data Quality in Data Warehousing." *Proceedings of the World Multiconference on Systemics, Cybernetics and Informatics*, 55–65.
- Hevner, Alan R., Salvatore T. March, Jinsoo Park, and Sudha Ram. 2004. "Design Science in Information Systems Research." *MIS Quarterly*. 28 (1). Minneapolis, MN, USA: Society for Information Management and The Management Information Systems Research Center: 75–105.
- Holgado-Terriza, J A, and S Rodr'iguez-Valenzuela. 2011. "Services Composition Model for Home-Automation Peer-to-Peer Pervasive Computing." In *Proceedings of the Federated Conference on Computer Science and Information Systems*, 529-536.
- Holgado-Terriza, J.A., and S. Rodriguez-Valenzuela. 2011. "Services Composition Model for Home Automation Peer-to-Peer Pervasive Computing." In *Computer Science and Information Systems (FedCSIS), 2011 Federated Conference on*, 529–36. Szczecin: IEEE. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6078231.
- Huang, K.T., Y.W. Lee, and R.Y. Wang. 1998. "Quality Information and Knowledge.", Prentice Hall PTR <http://dl.acm.org/citation.cfm?id=288780>.
- Hurteau, M., S. Houle, and S. Mongiat. 2009. "How Legitimate and Justified Are Judgments in Program Evaluation?" *Evaluation* 15 (3): 307–19. <http://evi.sagepub.com/content/15/3/307.short>.
- Iivari, J., and J. Venable. 2009. "Action Research and Design Science Research – Seemingly Similar but Decisively Dissimilar." *17th European Conference on Information Systems*. <http://aisel.aisnet.org/cgi/viewcontent.cgi?article=1025&context=ecis2009>.
- Irmert, F., M. Meyerhöfer, and M. Weiten. 2007. "Towards Runtime Adaptation in a SOA Environment." In *European Conference on Object-Oriented Programming - ECOOP'07*, edited by Michael Cebulla, 17–26. Berlin, Germany: Springer Science..
- ISO. 2015. "Ergonomics of Human-System Interaction (ISO 9241-210)." ISO. http://www.iso.org/iso/catalogue_detail.htm?csnumber=52075.
- Ives, B. 1982. "Graphical User Interfaces for Business Information Systems." *Management Information Systems Quarterly* 6 (4): 15–48. <http://www.jstor.org/stable/248990>.
- Jammes, F., A. Mensch, and H. Smit. 2005. "Service-Oriented Device Communications Using the Devices Profile for Web Services." *21st International Conference on Advanced Information Networking and Applications Workshops, AINAW '07*, pp. 947 - 955, Niagara Falls, Ont.
- Jessup, L.M., J.S. Valacich, and P. Hall. 2008. *Information Systems Today: Managing in the Digital World*. 6th ed. Prentice Hall. https://web.njit.edu/~egan/MIS245/Fall/MIS245_Syllabus_Fall.doc.
- Jeusfeld, M.A., C. Quix, and M. Jarke. 1998. "Design and Analysis of Quality Information for Data Warehouses." *Conceptual Modeling–ER'98*. http://link.springer.com/chapter/10.1007/978-3-540-49524-6_28.

- Jr, J.A. Archibald. 1975. "Computer Science Education for Majors of Other Disciplines." ... 22, 1975, *National Computer Conference and Exposition*.
<http://dl.acm.org/citation.cfm?id=1500154>.
- Kang, D., S. Lee, K. Kim, and J.Y. Lee. 2009. "An OWL-Based Semantic Business Process Monitoring Framework." *Expert Systems with Applications* 36 (4): 7576–80.
<http://www.sciencedirect.com/science/article/pii/S0957417408006805>.
- Kaplan, B, and D Duchon. 1988. "Combining Qualitative and Quantitative Methods in Information Systems Research: A Case Study." *MIS Quarterly*.
<http://www.jstor.org/stable/249133>.
- Kaplan, B., and N.T. Shaw. 2004. "Future Directions in Evaluation Research: People, Organizational, and Social Issues." *Methods of Information in Medicine*, no. 43: 215–31.
- Katerattanakul, P., and K. Siau. 2002. "Information Quality in Internet Commerce Design." *Information and Database Quality*. Vol 25, pp. 45-56
http://link.springer.com/chapter/10.1007/978-1-4615-0831-1_3.
- Kazhamiakin, R., A. Metzger, and M. Pistore. 2008. "Towards Correctness Assurance in Adaptive Service-Based Applications." In *Towards a Service-Based Internet*, 22–37. Madrid: Springer. http://link.springer.com/chapter/10.1007/978-3-540-89897-9_3.
- Kazhamiakin, R., and M. Pistore. 2006. "Choreography Conformance Analysis: Asynchronous Communications and Information Alignment." In *Web Services and Formal Methods*, 227–41. Vienna, Austria: Springer.
http://link.springer.com/chapter/10.1007/11841197_15.
- Kovac, R., and C. Weickert. 2002. "Starting with Quality: Using TDQM in a Start-Up Organization." *Seventh International Conference on Information Quality (IQ 2002)*. Boston, MA, http://ssm-vm030.mit.edu/ICIQ/Documents/IQ_Conference_2002/Papers/StartWithQualityUsingTDQMinaStartupOrg.pdf.
- Krafzig, D., K. Banke, and D. Slama. 2005. *Enterprise SOA: Service-Oriented Architecture Best Practices*. Prentice Hall Professional.
- Kreger, H., V. Brunssen, and R. Sawyer. 2012. "Overview of the SOA Reference Architecture." *The IBM Advantage for SOA Reference Architecture Standards*.
<http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.394.157&rep=rep1&type=pdf>.
- Kutterer, Cornelia. 2012. "Ubiquitous Computing: Serving User Needs Anytime, Anywhere." <http://www.microsoft.com/eu/whats-next/article/ubiquitous-computing-serving-user-needs-anytime-anywhere.aspx>.
- Lee, Y.W. 2003. "Crafting Rules: Context-Reflective Data Quality Problem Solving." *Journal of Management Information Systems*. Vol. 20(3) pp. 93-119
- Lee, Y.W., L.L. Pipino, J.D. Funk, and R.Y. Wang. 2009. "Journey to Data Quality." The MIT Press <http://dl.acm.org/citation.cfm?id=1804429>.

- Lee, Y.W., D.M. Strong, B.K. Kahn, and R.Y. Wang. 2002. "AIMQ: A Methodology for Information Quality Assessment." *Information & Management*. Vol.40(2) 133-146 <http://www.sciencedirect.com/science/article/pii/S0378720602000435>.
- Levis, M., M. Helfert, and M. Brady. 2007. "Information Quality Management: Review of an Evolving Research Area." In *International Conference on Information Quality (ICIQ'07), Boston, MA* <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.84.1748&rep=rep1&type=pdf>.
- Liquid XML. 2015. "Liquid XML WSDL Editor." Accessed November 5. <http://www.liquid-technologies.com/wSDL-editor.aspx>.
- Long, J., and C. Seko. 2002. "A New Method for Database Data Quality Evaluation at the Canadian Institute for Health Information (CIHI)." In *International Conference on Information Quality (ICIQ'02), Boston, MA*. <http://mitiq.mit.edu/ICIQ/Documents/IQConference2002/Papers/ANewMethod4DBDQEvaluationatCIHI.pdf>.
- Loshin, David. 2001. *Enterprise Knowledge Management: The Data Quality Approach*. Morgan Kaufmann.
- . 2009. "Master Data Management." In *Master Data Management*, 94–96. Morgan Kaufmann Publishers.
- Lusk, E.J., and M. Kersnick. 1979. "The Effect of Cognitive Style and Report Format on Task Performance: The MIS Design Consequences." *Management Science* 25 (8): 787–98. <http://pubsonline.informs.org/doi/abs/10.1287/mnsc.25.8.787>.
- Lyytinen, K., and Y. Yoo. 2002. "Ubiquitous Computing." *Communications of the ACM*. http://wiki.daimi.au.dk/OT-intern/_files/cacm-ubicomp.pdf.
- MacKenzie, C.M., K. Laskey, and F. McCabe. 2006. "Reference Model for Service Oriented Architecture 1.0." *OASIS*. <https://www.oasis-open.org/committees/download.php/18486/pr2changes.pdf>.
- Madnick, S., and H. Zhu. 2006. "Improving Data Quality through Effective Use of Data Semantics." *Data & Knowledge Engineering*. Vol 59(2) pp. 460-475 <http://www.sciencedirect.com/science/article/pii/S0169023X05001497>.
- Madnick, S.E. 1995. "Integrating Information from Global Systems: Dealing with the 'on-and Off-ramps'; of the Information Superhighway." *Journal of Organizational Computing*. Vol. 5(2) pp. 69-82
- Manning, Christopher D., and Prabhakar Raghavan. 2009. "An Introduction to Information Retrieval." Edited by A Cannon-Bowers E Salas. *Online*. Cambridge University Press. doi:10.1109/LPT.2009.2020494.
- Marks, E.A., and M. Bell. 2008. *Service Oriented Architecture (SOA): A Planning and Implementation Guide for Business and Technology*. John Wiley & Sons, 2008..

- Martin, P.Y., and B.A. Turner. 1986. "Grounded Theory and Organizational Research." *The Journal of Applied Behavioral Science* 22 (2): 141–57.
<http://jab.sagepub.com/content/22/2/141.short>.
- McCallum, Andrew. 2005. "Information Extraction." *Queue* 3 (9). ACM: 48.
doi:10.1145/1105664.1105679.
- McGraw-Hill. "Dictionary of Scientific & Technical Terms, 6E." Copyright © 2003 by The McGraw-Hill Companies, Inc.
- Mecca, G., P. Atzeni, and A. Masci. 1998. "The Araneus Web-Based Management System." *Proceedings of the 1998 ACM SIGMOD international conference on Management of data*. pp 544-546. <http://dl.acm.org/citation.cfm?id=276375>.
- Mos, A., A. Boulze, S. Quaireau, and C. Meynier. 2008. "Multi-Layer Perspectives and Spaces in SOA." In *Proceedings of the 2nd international workshop on Systems development in SOA environments* pp. 69–74.
<http://dl.acm.org/citation.cfm?id=1370934>.
- Myers, M.D. 1997. "Qualitative Research in Information Systems." *Management Information Systems Quarterly* 21 (2): 241–42.
- Nadkarni, P. 2006. "Delivering Data On Time: The Assurant Health Case." In *International Conference on Information Quality (ICIQ'06), Boston, MA*.
[http://mitiq.mit.edu/ICIQ/Documents/IQ Conference 2006/papers/Delivering Data on Time- The Assurant Health Case.pdf](http://mitiq.mit.edu/ICIQ/Documents/IQ%20Conference%202006/papers/Delivering%20Data%20on%20Time-The%20Assurant%20Health%20Case.pdf).
- Naiman, C.F., and A.M. Ouksel. 1995. "A Classification of Semantic Conflicts in Heterogeneous Database Systems." *Journal of Organizational Computing*, Vol. 5 (2) 163-193 <http://www.tandfonline.com/doi/abs/10.1080/10919399509540248>.
- Nandhakumar, J., M. Rossi, and J. Talvinen. 2005. "The Dynamics of Contextual Forces of ERP Implementation." *The Journal of Strategic Information Systems* 14 (2): 221–42.
<http://www.sciencedirect.com/science/article/pii/S0963868705000284>.
- Naumann, F., and C. Rolker. 2000. "Assessment Methods for Information Quality Criteria." *MIT Conference on Information Quality - IQ*, 148–62. <http://edoc.hu-berlin.de/docviews/abstract.php?id=25314>.
- Newcomer, E., and G. Lomow. 2005. "SOA and Web Services." In *Understanding SOA with Web Services*, 104–35. Addison-Wesley.
http://cds.cern.ch/record/1508600/files/0321180860_TOC.pdf.
- O'Brien, J. A.. 2003. *Introduction to Information Systems: Essentials for the E-Business Enterprise*. Boston, MA: McGraw-Hill.
- O'Brien, L., P. Merson, and L. Bass. 2007. "Quality Attributes for Service-Oriented Architectures." *Proceedings of the International Workshop on Systems Development in SOA Environments*. Minneapolis, MN, <http://dl.acm.org/citation.cfm?id=1270300>.
- OASiS. "Devices Profile for Web Services (DPWS)."

- OASIS. "Web Services Dynamic Discovery (WS-Discovery) Version 1.1." http://docs.oasis-open.org/ws-dd/discovery/1.1/os/wsdd-discovery-1.1-spec-os.html#_Toc234231797.
- Okoli, C., and K Schabram. 2010. "A Guide to Conducting a Systematic Literature Review of Information Systems Research." *Available at SSRN 1954824*.
http://papers.ssrn.com/sol3/Papers.cfm?abstract_id=1954824.
- Oracle. 2008. "Understanding Data Quality Management. In Oracle Warehouse Builder User's Guide."
- Oracle, Apache. 2015. "Reasoners and Rule Engines: Jena Inference Support."
<https://jena.apache.org/documentation/inference/>.
- Orlikowski, WJ, and JJ Baroudi. 1991. "Studying Information Technology in Organizations: Research Approaches and Assumptions." *Information Systems Research* 2 (1): 1–28.
<http://pubsonline.informs.org/doi/abs/10.1287/isre.2.1.1>.
- Ortiz, S. 2007. "Getting on Board the Enterprise Service Bus." *Computer* 40 (4): 15–17.
http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4160214.
- Ostrowski, L. 2012. "Detailed Design Science Research and Its Impact on the Quality of Design Artefacts." *Practical Aspects of Design Science* 286: 60–70.
http://link.springer.com/chapter/10.1007/978-3-642-33681-2_6.
- Ostrowski, L., and M. Helfert. 2012. "Reference Model in Design Science Research to Gather and Model Information." *Americas Conference on Information Systems (AMCIS)*, Seattle. Paper 3.
<http://aisel.aisnet.org/amcis2012/proceedings/SystemsAnalysis/3/>.
- Papazoglou, M.P., and W.J. Van Den Heuvel. 2007. "Service Oriented Architectures: Approaches, Technologies and Research Issues." *The VLDB Journal* 16 (3): 389–415.
<http://link.springer.com/article/10.1007/s00778-007-0044-3>.
- Parra, J, M.A. Hossain, and A Uribarren. 2009. "Flexible Smart Home Architecture Using Device Profile for Web Services: A Peer-to-Peer Approach." *International Journal of Smart Home* 3(2).
- Patton, M.Q. 2015. *Qualitative Research & Evaluation Methods: Integrating Theory and Practice*. Thousand Oaks: Sage Publications.
- Peppers, K., and T. Tuunanen. 2007. "A Design Science Research Methodology for Information Systems Research." *Journal of Management Information Systems* 24 (3): 45–77.
- Peltz, C. 2003. "Web Services Orchestration and Choreography." *Computer*.
<http://www.computer.org/csdl/mags/co/2003/10/rx046.pdf>.
- Pernici, B., and M. Scannapieco. 2003. "Data Quality in Web Information Systems." *Journal on Data Semantics I*. pp. 48-68 http://link.springer.com/chapter/10.1007/978-3-540-39733-5_3.

- Petkov, P., and M. Helfert. 2012. "Data Oriented Challenges of Service Architectures. A Data Quality Perspective." In *Proceedings of the 13th International Conference on Computer Systems and Technologies CompSysTech'12*, 163–70. Ruse: ACM. doi:10.1145/2383276.2383302.
- . 2013a. "A Methodology For Analyzing and Measuring Semantic Data Quality in Service-Oriented Architectures." In *CompSysTech '13 Proceedings of the 14th International Conference on Computer Systems and Technologies*, 201–8. <http://dl.acm.org/citation.cfm?id=2516782>.
- . 2013b. "Developing a Data Quality Methodology in Service Oriented Context Using Design Science Approach." In *Perspectives in Business Informatics Research*, 158:254–66. Springer. doi:10.1007/978-3-642-40823-6_20.
- Petkov, P., Ł. Ostrowski, and M. Helfert. 2013. "Process for Assessment Data Quality in Complex Service Oriented Architectures Using Design Science Approach." In *European Design Science Symposium, EDSS 2012*, 76–87. doi:978-3-319-04090-5.
- Petkov, P., and Rodríguez-Valenzuela, S., M.Helfert. 2013. "Modeling Context-Awareness in a Pervasive Computing Middleware Using Ontologies and Data Quality Profiles." *Evolving Ambient Intelligence Communications in Computer and Information Science* 413: 271–82. http://link.springer.com/chapter/10.1007/978-3-319-04406-4_28.
- Petkov, Plamen, Markus Helfert, and Thoa Pham. 2012. "Discovering Data Quality Issues in Service-Oriented Architectures: A Framework and Case Study." In *BMSD 2012 - Proceedings of the 2nd International Symposium on Business Modeling and Software Design*, 72–80.
- Petkov, P., Rodríguez-Valenzuela, S., M. Helfert. 2013. "Modeling Context-Awareness in a Pervasive Computing Middleware Using Ontologies and Data Quality Profiles." In *Evolving Ambient Intelligence. Communications in Computer and Information Science Volume*, 271–82. Springer.
- Pierce, E.M. 2004. "Assessing Data Quality with Control Matrices." *Communications of the ACM*. Vol. 47(2) <http://dl.acm.org/citation.cfm?id=966395>.
- Pipino, L.L., Y.W. Lee, and R.Y. Wang. 2002. "Data Quality Assessment." *Communications of the ACM*. <http://dl.acm.org/citation.cfm?id=506010>.
- Preuveneers, D., and J. Van den Bergh. 2004. "Towards an Extensible Context Ontology for Ambient Intelligence." *Ambient Intelligence* 3295: 148–59. http://link.springer.com/chapter/10.1007/978-3-540-30473-9_15.
- Rahm, E., and H.H. Do. 2000. "Data Cleaning: Problems and Current Approaches." In *Bulletin of the Technical Committee on Data Engineering*. Vol. 23. IEEE Computer Society. <http://dc-pubs.dbs.uni-leipzig.de/files/Rahm2000DataCleaningProblemsand.pdf>.
- Rapoport, R.N. 1970. "Three Dilemmas in Action Research with Special Reference to the Tavistock Experience." *Human Relations*, 499–513. <http://hum.sagepub.com/content/23/6/499.short>.

- Redman, T.C. 2001. "Data Quality: The Field Guide." Digital Press.
- Ross, P.H., M.W. Ellipse, and H.E. Freeman. 2004. *Evaluation: A Systematic Approach*. 7th ed. Thousand Oaks: Sage.
- Sangwan, R., M. Bass, N. Mullick, DJ Paulish, and J. Kazmeier. 2006. *Global Software Development Handbook*. CRC Press.
- Satyanarayanan, M. 2001. "Pervasive Computing: Vision and Challenges." *Personal Communications, IEEE*. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=943998.
- Sauro, J. 2011. "Measuring Usability with the System Usability Scale (SUS)." <http://www.measuringu.com/sus.php>.
- Scacchi, W. 1982. "The Web of Computing: Computer Technology as Social Organization." *Advances in Computers* 21: 2–90.
<http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.296.7103&rep=rep1&type=pdf>.
- Scannapieco, M., A. Virgillito, and C. Marchetti. 2004. "The DaQuinCIS Architecture: A Platform for Exchanging and Improving Data Quality in Cooperative Information Systems." *Information Systems*.
<http://www.sciencedirect.com/science/article/pii/S0306437904000031>.
- Schroder, H.M., M.J. Driver, and S. Streufert. 1967. *Human Information Processing*. New York: Holt, Rinehart and Winston.
https://scholar.google.com/scholar?q=schroder+human+information+processing&btnG=&hl=bg&as_sdt=0%2C5#0.
- Shanks, G.G., and P. Darke. 1998. "Understanding Data Quality and Data Warehousing: A Semiotic Approach." *The Australian Computer Journal* 30: 122–28. [http://ssm-vm030.mit.edu/ICIQ/Documents/IQ Conference 1998/Papers/UnderstandDQinDataWarehouseSemioticAppr.pdf](http://ssm-vm030.mit.edu/ICIQ/Documents/IQ%20Conference%201998/Papers/UnderstandDQinDataWarehouseSemioticAppr.pdf).
- Shaw, M.E. 1971. *Group Dynamics: The Psychology of Small Group Behavior*. New York: McGraw-Hill.
<http://profiles.wizfolio.com/DongMyungKim/publications/18458/134511/>.
- Singh, R., and K. Singh. 2010. "A Descriptive Classification of Causes of Data Quality Problems in Data Warehousing." *International Journal of Computer Science Issues IJCSI '10* 7 (3). <http://www.ijcsi.org/papers/7-3-2-41-50.pdf>.
- Stake, R.E. 2013. *Multiple Case Study Analysis*. Guilford Press..
- Stamper, Ronald, Kecheng Liu, Mark Hafkamp, and Yasser Ades. 2000. "Understanding the Roles of Signs and Norms in Organizations - a Semiotic Approach to Information Systems Design." *Behaviour & Information Technology* 19 (1): 15–27.
doi:10.1080/014492900118768.
- Stojanović, Z., and A. Dahanayake. 2005. "Service-Oriented Software System Engineering: Challenges and Practices." IGI Publishing Hershey, PA, USA

- Strong, D.M., Y.W. Lee, and R.Y. Wang. 1997. "Data Quality in Context." *Communications of the ACM*. Vol. 40(5) pp. 103-110 <http://dl.acm.org/citation.cfm?id=253804>.
- Su, Y., and Z. Jin. 2006. "A Methodology for Information Quality Assessment in the Designing and Manufacturing Process of Mechanical Products." *Information Quality Management: Theory and Applications: Theory and Applications*. Idea Group Inc (IGI)
- Taylor, M.S. 1981. "The Motivational Effects of Task Challenge: A Laboratory Investigation." *Organizational Behavior and Human Performance*, no. 27: 255–78. <http://www.sciencedirect.com/science/article/pii/0030507381900490>.
- TechTarget. 2010. "Research State of SOA Survey for 2010." In . Forrester.
- The Open Group. 2015a. "The SOA Open Group." <http://www.opengroup.org/subjectareas/soa>.
- . 2015b. "SOA Reference Architecture Technical Standard : Overview of the SOA RA Layers." https://www.opengroup.org/soa/source-book/soa_refarch/layers.htm.
- McCue, T.J. 2013. "24 Electronic Products Per Household -- Got Recycling?" <http://www.forbes.com/sites/tjmccue/2013/01/02/24-electronic-products-per-household-got-recycling/>.
- Urquhart, C. 2010. "Putting the 'Theory'back into Grounded Theory: Guidelines for Grounded Theory Studies in Information Systems." *Information Systems Journal* 20 (4): 357–81. <http://onlinelibrary.wiley.com/doi/10.1111/j.1365-2575.2009.00328.x/full>.
- Venable, J., J. Pries-Heje, and R. Baskerville. 2012. "A Comprehensive Framework for Evaluation in Design Science Research." *Design Science Research in Information Systems. Advances in Theory and Practice* 7286: 423–38. http://link.springer.com/chapter/10.1007/978-3-642-29863-9_31.
- W3C. 2014. "Web Services Eventing (WS-Eventing)." <http://www.w3.org/Submission/WS-Eventing/#Introduction>.
- Wahli, U. 2006. "Web Services Handbook for Websphere Application Server Version 6.1." In *IBM Redbooks*. IBM International.
- Wand, Y., and R.Y. Wang. 1996. "Anchoring Data Quality Dimensions in Ontological Foundations." *Communications of the ACM*. Vol. 39(11) pp. 86-95 <http://dl.acm.org/citation.cfm?id=240479>.
- Wand, Yair, and Richard Y. Wang. 1996. "Anchoring Data Quality Dimensions in Ontological Foundations." *Communications of the ACM* 39 (11). New York, NY, USA: ACM: 86–95. doi:10.1145/240455.240479.
- Wang, R.Y., Y.W. Lee, L.L. Pipino, and D.M. Strong. 1998. "Manage Your Information as a Product." *Sloan Management Review*. <http://elibrary.ru/item.asp?id=4836769>.
- Wang, R.Y., V.C. Storey, and C.P. Firth. 1995. "A Framework for Analysis of Data Quality Research." *IEEE Transactions on Knowledge and Data Engineering* Vol 7(4) pp 623-640 http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=404034.

- Wang, R.Y., and D.M. Strong. 1996. "Beyond Accuracy: What Data Quality Means to Data Consumers." *Journal of Management Information Systems*. Vol 12(4) pp 5-33
<http://www.jstor.org/stable/40398176>.
- Wang, R.Y., M. Ziad, and Y.W. Lee. 2001. "Data Quality." Springer Science & Business Media.
- WC3. 2015. "WC3: Log Files." In *World Wide Web Consortium*.
http://publib.boulder.ibm.com/tividd/td/ITWSA/ITWSA_info45/en_US/HTML/guide/c-logs.html#combined.
- Webster, J., and R.T. Watson. 2002. "Analyzing the Past to Prepare for the Future: Writing a Literature Review." *Management Information Systems Quarterly*. Vol26(2) pp.13-23
<http://aisel.aisnet.org/cgi/viewcontent.cgi?article=2625&context=misq>.
- Weiser, M. 2002. "The Computer for the 21st Century." *IEEE Pervasive Computing*. Vol 1(1) pp. 19-25 <http://dl.acm.org/citation.cfm?id=2212302>.
- White, S.A. 2004. "Introduction to BPMN." *IBM Cooperation*. http://www.ebm.nl/wp-content/uploads/2010/05/Introduction_to_BPMN.pdf.
- Wren, J.D. 2008. "URL Decay in MEDLINE—a 4-Year Follow-up Study." *Bioinformatics*.
<http://bioinformatics.oxfordjournals.org/content/24/11/1381.short>.
- Yin, R.K. 1994. "Discovering the Future of the Case Study Method in Evaluation Research." *Evaluation Practice*.
<http://www.sciencedirect.com/science/article/pii/088616339490023X>.
- Zhai, Y., J. Zhang, and K.J. Lin. 2009. "SOA Middleware Support for Service Process Reconfiguration with End-to-End QoS Constraints." In *International Conference on Web Services ICWS' 09*, 815–22. Los Angeles, CA: IEEE.
http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5175901.

(All web links are tested 'accessible' circa August 15, 2016)

APPENDIX A : SOA SURVEY

SOA Survey Questions

The targets of this survey are Business, Technical and Enterprise architects and developers who are using SOA or having SOA project underway. This survey is a part of a research project on a topic "Analysing and measuring data quality in Service Oriented Architectures" and the results of it will significantly contribute to the research output. Data collected from this survey will be used only for research purposes. The result will be published afterwards, once they have been processed.

The completion of this survey would require no more than 10-15 minutes. Various responses from the same institution are acceptable.

Thank you for your time!

Section I: Organisational Information

Name of your company		
Your position at your company		
Your experience at the taken position	years	
In which industry does your institution operate? (Please check. Multiple choices are available)	Banking	
	Finance/Accountancy	
	Education	
	Transportation/Logistics	
	Healthcare	
	other	
... if 'other' please specify:		
May the information you provide be shared?		

Section II: SOA Challenges

What is the most substantial challenge you are confronting with your SOA project?

Statement	Score 1 – minor 5 – substantial
Evaluate and select appropriate services	
How to integrate services	
Selecting SOA environment and designing business processes	
Dealing with data integration issues	
Efficient Security and Communication	
Developing ,Testing and Deployment of services	
other(please specify)	

Do you encounter any data quality misadventures in your SOA project?

- ☐ Yes
☐ No

If yes please specify which is/are the most commonly occurred one/s:

Type of issue	Occurrence 1 – less 5 – most
Inaccurate data (i.e. data values do not correctly reflect real-world conditions)	
Incomplete data (i.e. data does not fully describe the real-world object)	
Inconsistent data (i.e. data metrics, scales, units and measures are not mapped correctly)	
Outdated data (i.e. data provided from the services is out of date)	
Redundant/duplicated data (i.e. exactly the same information is representative by multiple services)	
Missing data (i.e. information about the object does not persist)	
other(please specify)	

Do you have monitoring tool integrated into your project?

- ☐ Yes
☐ No

If yes, what does the tool monitor?

- ☐ **business processes**
☐ **technical data** *(i.e. cpu load, memory usage etc.)*
- ☐ **policies and rules**
☐ **business data** *(i.e. customers, addresses, etc.)*
- ☐ **other (please specify)**

Does monitoring in your company operate on a real-time basis or daily basis?

- ☐ **real-time**
☐ **none**
- ☐ **daily**

If you monitor business data, when do you perform integrity, validity, etc. checks?

- ☐ **before data is used by target service/s**
- ☐ **after data is consumed by the service/s**

Does the monitoring tool you use meet the business and technical requirements e.g. monitoring capabilities, ease of use, reliability, etc.?

- ☐ **Yes, it meets the all requirements**
- ☐ **Yes, it meets the requirements but lack of some functionality/ies**
- ☐ **No, it meets just partly the requirements and it significantly lacks of features**
- ☐ **No, it does not meet the requirements at all**
- In case of option 2 or 3 please briefly describe the requirement/s the tool you use does not meet and functionality/ies it lacks:**

Section III: SOA Technologies

Are any of the following products part of your service portfolio or SoA framework?

ESB solutions		SOA governance/management		SOA suites	
JBoss ESB		Oracle Service Registry		Oracle SoA suite	
WSO2 ESB		Kuali Service Registry		Red Hat JBoss Enterprise SOA Platform	
Mulesoft		Kuali Service Registry		WSO2 Suite	
Apache ServiceMix		IBM Websphere Service Registry		IBM Websphere	
IMB Webshere ESB		other(refer to the next question)		Microsoft Biztalk	

Kuali Service Bus			TIBCO ActiveMatrix 3.0	
other(refer to the next question)			other(refer to the next question)	

If Other SOA related products
please comment on any other products that you are implementing or have implemented.

How widely used are the following SoA technology standards in your architecture?
(1 = Not used, not important; 5 = Widely used, very important)

SOAP		BPEL		WS-Security		JSON	
REST		SAML		WS-Trust		WS-Transaction	
POX (Plain Old XML)		XML binding: JAXB, JAX-WS		WS-Policy		others	

Please briefly explain one SOA-related initiative at your institution in the last 3 years.

Is there anything else you would like to share about SOA at your institution?

APPENDIX B : SUS SURVEY TOOL

System Usability Scale Survey

1. I think that I would like to use DQ tool frequently

1	2	3	4	5

2. I found the DQ tool unnecessarily complex

1	2	3	4	5

3. I thought the DQ tool was easy to use

1	2	3	4	5

4. I think that I would need the support of a technical person to be able to use this tool

1	2	3	4	5

5. I found the various functions in DQ tool were well integrated

1	2	3	4	5

6. I thought there was too much inconsistency in this tool

1	2	3	4	5

7. I would imagine that most people would learn to use this tool very quickly

1	2	3	4	5

8. I found the system very cumbersome to use

1	2	3	4	5

9. I felt very confident using the system

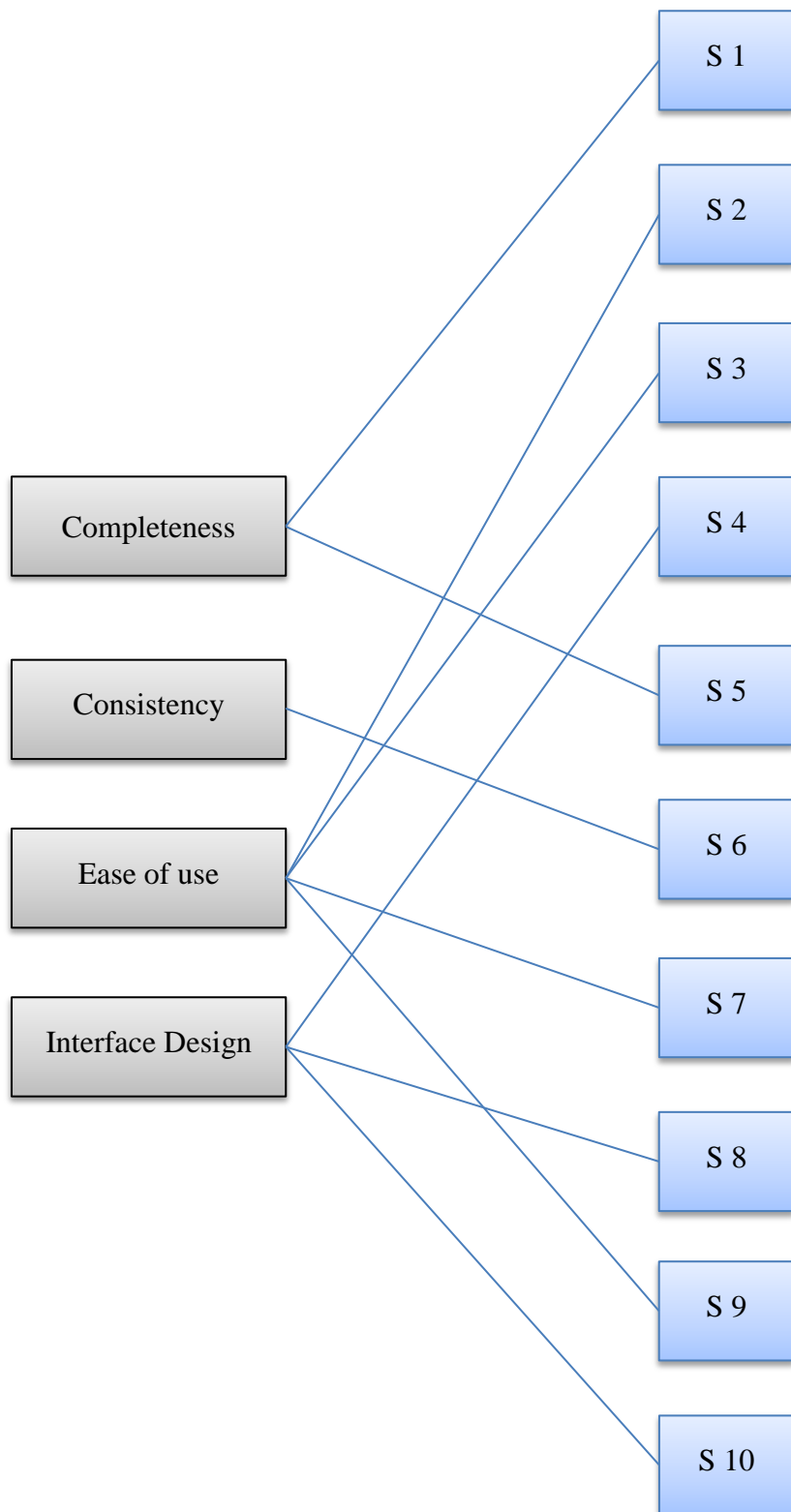
1	2	3	4	5

10. I needed to learn a lot of things before I could get going with the DQ tool

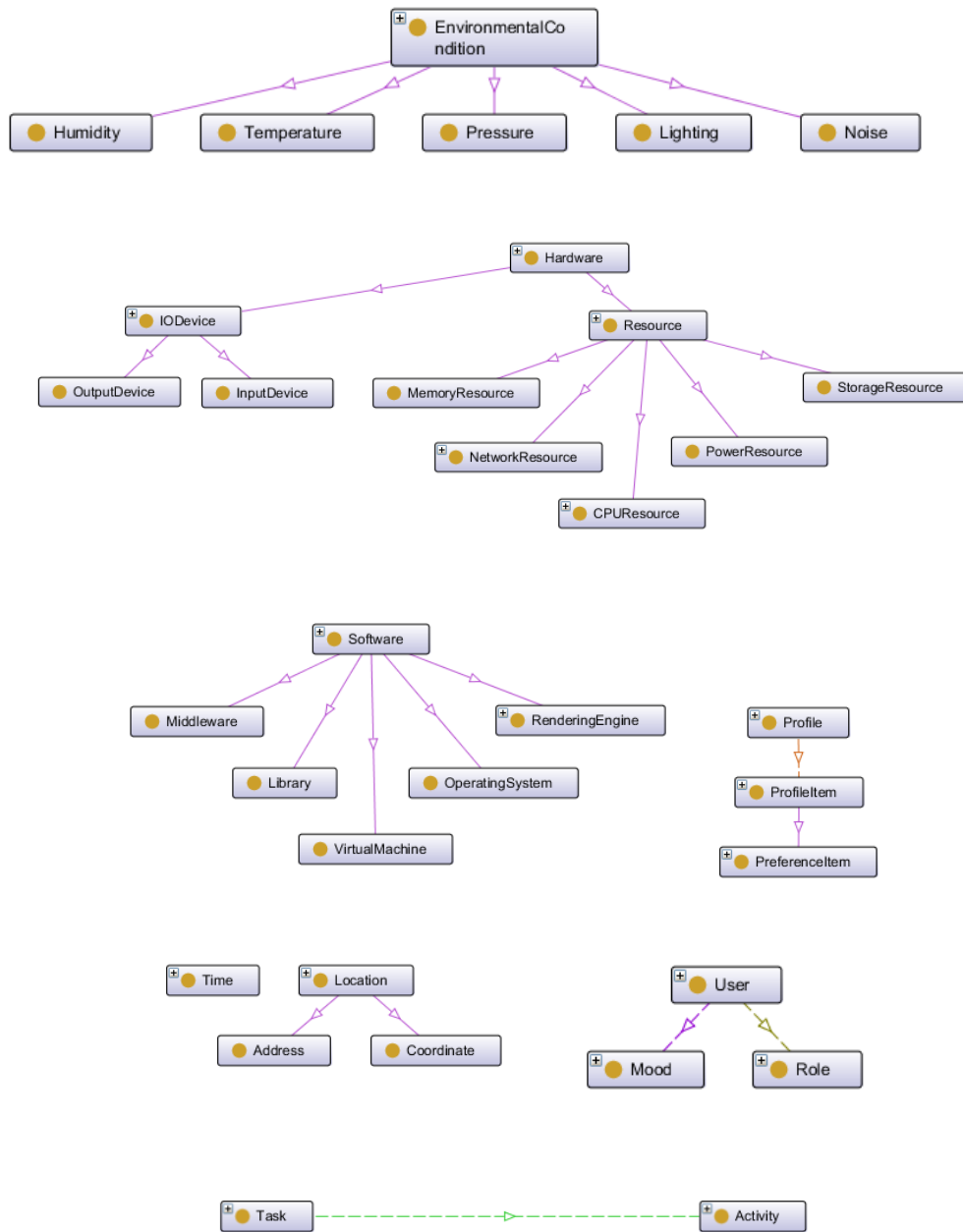
1	2	3	4	5

Please provide your comments about the DQ tool and process and a short description about your choice for each of the SUS statements. Why and what make you take your decision? Please elaborate in the separate blank sheet provided for you.

Evaluation Criteria to SUS Statements



APPENDIX C : CODAMOS ONTOLOGY



APPENDIX D : SERVICE CODE LISTING (PARTIAL)

Simple Service operation

D:\gSC_BIG\DOHA-
improvement\Implementation\ServiceShutters\src\ugr\sc\shutters\OperationSetGetShutterPosition.java

```
1 package ugr.sc.shutters;
2
3 import doha.Param;
4 import doha.Service;
5 import doha.SimpleOperation;
6 import java.util.ArrayList;
7 import java.util.logging.Level;
8 import java.util.logging.Logger;
9
10 /**
11  *
12  * @author plamen
13  */
14 public class OperationSetGetShutterPosition extends SimpleOperation {
15
16     int shutPosition; // Shutter position in percents - 0 open; 100 closed
17
18     public OperationSetGetShutterPosition(Service s) {
19         // super(s);
20
21         Param pout = new Param("posOut", "OUTPUT");
22         this.setpOUT(pout);
23
24         ArrayList<Param> aux = new ArrayList();
25         Param pin = new Param("posIn", "INPUT");
26         aux.add(pin);
27         this.setpIN(aux);
28     }
29
30     @Override
31     public String invoke() {
32         try {
33             Thread.sleep(1500);
34         } catch (InterruptedException ex) {
35             Logger.getLogger(OperationSetGetShutterPosition.class.getName())
36                 .log(Level.SEVERE, null, ex);
37         }
38         //retrieve Shutter position (in persents) 0 open; 100 closed
39         if (this.getParamIN("posIn").getValue() == null) {
40             shutPosition = 40; //default value 40% closed
41         } else {
42             shutPosition = Integer.parseInt(this.getParamIN("posIn").getValue());
43             if (shutPosition > 100) shutPosition = 100; //100% is max - fully closed
44         }
45
46         System.out.println("Shutters position: "+shutPosition+"%");
47         this.getParamOUT().setValue(String.valueOf(shutPosition));
48         return String.valueOf(shutPosition);
49     }
50 }
```

Simple service initialization

D:\gSC_BIG\DOHA-improvement\Implementation\ServiceShutters\src\ugr\sc\shutters\main.java

```
1 package ugr.sc.shutters;
2
3 import doha.Operation;
4 import doha.Service;
5 import java.util.ArrayList;
6
7 public class main {
8
9     /**
10      * @param args
11      */
12     public static void main(String[] args) throws InterruptedException {
13
14         // Initialization serviceSimple
15         Service serviceSimple = Service.getServiceFactory(Service.DPWS);
16         serviceSimple.setName("ServiceShutters");
17         serviceSimple.setId("0053");
18
19         OperationSetGetShutterPosition op = new
20         OperationSetGetShutterPosition(serviceSimple);
21         op.setName("OperationSetGetShutterPosition");
22         serviceSimple.addOp(op);
23
24         serviceSimple.initservice();
25
26     }
27 }
28
```

Composite service operation

D:\gSC_BIG\DOHA-

improvement\Implementation\ServiceAmbientLightControl\src\ugr\sc\ambientLightControl\OperationAmbientLightControl.java

```
1 package ugr.sc.ambientLightControl;
2
3 import doha.CompositeOperation;
4 import doha.Operation;
5 import doha.Param;
6 import doha.Service;
7 import java.util.ArrayList;
8 import java.util.logging.Level;
9 import java.util.logging.Logger;
10 import java.io.File;
11 import java.io.IOException;
12 import java.text.SimpleDateFormat;
13 import java.util.Calendar;
14 import java.util.Date;
15 import jxl.*;
16 import jxl.write.*;
17
18 /**
19 *
20 * @author plamen
21 */
22 public class OperationAmbientLightControl extends CompositeOperation {
23
24     public OperationAmbientLightControl() {
25
26         Param pOut = new Param("lumensOut", "OUTPUT");
27         this.setpOUT(pOut);
28
29         ArrayList<Param> aux = new ArrayList();
30         Param pIn = new Param("lumensIn", "INPUT");
31         aux.add(pIn);
32         this.setpIN(aux);
33     }
34
35     @Override
36     public String invoke() {
37         Service s = Service.getService(); //get singleton service
38         s.getS_interaction().searchServices(); //search for published services
39
40         int rowShut = 0, rowBulb = 0, rowLum = 0;
41         WritableWorkbook workbook;
42         String currentLuminance = null;
43         int currentLuminanceInt = 0;
44         String userLuminance = null;
45         int userLuminanceInt;
46         String setLuminance = null;
47         String bulbPower = null;
48         int bulbPowerInt = 0;
49         String bulbType;
50         String shutterPos = null;
51         int shutterPosInt = 0;
52
53         ArrayList<Param> aux = new ArrayList();
54         Param p;
55
56         //LOAD USER PREFS
57         userLuminance = this.getParamIN("lumensIn").getValue();
58         userLuminanceInt = Integer.parseInt(userLuminance);
59
60         try {
61             String timeStamp = new SimpleDateFormat("yyyyMMdd_HH:mm:ss").format(Calendar.getInstance().getTime());
62             System.out.println("creating file: " + timeStamp);
63             workbook = Workbook.createWorkbook(new File(timeStamp + ".xls"));
64             WritableSheet sheet = workbook.createSheet("Data Sheet", 0);
65         }
```

```

66 sheet.addCell(new Label(0, 0, "Luminance level (lum)")); //A1
67 sheet.addCell(new Label(1, 0, "Shutters position (%")); //B1
68 sheet.addCell(new Label(2, 0, "Bulb Powers (W)")); //C1
69 sheet.addCell(new Label(3, 0, "Time Stamp (HH:mm:mm)")); //A1
70
71 if (s.getS_interaction().searchService("ServiceBrightnessSensor", "0002")) {
72     //GET CURENT AMBIENT LIMINANCE
73     currentLuminance = s.getS_interaction().invokeOperation("ServiceBrightnessSensor", "0002", "OperationGetBrightness", null);
74     currentLuminanceInt = Integer.parseInt(currentLuminance);
75
76     //GET CURENT SHUTTER POSITION
77     shutterPos = s.getS_interaction().invokeOperation("ServiceShutters", "0053", "OperationSetGetShutterPosition", null);
78     shutterPosInt = Integer.parseInt(shutterPos);
79
80
81     while (currentLuminanceInt < userLuminanceInt) {
82         if (shutterPosInt != 0) {
83             shutterPosInt -= 5; // decrees the angle by 5% e.g. opens the sutter
84             p = new Param("posIn", "INPUT", String.valueOf(shutterPosInt));
85             aux.add(p);
86             shutterPos = s.getS_interaction().invokeOperation("ServiceShutters", "0053", "OperationSetGetShutterPosition", aux);
87             jxl.write.Number cell = new jxl.write.Number(1, 1 + rowLum, Integer.parseInt(shutterPos));
88             sheet.addCell(cell);
89             cell = new jxl.write.Number(2, 1 + rowLum, 0);
90             sheet.addCell(cell);
91             currentLuminanceInt += 100; //increase brightness by 100 lumens
92         } else if (bulbPowerInt != 100) { //turn on the bulbs the bulbs
93             bulbPowerInt += 10; //turn on incandescent bulb on 10 w power
94             p = new Param("bulbIn", "INPUT", String.valueOf(bulbPowerInt));
95             aux.add(p);
96             bulbPower = s.getS_interaction().invokeOperation("ServiceLightBulb", "0052", "OperationSetLightBulbPower1", aux);
97             jxl.write.Number cell = new jxl.write.Number(2, 1 + rowLum, Integer.parseInt(bulbPower));
98             sheet.addCell(cell);
99             cell = new jxl.write.Number(1, 1 + rowLum, 0);
100             sheet.addCell(cell);
101             currentLuminanceInt += 50; //increase brightness by 200 lumens
102         } else {
103             System.out.println("It cant be more bright");
104             break;
105         }
106         aux.clear();
107
108         p = new Param("setSensor", "INPUT", String.valueOf(currentLuminanceInt));
109         aux.add(p);
110         currentLuminance = s.getS_interaction().invokeOperation("ServiceBrightnessSensor", "0002", "OperationGetBrightness", aux);
111         jxl.write.Number cell = new jxl.write.Number(0, 1 + rowLum, Integer.parseInt(currentLuminance));
112         sheet.addCell(cell);
113         timeStamp = new SimpleDateFormat("HH:mm:ss").format(Calendar.getInstance().getTime());
114         Label label = new Label(3, 1 + rowLum, timeStamp);
115         sheet.addCell(label);
116         rowLum++;
117         aux.clear();
118         System.out.println("End of cycle");
119     }
120
121     while (currentLuminanceInt > userLuminanceInt) {
122         if (shutterPosInt != 100) {
123             shutterPosInt += 5; // increase the angle by 5% e.g. closes the shutter
124             p = new Param("posIn", "INPUT", String.valueOf(shutterPosInt));
125             aux.add(p);
126             shutterPos = s.getS_interaction().invokeOperation("ServiceShutters", "0053", "OperationSetGetShutterPosition", aux);
127             jxl.write.Number cell = new jxl.write.Number(1, 1 + rowLum, Integer.parseInt(shutterPos));
128             sheet.addCell(cell);
129             cell = new jxl.write.Number(2, 1 + rowLum, 0);
130             sheet.addCell(cell);
131             currentLuminanceInt -= 100; // decrees brightness by 100 lumens
132         } else if (bulbPowerInt != 100) {
133             bulbPowerInt += 10; //turn on incandescent bulb on 10 w power

```



```

134         p = new Param("bulbIn", "INPUT", String.valueOf(bulbPowerInt));
135         aux.add(p);
136         bulbPower = s.getS_interaction().invokeOperation("ServiceLightBulb", "0052", "OperationSetLightBulbPower1", aux);
137         jxl.write.Number cell = new jxl.write.Number(2, 1 + rowLum, Integer.parseInt(currentLuminance));
138         sheet.addCell(cell);
139         cell = new jxl.write.Number(1, 1 + rowLum, 0);
140         sheet.addCell(cell);
141         currentLuminanceInt += 50; //increase brightness by 200 lumens
142
143     } else {
144         System.out.println("It cant be more bright");
145         break;
146     }
147     aux.clear();
148     p = new Param("setSensor", "INPUT", String.valueOf(currentLuminanceInt));
149     aux.add(p);
150     currentLuminance = s.getS_interaction().invokeOperation("ServiceBrightnessSensor", "0002", "OperationGetBrightness", aux);
151     jxl.write.Number cell = new jxl.write.Number(0, 1 + rowLum, Integer.parseInt(currentLuminance));
152     sheet.addCell(cell);
153     timeStamp = new SimpleDateFormat("HH:mm:ss").format(Calendar.getInstance().getTime());
154     Label label = new Label(3, 1 + rowLum, timeStamp);
155     sheet.addCell(label);
156     rowLum++;
157     aux.clear();
158     System.out.println("End of cycle 2");
159 }
160
161 System.out.println("Result getLuminance:" + currentLuminance);
162 this.getParamOUT().setValue(String.valueOf(currentLuminanceInt));
163 // All sheets and cells added. Now write out the workbook
164 workbook.write();
165 workbook.close();
166 } else {
167     currentLuminance = "0";
168     System.out.println("Service: ServiceBrightnessSensor is unavailable at the moment");
169     System.out.println("Composite Operation Control ambient Light cannot be executed at the moment");
170     this.getParamOUT().setValue(String.valueOf(currentLuminance));
171 }
172
173 return setLuminance;
174
175 } catch (IOException ex) {
176     Logger.getLogger(OperationAmbientLightControl.class.getName()).log(Level.SEVERE, null, ex);
177     return null;
178 } catch (WriteException ex) {
179     Logger.getLogger(OperationAmbientLightControl.class.getName()).log(Level.SEVERE, null, ex);
180     return null;
181 }
182 }
183 }
184

```