A FRAMEWORK FOR QUALITATIVE ASSESSMENT OF DOMAIN SPECIFIC LANGUAGES

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
THE MIDDLE EAST TECHNICAL UNIVERSITY

BY

GÖKHAN KAHRAMAN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

SEPTEMBER 2013

Approval of the thesis:

A FRAMEWORK FOR QUALITATIVE ASSESSMENT OF DOMAIN SPECIFIC LANGUAGES

submitted by **GÖKHAN KAHRAMAN** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen _____
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Gönül Turhan Sayan _____
Head of Department, **Electrical and Electronics Engineering**

Prof. Dr. Semih Bilgen _____
Supervisor, **Electrical and Electronics Engineering Dept.**

Examining Committee Members

Assoc. Prof. Dr. Cüneyt Bazlamaçcı _____
Electrical and Electronics Engineering Dept., METU

Prof. Dr. Semih Bilgen _____
Electrical and Electronics Engineering Dept., METU

Assoc. Prof. Dr. Ece Güran Schmidt _____
Electrical and Electronics Engineering Dept., METU

Prof. Dr. Ali Doğru _____
Computer Engineering Dept., METU

Prof. Dr. Hayri Sever _____
Computer Engineering Dept., Hacettepe University

Date: 03.09.2013

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name	:	Gökhan KAHRAMAN
Signature	:	

ABSTRACT

A FRAMEWORK FOR QUALITATIVE ASSESSMENT OF DOMAIN SPECIFIC LANGUAGES

Kahraman, Gökhan
Ph.D., Department of Electrical and Electronics Engineering
Supervisor: Prof. Dr. Semih Bilgen

September 2013, 76 pages

Domain Specific Languages(DSLs)have been proposed in the literature with the aim of providing a wide range of advantages such as better productivity and quality for stakeholders involved with many aspects of software development. The objective of this study consists of supporting the improvement of DSL maturity by providing an A Framework for Qualitative Assessment of DSLs(FQAD). A formal approach is proposed for the assessment of DSLs. Metrics for DSL success measurement and the parameters that affect these metrics are determined. FQAD is constructed and FQAD process is defined in a step by step manner. A number of assessment paths in terms of levels are proposed, which are intended to be used for decision support when determining the success of the DSL.A multiple-case study consisting of two cases is conducted in order to mature and validate the proposed FQAD method.

Keywords: Domain Specific Languages, Metamodel, Software Generation, Success Assessment.

ÖZ

ALANA ÖZGÜ DİLLERİN NİTELİKSEL DEĞERLENDİRİLMESİ İÇİN BİR ÇERÇEVE

Kahraman, Gökhan
Doktora, Elektrik Elektronik Mühendisliği Bölümü
Tez Yöneticisi: Prof. Dr. Semih Bilgen

Eylül 2013, 76 sayfa

Alana Özgü Diller (AÖD) belirli bir alandaki yazılım problemlerini çözmeyi amaçlar. Literatürde AÖD, daha yüksek üretkenlik elde edilmesi, kalitenin artması, bakım ihtiyacının azalması ve kullanılabilirliğin yükselmesi amaçlarıyla önerilmiştir. Bu ilerlemelerin elde edilebilmesi için AÖD'lerin uygulama ortamına uygun olduğunun ve kalitesinin yüksek olduğunun belirlenmesi gereklidir. Bu çalışmanın amacı AÖD'ler için bir etkinlik değerlendirme çatısı (FQAD) oluşturarak alana özgü dillerin olgunluk seviyesinin iyileştirilmesine katkıda bulunmaktır. AÖD'lerin değerlendirilmesi için formal bir yaklaşım sunulmaktadır. Önce, literatürdeki programlama dillerinin etkinliğinin değerlendirilmesi çalışmaları, değerlendirme ölçütlerine göre sınıflandırılmış, ardından etkinlik, ölçüm ve yazılım modelleme teknolojileri kavramlarını oluşturan yapıları içeren teorik çatı sunulmuştur. AÖD etkinlik değerlendirmesi için metrikler ve bu metrikleri etkileyen parametreler belirlenmiştir. FQAD işleyişi adım adım tanımlanarak FQAD önerilmiştir. Değerlendirme modeli içinde, AÖD'lerin etkililiğini değerlendirirken karar vermeye yardımcı olmak amacıyla, seviyelerle ifade edilen değerlendirme yolları önerilmiştir. Durum çalışmaları planlanmış ve durum çalışmaları sırasında izlenecek yöntemler tasarlanmıştır. İki farklı durumu içerecek bir çoklu örnek durum çalışması yapılarak önerilen FQAD olgunlaştırılmış ve doğrulanmıştır.

Anahtar Kelimeler: Alana Özgü Diller, Metamodel, Yazılım Üretimi, Etkinlik Değerlendirme, Kalite Ölçütleri.

dedicated to my family

ACKNOWLEDGMENTS

First of all, I would like to express my sincere gratitude to my supervisor Prof. Dr. Semih Bilgen for his guidance, insightful suggestions and comments throughout the whole PhD process and the development of this thesis. I also appreciate deeply his never ending support and positive attitude.

I wish to express my sincere thanks to my committee members Assoc. Prof. Dr. Cüneyt Bazlamaçcı and Prof. Dr. Ali Doğru for their comments and suggestions over the last three years.

I would like to thank my superiors at ASELSAN for their support and tolerating me time for the Phd. studies.

Finally, I am grateful to my parents and especially my wife, she gave me confidence, I knew that she was always there; ready to support me in any possible means.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

BWW: Bunge-Wand-Weber

CMM: Capability Maturity Model

CMMI: Capability Maturity Model Integration

DSM: Domain Specific Modeling

DSL: Domain Specific Languages

FQAD: Framework for Qualitative Assessment of DSLs

GMF: Graphical Modeling Framework

GPL: General Purpose Languages

IEC: International Electrotechnical Commission

ISO: International Standards Organization

ORMSC: Object and Reference Model AB Subcommittee

PL: Programming Language

SEI: Software Engineering Institute

CHAPTER 1

INTRODUCTION

Domain Specific Languages (DSLs) are used for improving productivity and quality of the software development process, but the measurement of those improvements and whether and to what extent DSLs provide desired benefits are important issues that must be addressed (Karna et al. 2009). Gabriel (2010) has compiled a survey on the assessment of DSLs and points out the absence of a systematic assessment of the DSLs. The growing number and complexity of DSLs raise the necessity of a systematic approach for assessment of DSLs (Kelly and Tolvanen2008; Strembeck and Zdun 2009).

Definition of the term "high-quality" in the context of DSLs is critical for the success of the usage of DSLs in software development process. This study focuses on the important challenge within DSLs of how to determine the specific quality characteristics and apply these characteristics in the DSL assessment process. A number of hierarchical assessment paths are proposed, which are intended to be used as decision support when determining the success of a DSL. The success of a DSL is a cluster of related characteristics in a DSL, which, when owned collectively, satisfy a goal considered important for the DSL.

The Framework for Qualitative Assessment of DSLs (FQAD) is useful for not only assessing the end result of the DSL development process, the language, but also aids DSL developers on determining the required quality characteristics at the outset of DSL development. Concentrating only on the needed characteristics instead of trying to develop a DSL with all-characteristics satisfied reduces the necessary effort and enhances quality.

Despite the importance of having effective DSLs, the quality of a DSL is an in-progress concept. The existing research evaluates how well DSLs perform in use and lists desirable or undesirable properties that can be found in "good" or "bad" DSLs. What distinguishes the approach presented in this study from various others (Kolovos et al. 2006; Haugen et al. 2007; Merilinna and Parsinnen 2007; Kahlaoui et al. 2008; Hermans et al. 2009; Karna et al. 2009; Kelly and Pohjonen 2009; Kosar et al. 2010; Wu et al. 2010; McKean and Sprinkle 2012) is that FQAD first determines the perspective of the evaluator by eliciting assessment goals and then selects DSL assessment characteristics that generally conform to the international systems and software quality standard,

ISO/IEC 25010:2011, yielding an assessment model for decision support. The aim of this study is to assess the DSL, in order that the DSL meet stakeholders' goals, by increasing their satisfaction.

We also present two industrial case studies where FQAD is applied and evaluated in the context of two DSLs developed in different software development departments in ASELSAN which is a high technology defense industry company in Ankara, Turkey, with its most distinctive expertise in the fields of design, development, production and system integration of hardware/software systems. It must be noted at the outset that as with any research study involving case studies (Runeson and Host 2009), neither our claim, nor our goal is to establish a universally applicable theory, but rather, to formulate well-developed and validated hypotheses to form the foundation for possible future studies, either application-oriented and to be carried out by industrial organizations for decision making, or theoretical and to be carried out by future researchers on DSL quality or success.

1.1. The Scope of the Study

A number of studies have addressed specific quality characteristics for DSLs. The studies (Kolovos et al. 2006; Haugen et al. 2007; Merilinna and Parsinnen 2007; Kahlaoui et al. 2008; Hermans et al. 2009; Karna et al. 2009; Kelly and Pohjonen 2009; Kosar et al. 2010; Wu et al. 2010; McKean and Sprinkle 2012) on DSL evaluation adopt an approach similar to ours in that they all derive from relevant literature on computer languages assessment, but none of them considers the evaluation perspective explicitly. The focus of those studies are on specific technical issues, whereas we aim to assess what is relevant for different stakeholders.

Comparison type studies (Merilinna and Parssinen 2007; Kosar et al. 2010; Kelly and Tolvanen 2008) provide valuable results pointing out the advantages and disadvantages of using DSLs over other approaches. Comparison or combined evaluation of stakeholders' viewpoints is intentionally avoided in our study, as we emphasize the need as well as ability to address separate, and possibly even contradictory viewpoints.

To create and work with DSLs various tools which aim to reduce additional efforts to design languages have been developed. These tools that generate modeling tools are named as Meta-Case tools and popular ones include: MetaEdit+, Obeo Designer, GMF (Kouhen et al. 2012). Although the quality of a DSL development tool affects the resulting DSL, we prefer to assess the characteristics of DSLs independently from the tools that may have been used to generate them, so as to focus on the success of the languages proper. We leave the evaluation of tools out of the scope of this study.

In this study, a Framework for Qualitative Assessment of DSLs is proposed. This comprehensive framework adapts and integrates the ISO/IEC 25010:2011 standard, CMMI maturity level evaluation approach (SEI 2012) and the scaling approach used in DESMET (Kitchenham et al. 1997) into a perspective-based model.

The maturity level evaluation approach used in CMMI (SEI 2012) framework is adapted to assess DSLs in this study. The focus of capability maturity models (CMMs) is on improving processes in an organization. CMMs provide the essential elements of effective processes and describe the improvement path with improved quality. Specific and generic goals are defined as the rating elements in CMMI according to which goals are rated using evidence recorded against each specific and generic practice. We take the same approach and define DSL characteristics as the rating elements and use DSL sub-characteristics for rating DSL characteristics. We define DSL success as a combination of related characteristics in a DSL, which, when possessed collectively, satisfy the evaluator's goal.

In our approach, success of a DSL relates the goals of a DSL to the completeness with which these goals can be achieved. Hence, for a DSL, the measure of success is the level of completion of the sub-characteristic which means that the expected results are obtained from the assessment of the sub-characteristics. Assessment does not take into account how DSL sub-characteristics (sub-goals) are achieved; only the extent to which they are achieved is assessed.

DSL characteristics consist of both objective and subjective characteristics. Although subjective characteristics mostly address satisfaction, those characteristics are also important in the assessment of the effectiveness of the DSL.

While defining DSL success levels, DESMET approach is used and levels defined in (Kitchenham et al. 1997) are adapted to cover the concepts in the DSL assessment domain.

The effectiveness assessment approach presented in (Cameron 1980) is adopted in the present study. In Cameron (1980) the effectiveness of an organization is investigated. It is stated that evaluating the effectiveness of organizations requires restricting assessment to a set of appropriate criteria selected in an a-priori fashion. Such an assessment provides a basis for inevitable trade-offs. Hence, we start by capturing the evaluators' perspective on the success of a DSL.

1.2. The Objective of the Study

The objective of this study is to propose a framework that provides quality characteristics and sub-characteristics for DSLs that conform with ISO/IEC 25010 standard and literature, and provides a qualitative method for the measurement of the success level of the DSLs according to different perspectives. The two research questions are:

1. What are the quality characteristics of DSL success?
2. How can an evaluator measure the success of a DSL using measures aligned with stakeholder perspectives?

1.3. Contributions

This research makes the following contributions:

1. Presents a comprehensive model for DSL success assessment using quality characteristics balanced according to an evaluator perspective.
2. Unifies and brings together the experiences from the fields of DSL assessment, information systems assessment, and software product quality.
3. Elaborates a detailed list of domain specific language quality characteristics and proposes a novel assessment method.
4. Extends previous work on domain specific language success assessment by focusing on the ISO/IEC 25010 standard software product characteristics and extending and adapting those characteristics for DSL assessment.
5. The two in-depth case studies provide rich insight into the DSL quality field.
6. For managers, domain specific language developers and decision makers who often face domain specific language success assessment, they would benefit from the research deliverables through a deep understanding of the quality characteristics related with domain specific language.

1.4. Thesis Outline

The remainder of the thesis is organized as follows. Chapter 2 sets the theoretical background. Chapter 3 describes FQAD and outlines the DSL assessment process. Chapter 4 includes a detailed description of the research approach and the research design. The case studies are presented in Chapter 5. The main findings from the questionnaire-based evaluation of FQAD case studies are also presented in Chapter 5. Finally, Chapter 6 presents a conclusion on the findings and identifies issues for further research.

CHAPTER 2

LITERATURE REVIEW

This chapter provides a review of the relevant literature from the specific viewpoint of the present study, and is divided into three sections. The first section consists of a literature review concerning programming language (PL) assessment and measurement issues. The second section focuses on the subject of domain specific languages (DSLs) and reviews related work on DSL assessment. The third section provides the foundations of the framework for qualitative assessment of DSLs (FQAD); ISO/IEC 25010:2011, software engineering institute's software capability maturity model (CMM), organizational effectiveness and DESMET methodology.

2.1. Programming Language Assessment

According to Hoare (1973) the primary purpose of a PL is to help the programmer in the practice of his art. There are many desirable properties of a PL. Collectively, the level at which a PL possesses those properties can be called its success. Success is related to the answer to the question of "what" a PL does. Before assessment of a PL we must answer the question "what" and come to an agreement on the goals to achieve.

2.1.1. Early Work On Programming Language Design and Assessment

Design guidelines for PLs have been intensively discussed since the early 70s. For the design of good languages Hoare (1973) introduced simplicity, security, fast translation, efficient object code, and readability as general principles. Furthermore, Wirth (1974) discussed several guidelines for the design of languages and corresponding compilers. According to Watt (2004) a PL must be universal, natural for expressing computations in its intended application area and implementable. The main idea behind most of the guidelines can be accepted as still valid today, but the technical opportunity has changed since the 70s. Computer power has increased significantly. Therefore, speed and space problems have become less important (Karsai et al., 2009).

Hoare's seminal book (1973) is one of the most important sources on PL design. It was written in 1973 but general principles are still very pertinent. Three most difficult tasks in programming that a PL should support are listed as: program design, programming documentation and program debugging. Principles Hoare lists are simplicity, security, fast translation, efficient object code and readability. Simplicity is very important because programmers are reluctant to learn new languages.

One of the concentrated areas in the early research into evaluation of PLs was the methodological approaches used in the PL evaluation process. Brooks (1980) reviews the methodological problems that will be encountered during the PL constructs and techniques evaluation. Tactics suggested to address these problems deal with the major methodological issues. In methodology selection the fundamental issues are given as subjects, materials and measures. When employing a methodology to find out the effect of PL construct or technique, researcher has some hypothesis. For observing and measuring the behavioral effect of this hypothesis, a situation must be constructed. Researcher must select appropriate subjects, manage the materials to evoke an experimental effect and use appropriate measures to create such situations. The essential problem in the evaluation of PL constructs and technique is the strong interaction between the characteristics of the programmer and those of the program. The difficulty in writing, modifying or understanding a program may be related to programmer's knowledge level with that type of program or the intrinsic characteristics of the program itself.

Most early attempts at assessing PLs centered on software psychology. Software psychology is described in Curtis et al. (1986) as the study of human factors in computer systems. This review presents the methodological issues involved in the application of experimental research for the study of human factors in computer systems. Human computer interaction can be described under different topics according to their impact on the performance of programmers in developing computer systems as:

- *Variation among individual participants in programming systems*
  The wide scope of variation among individual participants is one of the most important problems while studying the human factors in programming. Studies in Sackman et al. (1968) show a 28:1 range in performance. According to Boehm (1981) team and personnel capability differences are the most significant factor affecting programming productivity.
- *Problem solving in unstructured domains*
  For well-defined problems with finite solution states optimal path to the solution can easily be defined. But for domains like programming which are semantically rich, there is no clearly defined solution state. For this reason, problem solving in programming is a different task that must be handled during the research process.
- *The design (use) of PLs*
  Human-computer interaction languages can be characterized according to their necessity to write programs in the range from machine level to natural language level. The details of integrating software with hardware are hidden as one uses languages closer to natural languages. When natural-like languages are used for programming, computers are more accessible to people because users do not need to understand low-level programming details. One of the studies in this area is by Miller (1974), about natural language usage by people who are not familiar with a computer language. It is stated that a vocabulary of 100 words would have been sufficient to express the specifications that have been collected from 84 protocols where totally 610 unique words are used.

- *Knowledge representation in programmers*
  Programmer's maturation is directly related to the observed patterns and blocks built by the programmer. Some of the studies in this area analyze the long-term memory and short-term memory relationship. Experience and education affect the knowledge base in long term memory. The integration of knowledge from several different domains is required to develop skill in writing programs. These different domains of knowledge may require years of training and experience to master. With this perspective one can be a talented specific domain programmer and still be a novice at programming in a different specific domain.
- *Program comprehension and text comprehension relationship*
  A program is also a text. Programs can be understood and executed by a computer, and can be understood and modified by other programmers. So it is easy to state that there is a parallelism between program comprehension and natural language text understanding. Since text has been around longer, text research is more developed than programming research. Thus text researches can be used for programming. Software metrics used in programming are the equivalent to the readability formulas used in text research.
- *The terminology used for commands*
  Choosing the terminology for commands is an important issue in PLs. Studies that investigate how various factors affect learnability and usability of commands have shown that verbs are the best command names that clearly discriminate among the operations performed.

2.1.2. Programming Language Evaluation Methods and Criteria

Every PL has its own strengths and weaknesses. The reason for choosing a particular language may ultimately be based on factors that are not related with the technical merits of the language (Naiditch, 1999).

Chen et al. (2005) have studied PL trends. This study tries to predict the evolution of the software engineering technology. Two kind of factors are determined affecting this evolution; first the factors that can be used to describe the general design criteria of PLs are listed as: generality, orthogonality, reliability, maintainability, efficiency, simplicity, machine independence, implementability, extensibility, expressiveness and influence or impact , second the factors that are not directly related to PLs but still can have effects are explained as: institutional, industrial, governmental, organizational, grassroots and technology supports.

PL evaluation criteria are classified in Howatt (1995) under 4 major categories.

- *Language Design and Implementation Criteria* are used to assess language design and how easily compilers can be written for it.
- *Human Factors Criteria* are used to assess the human interface of a language. The user friendliness of a language is important so that programmer can easily and correctly code a program. Process of learning a language is also important. This process must be easy since programming is human-intensitive.

- *Software Engineering Criteria* are used to assess the language's ability to support portability, reliability and all other engineering concerns.
- *Application Domain Criteria* are used to assess how well a language supports for programming for specific domain of applications.

In their study Furuta and Kemp (1979) examine the effects of the programmer-PL interaction on the maintainability, understandability and reliability of the resulting programs. Several researches on this topic are performed with naive subjects raising the question about the applicability of the results to the professional subjects. Experimental research of PLs is a difficult task because of the problems involved. Controlled experiments provide more accurate and reliable results but are difficult to generalize. Two types of studies are examined:

- *The comparison of entire languages*: Coding a benchmark problem in the compared languages and then comparing the resulting programs according to the certain criteria is the approach used to evaluate several languages. The languages are so wide and measurement techniques are so broad. For this reason it is difficult to decide which language performs better.
- *The comparison of language constructs (e.g. conditional branching statement, assignment operator)*: Determination of which feature will enhance and which will degrade program maintainability, readability and reliability is important. This kind of comparison experiments and their results are specific to the conditions of the experiment. To generalize these results they must be examined carefully.

Rombach (1990) proposes a framework for assessing process representation languages. Four dimensions are defined for language assessment:

- *Assessment Objectives*: What are the issues of the process being assessed? What is the assessment purpose?
- *Language that is Assessed*: What are the language features being assessed?
- *Assessment Context*: How - in what context - is the assessment being performed?
- *Results of the Assessment*: What are the results of the assessment wrt. each assessment objective?

Kulkarni et al. (2008) present a survey on PLs; C++, Perl, Lisp and Java. That study involves a comparative study of these languages with respect to the following parameters like: reusability, portability, reliability, readability, efficiency, availability of compilers and tools, familiarity and expressiveness.

Parker et al. (2006) aim to develop a method for selecting PLs for introductory courses. The strategy followed consists of following steps:

- Compile a list of language selection criteria
- Weight each of the criteria. Ask each evaluator to weight, specific to the department's needs, the value of importance for each criterion.

- Determine a list of candidate languages. The list should be comprised of languages nominated by the faculty rather than a complete list of available languages.
- Evaluate the language. Each candidate language should be assigned a rating for each criterion.
- Calculate weighted score. For each candidate language, a weighted score can be calculated by adding together the language score multiplied by the weight assigned to each criterion. The language with the highest weighted score is the optimal choice, given the evaluators' assessments.

The criteria used in Parker et al. (2006) is derived by investigating papers relevant to language selection. Used criteria can be listed as: "*Reasonable Financial Cost, Availability of Student/Academic Version, Academic Acceptance, Availability of Textbooks, Stage in Life Cycle, Industry Acceptance, Marketability (Regional and National), System Requirements of Student/Academic/Full Version, Operating System Dependence, Proprietary/Open Source, Development Environment, Debugging Facilities, Ease of Learning Fundamental Concepts, Support for Secure Code, Advanced Features for Subsequent Programming Courses, Scripting or Full-Featured Language, Support of Web Development, Supports Target Application Domain, Teaching Approach Support, Object-Oriented Support, Availability of Support, Instructor and Staff Training, Anticipated Experience Level for Incoming Students".*

2.1.3. Conclusion On Programming Language Assessment Review

Many steps toward the PL assessment have been taken but the journey is still in progress. While there exists a significant amount of advice in the form of lists of potential criteria or possible criteria dimensions, there is very little research available to state which criteria are appropriate for the various circumstances.

PL evaluations can be performed before, after or during the PL development. The number of involved factors for the evaluation is very large and requires a multiple viewpoint approach. Selection of the criteria for PL evaluation can be made easier with the aid of organizing this large number of factors according to different viewpoints.

2.2. Domain Specific Languages

In this section we first define DSLs, comparing them with general purpose languages (GPLs). Then after reviewing the relationship between models and languages we detail the constructs of DSLs and present the related work on DSL assessment.

2.2.1. DSLs vs. GPLs

Programming languages can be categorized in two groups: GPLs and DSLs.

GPLs aim to solve any software problem regardless of the domain. They are general and widely used, so these languages are accepted by the majority of programmers. Availability of well tested optimizing compilers and the existence of good development environments are crucial for this preference. On the other hand, GPLs also have many drawbacks like writing and reading, or understanding the programs. A lot of programming expertise is needed for good programming otherwise everyone is not able to use them properly. Different programming techniques which aim to raise the abstraction level are not enough to overcome the difficulties observed in comprehending GPL programs (Oliveira et al., 2009).

DSLs aim to solve software problems in a particular domain. The term DSL is defined in Mernik et al. (2005) as languages tailored to a specific application domain. DSL's claim better expressiveness, ease of use and more productivity in comparison to GPLs (Kosar et al., 2009). The development of a DSL is hard and requires both knowledge in the domain and expertise in the language development.

2.2.2. Models and Languages

There are many definitions for models. According to Mellor (2004) models are the combination of sets of elements that describe some physical, abstract or hypothetical reality.

In model driven architecture, a model is defined as a representation of a part of the function, structure and/or behavior of a system (ORMSC, 2001).

Among these definitions Benyon's (1997) definition "A model is a representation of something, constructed and used for a particular purpose" is detailed in Overbeek (2006).

Modelers construct model for a particular purpose to represent something. Interpreters use models for a particular purpose. In this view model is the representation of something. But a model has no meaning on its own. Information in the model can be understood if model is combined with an interpretation. Extracting the correct meaning from the model can be done only if a common understanding of the concepts between modeler and the interpreter is established.

This common understanding leads us to the *languages*.

2.2.3. Constructs of DSLs

A domain is a field of study characterized by a set of concepts and terminology understood by practitioners and is bounded relative to the problem at hand (Sánchez-Ruíz et al., 2006).

Instead of aiming to be best for any domain, DSLs focus on one particular domain and solve a specific class of problems in that domain. The constructs and abstraction of the domain are used within the language which makes domain knowledge explicit. Using DSLs it is possible to express the solution of the problems in the domain with a little effort.

The idea of DSLs is presumably as old as the notion of programming languages (Mernik et al. 2005). Some widely used DSLs are: Excel macro language (Spreadsheets), HTML (Hypertext web pages), LATEX (Typesetting), Make (Software building), SQL (Database queries), VHDL (Hardware design).

DSLs are also called: Application-oriented lang. , Special-purpose languages, Task-specific languages, Problem-oriented languages, 4GL, Specialized languages, Little (mini) languages, End-user languages.

Strembeck and Zdun (2009) describe the main DSL artifacts and their relations. A DSL has a concrete syntax and a language model. The concrete syntax is an interface for the language model. The language model provides the definitions of the language elements and consists of three sub-models: core language model, language model constraints and behavior specification. The core language model expresses domain abstractions and specifies the relations between them. The language model constraints (static semantics) define semantics that cannot be captured in the core language model. The DSL behavior specification (dynamic semantics) defines the behavioral effects that result from using a DSL language element. DSLs are defined to specify elements of the target domain.

DSLs are usually expressed as text or graphic diagrams; however representations such as matrices, tables, forms or even trees can also be used (Kelly and Pohjonen, 2009;Allen et al., 2005).

Fowler (2010) handles DSLs in two different styles that can be distinguished with regard to the implementation approach of the DSL;

- An internal DSL (also called as embedded DSL) is defined as an extension to an existing GPL and uses the host language as its base (e.g. Ruby DSLs (Freeze,2006), Haskel example (Hudak1996)). An embedded DSL can directly access all host language's constructs and infrastructure including tools, libraries, frameworks, or other platform-specific components. Thus, it is not necessary to build a new generator for an embedded DSL. The compiler or interpreter of the existing language are used as DSL generator.
- An external DSL is defined in a different format than the host language of the application and transformed into it using some form of a compiler (e.g. Microsofts' OSLO framework (Microsoft, 2008), examples of graphical DSLs

presented in Kelly and Tolvanen (2008)). An external DSL can use all kinds of language constructs. The advantage of external DSLs is that designers of the DSL may define any possible syntax independent from the syntactical particularities of a given host language.

Strembeck and Zdun (2009) define four main activities in the DSL development process. These activites consist of sub-activities:

1. Define the core language model of the DSL: Identify domain abstraction, add domain abstraction to language model, define language model constraints, check language model.
2. Define the DSL language elements' behavior: Select language model element(s), define language model element(s) behavior, check DSL behavior
3. Define the concrete syntax(es) of the DSL: Define symbols for language model elements, define DSL production/composition rules, define DSL concrete syntax
4. Integrate DSL artifacts with the platform/infrastructure Map DSL artifacts to platform features, extend/adapt platform, define DSL-to-platform transformations, test DSL, check integrated DSL

These development activities can be performed in different orders. The order may be determined according to the size of the DSL or project.

2.2.4. Related Work On DSL Assessment

A number of studies have addressed specific quality characteristics for DSLs. Haugen et al. (2007) present a structured questionnaire based on three dimensions of a DSL - expressiveness, transparency, formalization. Merilinna and Parssinen (2007) investigate the benefits of using DSLs by making comparisons between the DSL approach and traditional approaches experimentally. Kosar et al. (2010) report an experiment that investigates the difference between general-purpose programming language and DSL program understanding, concluding that success rate for programmers is %15 better for DSL. Hermans et al. (2009) identify a number of success factors, perform an empirical study using the proposed questionnaire and evaluate the success factors with a case study. Wu et al. (2010) propose an approach to determine the effort in using DSLs during application development using quantitative measurement. After the classification of the effort, they propose effort related metrics. Kolovos et al. (2006) list the core quality requirements for a DSL. Kahlaoui et al. (2008) emphasize domain specificity and code generation ability as specific requirements on DSLs. Based on real DSL development cases Kelly and Pohjonen (2009) discuss worst practices for creating domain specific modeling (DSM) languages which developers should avoid. Karna et al. (2009) evaluate the DSM solution in a real case. Focusing on the productivity and usability of the DSM solution, they first determine the objectives for the creation of the DSM and then collect data via controlled laboratory studies in their approach. McKean and Sprinkle (2012) present criteria that will help in selecting a DSM or any other approach to use in system development. Gabriel (2010) presents a systematic review on evaluation of DSLs emphasizing the reduced concern on this subject and focuses on the evaluation of DSLs based on usability engineering concerns.

Comparison type studies (Merilinna and Parssinen 2007; Kosar et al. 2010; Kelly and Tolvanen 2008) provide valuable results pointing out the advantages and disadvantages of using DSLs over other approaches. But those studies use only one perspective (e.g. usability, productivity) to assess the DSLs. In our study we assess DSLs from a wider perspective and define the success of the DSL in parallel with the evaluator goals.

To create and work with DSLs various tools which aim to reduce additional efforts to design languages have been developed. These tools that generate modeling tools are named as Meta-Case tools and popular ones include: MetaEdit+, Obeo Designer, GMF (Kouhen et al. 2012). Although the quality of a DSL development tool affects the resulting DSL, we prefer to assess the characteristics of DSLs independently from the tools that may have been used to generate them, so as to focus on the success of the languages proper. We leave the evaluation of tools out of the scope of this study.

Considering the process of conceptual modeling, Wand and Weber (1993, 1995) created a quality framework based on Bunge's (1977) ontology and it is known as the Bunge–Wand–Weber (BWW) framework. An ontological evaluation is based on two mappings (Wand and Weber 1993): first, a representation mapping describes the mapping of the constructs of the BWW-model onto the grammatical constructs. Second, the interpretation mapping describes the mapping of the grammatical constructs onto the constructs of the BWW-model. Four ontological deficiencies can be identified: incompleteness, redundancy, excess, overload (Wand and Weber 1993).

The studies (Kolovos et al. 2006; Haugen et al. 2007; Merilinna and Parsinnen 2007; Kahlaoui et al. 2008; Hermans et al. 2009; Karna et al. 2009; Kelly and Pohjonen 2009; Kosar et al. 2010; Wu et al. 2010; McKean and Sprinkle 2012) on DSL evaluation adopt an approach similar to ours in that they all derive from relevant literature on computer languages assessment, but none of them considers the evaluation perspective explicitly. The focus of those studies are on specific technical issues, whereas we aim to assess what is relevant for different stakeholders.

2.3. Foundations

In this dissertation, a Framework for Qualitative Assessment of DSLs is proposed. This comprehensive framework adapts and integrates the ISO/IEC 25010:2011 standard, capability maturity model integration (CMMI) maturity level evaluation approach (SEI,2012) and the scaling approach used in DESMET (Kitchenham et al. 1997) into a perspective-based model.

2.3.1. ISO/IEC 25010

Since it is the most recent and mature international standard on the quality model of software-intensive computer systems and software products, in this study the terminology used in ISO/IEC 25010:2011 is followed. To determine DSL quality characteristics, ISO/IEC 25010:2011 standard is used together with the computer language assessment literature. ISO/IEC 25010:2011 standard revises ISO/IEC 9126:2001 and provides a model for software quality with well-formed definitions for

quality characteristics of software products. This quality model categorizes software product quality into characteristics which can be further subdivided into sub-characteristics. This multilevel hierarchy provides a convenient breakdown of software product quality. Making use of the hierarchical model of ISO 25010, success of a DSL is expressed in terms of DSL quality characteristics and sub-characteristics.

The ISO/IEC 25010 international standard defines two models for system and software quality:

1. Quality in use model: This model defines five characteristics related to the usage of the product in a particular context and its interaction with human beings. It can be applied to both computer systems in use and software products in use.
2. Product quality model: This model defines eight characteristics related to the static and dynamic properties of the computer systems. It can be applied to both to computer systems and software products.

Both models are composed of characteristics and sub-characteristics which are expressed in a hierarchical structure.

Quality in use model is composed of five characteristics: effectiveness, efficiency, satisfaction, freedom from risk, and context coverage (Table 1) which shows the user's view of the quality of a software product as a result of using the software product.

Table 1. Quality in use model (from ISO/IEC 25010)

Quality in Use				
Effectiveness	**Efficiency**	**Satisfaction**	**Freedom from risk**	**Context Coverage**
Effectiveness	Efficiency	Usefulness	Economic risk mitigation	Context completeness
		Trust	Health and safety risk mitigation	Flexibility
		Pleasure	Environmental risk mitigation	
		Comfort		

The focus of the present study is on the quality product model, which categorizes product quality properties into eight characteristics: Functional suitability, Performance efficiency, Compatibility, Usability, Reliability, Security, Maintainability, and Portability. Each characteristic is further decomposed in related sub-characteristics (Table 2).

Table 2. Product Quality model (from ISO/IEC 25010)

System/Software Product Quality						
Functional Suitability	Functional completeness	Functional correctness	Functional appropriateness			
Performance Efficiency	Time behavior	Resource utilization	Capacity			
Compatibility	Co-existence	Interoperability				
Usability	Appropriateness recognizability	Learnability	Operability	User error protection	User interface aesthetics	Accessibility
Reliability	Maturity	Availability	Fault tolerance	Recoverability		
Security	Confidentiality	Integrity	Non-repudiation	Accountability	Authenticity	
Maintainability	Modularity	Reusability	Analysability	Modifiability	Testability	
Portability	Adaptability	Installability	Replaceability			

ISO/IEC 25010 standard gives the definitions of the software quality (sub) characteristics. However, these definitions are given in a generic sense. In this study the quality product model of ISO/IEC 25010 standard is used and the (sub) characteristics are tailored (e.g. a redefinition or refinement of definitions of quality characteristics, the definition of additionally needed quality (sub) characteristics) using the concepts of the DSL assessment domain and giving the rationale for any changes.

2.3.2. CMMI

The highly popular maturity level evaluation approach used in the CMMI (SEI, 2012) framework is adapted to assess the success level of DSLs in this study. CMMs focus on improving processes in an organization. They provide the essential elements of effective processes and describe the improvement path with improved quality and success. Specific and generic goals are defined as the rating elements in CMMI according to which goals are rated using evidence recorded against each practice (specific and generic). We take the same approach and define DSL characteristics as the rating elements and use DSL sub-characteristics for rating DSL characteristics. We define DSL success as a combination of related characteristics in a DSL, which, when possessed collectively, satisfy the evaluator's goal.

The CMM was developed by the Software Engineering Institute (SEI) a part of Carnegie Mellon University. SEI defines a CMM as a reference model of mature practices in a specified discipline. This model is used to improve and appraise an organization's capability to put into practice that discipline (Caputo, 1998). CMMs were developed for many disciplines, e. g. for systems engineering. CMMs differ by structure, discipline and definitions of the capability and maturity. These different structures emerged a need to integrate the CMMs to avoid confusions especially when using more than one model. At that point the CMM Integration project was formed to build a set of integrated models and to establish a framework for integration of future models. There are four

disciplines available in CMMI: software engineering, systems engineering, integrated product and process development, supplier sourcing.

All CMMI models identify process areas. The process areas are used to cover concepts that are important to process improvement in any area of interest.

CMMI uses levels to support two improvement paths. One path associated with the "capability levels" enables organizations to incrementally improve processes corresponding to an individual process area selected by the organization. The other path associated with the "maturity levels" enables organizations to improve a set of related processes by incrementally addressing successive sets of process areas.

These levels are called "representations". The two representations are named as "continuous" and "staged". Using the continuous representation enables organizations to achieve "capability levels". Using the staged representation enables organizations to achieve "maturity levels".

A capability level defines a set of practices that must be present and applied for a certain process area to obtain the respective capability level.

The four capability levels are explained below:

- **Incomplete**: An incomplete process means that process is either not performed or partially performed. For an incomplete process, one or more of the specific goals of the process area are not satisfied and no generic goals exist for this level.
- **Performed**: A performed process is a process that accomplishes the needed work to produce work products; the specific goals of the process area are satisfied.
- **Managed**: A managed process means that a performed process is planned and executed according to the policy; skilled people are employed having adequate resources to produce controlled outputs; relevant stakeholders are involved; is monitored, controlled, and reviewed; and is evaluated for adherence to its process description.
- **Defined**: A defined process means that a managed process is tailored from the organization's set of standard processes according to the organization's tailoring guidelines; has a maintained process description; and contributes process related experiences to the organizational process assets.

CMMI models are organized into five maturity levels. Each maturity level describes a stage in the process improvement of an organization.

- **Level 1 - Initial**: Organizations at this level are characterized by working in an ad hoc manner and poorly controlled. On this level, success of the projects depends on the efforts of the people.
- **Level 2 - Managed**: This level is concerned with management. This level require to manage requirements, processes, work products, and services. This

level also ensures that the practices continues to be applied even there is time pressure.

- **Level 3 - Defined**: At the "defined level" the aim is the standardization of processes. Processes in a project are derived from the organizational standards. This facilitates to ensure the consistency among all process within the organization.
- **Level 4 - Quantitatively Managed**: This level introduces quantitative data for process performance. The aim is to attain a quantitative control over the processes.
- **Level 5 - Optimizing**: The quantitative predicted results are checked against the business objectives. This level focuses on continuous process improvement.

Except for Level 1, each maturity level consists of several process areas. Process areas are the indications of the areas that an organization should focus on to improve its process. To reach a certain maturity level within the CMM, each of the process areas of that level and lower levels have to be implemented by the organization. Moreover, for a process area to be considered implemented each of the goals of the process area should be reached. A process area includes goals and practices. An organization that implements all practices from a certain process area is expected to reach the goals of that process area.

2.3.3. DESMET

In the present study, while defining success levels, the DESMET approach is used and levels defined in Kitchenham et al. (1997) are adapted to cover the concepts in the DSL assessment domain. FQAD is also evaluated using DESMET methodology.

DESMET is a project which aims to develop and validate a method for evaluating software engineering methods and tools. DESMET methodology presents evaluation methods and a set of criteria to help the evaluator choose the most appropriate one for his needs (Kitchenham et al. 1997)

In DESMET methodology evaluation is performed in a comparative way. It is assumed that there are several alternative ways of performing a task and the aim is to identify which of the alternatives is best in specific circumstances. In DESMET, evaluations are context-dependent, which means that as a result of the evaluation it is not expected a specific method to be the best in all circumstances. This means that an evaluation in one company would result in one method being identified as best, but a similar evaluation in another company would come to a different result.

The DESMET Methodology identifies nine distinct evaluation methods. These are:

1. *Quantitative experiment* investigates the quantitative impact of tools by designing a formal experiment.
2. *Quantitative case study* investigates the quantitative impact of tools by designing a case study.
3. *Quantitative survey* investigates the quantitative impact of tools by designing a survey.

4. *Qualitative screening* is a feature-based evaluation. The evaluation is performed by a single individual who determines the features to be assessed and their rating scale.
5. *Qualitative experiment* is a feature-based evaluation. The evaluation is done by a group of potential user. The user group is expected apply the tools on typical scenarios before making their assessment.
6. *Qualitative case study* is a feature-based evaluation. The evaluation is performed someone who has used the tools on a real project.
7. *Qualitative survey* is a feature-based evaluation. The evaluation is done by people who used the tool. Discretion of the subject to participate in a survey is the the difference between a survey and an experiment.
8. *Qualitative effects analysis* is based on expert opinion. The evaluation is a subjective assessment of the quantitative effect of methods and tools.
9. *Benchmarking* is a process of comparing the relative performance of the tools against a number of standard tests.

The problem of selecting an appropriate evaluation method according to the existing specific circumstances is addressed in the DESMET. The evaluation method selection process will be affected by evaluation goals, the characteristics of the tool that will be evaluated, the characteristics of the organization in which the evaluation is performed, and the limitations of the evaluation exercise. These factors interact in complicated ways, which makes the identification of appropriate evaluation method difficult. The specific criteria that the DESMET method uses to determine the specific circumstances are:

1. The context of the evaluation.
2. The expected effect of using the method/tool.
3. The property of the object to be evaluated.
4. The size of impact of the method/tool.
5. The method/tool maturity.
6. The understandability of the method/tool.
7. The assessment capability of the organisation.

2.3.4. Organizational Effectiveness

Considerable work has been done in attempting to define and assess organizational effectiveness. The study in Krakower (1985) explains the criteria that can be employed to assess effectiveness in organizations and reasons for choosing them like a guide-book. Effectiveness is determined using different "point of view" approach. In this approach considerations about effectiveness are a function of the needs, preferences, and values of whoever's point of view is being considered.

Myers (2003) and Ozkan (2006) are other sources on information systems (IS) effectiveness assessment. In these dissertations background on effectiveness of organizational effectiveness are well established and assessment criteria are elaborated for information systems.

The effectiveness assessment approach used in Cameron (1980) is taken as the basis in the present study. Cameron suggests a useful strategy which is to restrict the concept of

organizational effectiveness by focusing on important a priori choices to guide the assessment of the organization. This assessment provides a basis for selecting among certain inevitable trade-offs. There can be no perfect evaluation, but the evaluations of the effectiveness can be improved when such critical questions are used. Cameron addresses 6 critical questions that must be considered in assessing effectiveness; these are subsequently expanded to the following 7 questions:

- *Whose perspective is being considered to judge effectiveness?* It is critical to make explicit from whose perspective effectiveness is defined and assessed, since each constituency will use different criteria.
- *What domain of activity is being the focus of the judgment?* Various domains exist in organizations and each one should be evaluated differently.
- *What level of analysis is being used?* Many levels can be used to evaluate the effectiveness: individual, subunit, organizational, industry, societal.
- *What is the purpose for judging effectiveness?* The purpose directly affects the judgment. Different data will be available, different sources will be appropriate, different amounts of cooperation or resistance will be encountered, different strategies will be necessary based on differences in purpose.
- *What time frame is being used?* Long-term effectiveness may be different from short-term effectiveness, and sometimes using wrong time frame may result not to detect effects and outcomes, since they may occur suddenly in the short term, or incrementally over the long term.
- *What types of data are being used?* Objective data or subjective, perceptual data? Objective data will tend to be more reliable, more easily quantifiable, and more representative. The type of the data also limits the scope and usefulness of the data. Subjective data can be used for a broader set of criteria, but can be biased, and lack validity and reliability.
- *What is the referent against which effectiveness is judged?* There are different possible methods for comparison: comparing competitors, comparing to a standard, comparing to the organizational goals, comparing to past performance, or evaluating on the basis of characteristics the organization possesses.

As the effectiveness assessment approach presented in Cameron (1980) is adopted in the present study, we start by capturing the evaluators' perspective on the effectiveness of a DSL.

CHAPTER 3

A FRAMEWORK FOR QUALITATIVE ASSESSMENT OF DOMAIN SPECIFIC LANGUAGES

For DSL assessment initially, a clear description of the DSL assessment objective is needed because the area is elusive and broad. It has been argued that assessment is subjective and depends on the context since DSLs are meaningful within their context. Assessment involves a large number of stakeholders, each with their own values and objectives.

In this chapter, a novel DSL assessment framework is presented. The literature review presented in Chapter 2 covered the broad field of PLs, models and DSL assessment and one of the main findings was the lack of a comprehensive, structured and systematic framework for DSL assessment. By analyzing the information gathered from the literature review, the building blocks of the framework were conceptualized to provide a general approach to DSL assessment. After presenting the objectives of the framework and the viewpoint approach used in the framework, this chapter describes FQAD and formulates a DSL assessment process.

3.1. Objectives Of The Framework

In order to gain some insight into what constitutes appropriate objectives of the assessment framework, a systematic approach is taken. Observing the parallelism of assessment processes of organizational effectiveness and DSL success, the following interrelated guidelines of DSL assessment process are deduced. Each of these is adapted from the guidelines developed for the context of organizational effectiveness by Cameron and Whetten (1983). The guidelines are taken as questions to ask when assessing DSLs. The questions are used both in the DSL assessment process to determine what and how to measure and in evaluating the success of the DSL assessment framework itself. These assessment guidelines are not mutually exclusive and shall be determined in practice according to the demands of a particular situation:

1. Explicit definition of the viewpoint
2. Determination of the domain
3. Determination of the level of analysis
4. Statement of the purpose
5. Determination of the time frame
6. Determination of the type of metrics

Theoretically, assessment objectives could be more systematically identified with the help of these six elements.

1. *Explicit definition of the viewpoint:*

   It is important to make explicit who is defining and assessing DSL, since each evaluator will use different criteria. Different stakeholders in a DSL environment may come to different conclusions about the success of the same DSL. Therefore, different stakeholders' viewpoints should be explicitly defined in a study to assess DSL.

2. *Determination of the domain*

   Many different domains exist for DSLs, and each one should be assessed differently. DSLs are meaningful only when they are considered within their domain. On the other hand, there are a large number of PL quality measures in the literature, making it difficult to determine what measures are appropriate in a particular domain. Therefore, focus must be on a particular domain while assessing a DSL.

3. *Determination of the level of analysis*

   DSL assessments can be made at many levels: individual, subunit, organizational, industry, societal. Therefore, the level of analysis should be explicitly defined in the DSL assessment.

4. *Statement of the purpose*

   The assessment almost always is affected by the purpose. Different data will be available, different strategies will be necessary based on differences in DSL assessment purpose. Therefore, the level of achievement of individual goals should be explicitly stated in the assessment.

5. *Determination of the time frame*

   Long-term success may be incompatible with short-term success, and sometimes effects and outcomes of the DSL may occur suddenly in the short term, or incrementally over the long term. Hence, the assessment should take the time frame into consideration.

6. *Determination of the type of measures*

   Objective or subjective data can be used in the assessment. Objective data tend to be more reliable. Subjective data allows broader set of criteria. Hence, the assessment should utilize both subjective data as well as objective data in assessing the success.

FQAD uses these guidelines to compose a comprehensive success assessment structure. FQAD considers the guidelines to clarify the meaning of the DSL success. Table 3 below shows the meaning of these guidelines as used in FQAD.

Table 3: Success Assessment Guidelines (Cameron and Whetten, 1983) adapted to FQAD

Guideline	Meaning in FQAD
Explicit definition of the viewpoint *Whose perspective is being considered to judge effectiveness??*	Stakeholders involved in the DSL development process namely: • Manager • Domain expert • Language developer • DSL implementer • DSL user
Determination of the domain *What domain of activity is being the focus of the judgment?*	Various tasks are performed in the DSL development process: • Perception of the organization/process • Deciding the investment in DSL • Deriving product roadmaps • Gathering the domain knowledge • Specification of the functional and non-functional requirements in an abstract way • Modeling variability • Specification of the language in a complete and consistent way • Formalization of the specification into metamodel • Construction of a library that implements the semantic notions in the DSL • Implementation of a compiler that translates DSL programs to a sequence of library calls(code generation framework) • Knowledge of using code generation tools • Writing DSL programs for all desired applications and compile them to use
Determination of the level of analysis *What level of analysis is being used?*	Varies depending on the evaluator understanding: • Individual • Subunit • Organizational • Societal
Statement of the purpose *What is the purpose for assessing effectiveness?*	Evaluation of success of DSL development processes in terms of the degree of meeting the evaluator goals

Table 3 (Continued)

Guideline	Meaning in FQAD
Determination of the time frame *What time frame is being used?*	• Long-term • Short-term
Determination of the type of measures *What type of data are being used for assessment of effectiveness*	• Objective data • Subjective data • Both objective and subjective data

3.2. Success Viewpoints

DSL success is a multi-perspective construct. To come up with a decision on whether a DSL is effective or not, or to what extent, several perspectives of DSL success should be reflected on the DSL assessment process.

Many different stakeholders may be involved in the assessment of a DSL. Each of these stakeholders focuses on certain characteristics of the DSL. To assess whether his concerns are addressed, each stakeholder forms a perspective of what properties the DSL should have. Figure 1 illustrates the use of two perspectives on a DSL, by showing two stakeholders that focus on different major concerns. Each of these stakeholders forms his perspective of the DSL success.

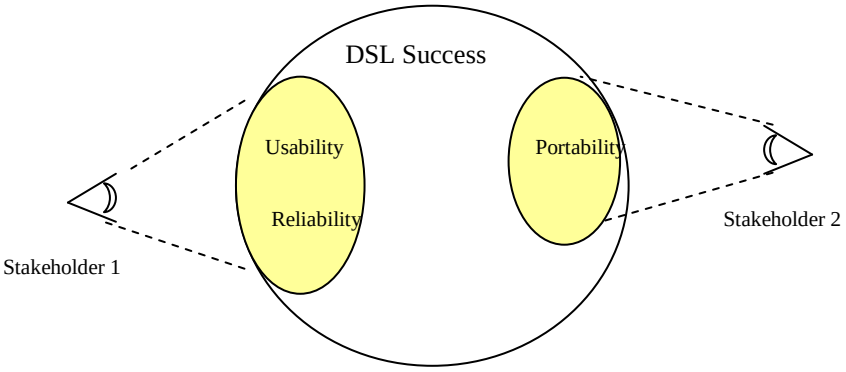

Figure 1. Stakeholders have different perspectives on DSL success

3.3. FQAD

3.3.1. DSL Quality Characteristics

Keeping in mind that the ISO/IEC 25010:2011 standard has a generic structure, we use this quality model as much as we can and we tailor it when needed, explaining the transition between the tailored and standard models. Below, we refine the basic characteristics, specify them for DSLs and link those characteristics with the computer language assessment literature.

1. Functional suitability: Functional suitability refers to the degree to which a DSL is fully developed. This means that all necessary functionality is present in the DSL. On the other hand, DSL should not include functionality that is not in the domain (Paige et al. 1999; Karsai et al. 2009). We use this characteristic to cover correctness, completeness (Mohagheghi et al. 2009), lack of domain understanding (Kelly and Pohjonen 2009), incompleteness (Wand and Weber 1993) and domain appropriateness (Krogstie 2003).
2. Usability: Usability of a DSL is the degree to which a DSL can be used by specified users to achieve specified goals. A DSL should be as simple as possible in order to express the domain concepts and to support its users (Khedker 1997; Paige et al. 1999; Kolovos et al. 2006; Karsai et al. 2009). Using symbols that are too simple or similar or unappealing should be avoided (Kelly and Pohjonen 2009).We consider understandability (Amstel et al. 2009), comprehensibility, appropriateness (Krogstie 2003), learnability (Karna et al. 2009), effort for adoption, effort required to build models (McKean and Sprinkle 2012), transparency (Haugen and Mohagheghi 2007), space economy (Paige et al. 1999), writability and readability (Khedker 1997) and simplicity, all under the title of usability.
3. Reliability: Reliability of a language is defined as the property of a language that aids producing reliable programs (Paige et al. 1999; Chen et al. 2005). DSL's support for error prevention and model checking is pointed out as a significant quality by Karna et al. (2009).
4. Maintainability: The degree to which a language promotes ease of program maintenance. DSLs can be altered and new concepts and concept extensions can be added (Kelly and Pohjonen 2009). Maintainability covers understandability and modifiability in this study (Chen et al. 2005; Haugen and Mohagheghi 2007). Modifiability can be described as the amount of effort required for modifying the DSL to provide different or additional functionality. We consider modularity (Amstel et al. 2009) also under this characteristic.
5. Productivity: Productivity of a DSL refers to the degree to which a language promotes programming productivity. Productivity is a characteristic related to the amount of resources expended by the user to achieve specified goals
6. Extendability: The degree to which a language has general mechanisms for users to add features (Chen et al. 2005; Kolovos et al. 2006; Haugen and Mohagheghi 2007). Scalability (Haugen and Mohagheghi 2007) is another sub-characteristic that is handled in extendability.
7. Compatibility: The degree to which a DSL is compatible to the domain and development process. We consider process compatibility under the title of

compatibility. It is the degree of a DSL to fit in a process since DSL is used as part of a development process with phases and roles.

8. Expressiveness: The degree to which a problem solving strategy can be mapped into a program naturally. In other words expressiveness is the relation between the program and what the programmer has in mind (Khedker 1997). Expressiveness is pointed out as one of the main characteristics of DSLs in (Haugen and Mohagheghi 2007). Uniqueness is the principle which can be defined as the sub-characteristic of the language that provides one and only one good way to express every concept of interest (Khedker 1997; Paige et al. 1999; Wand and Weber 1993). Duplicating the concepts and semantics of traditional programming languages, choosing the wrong representational paradigm and using libraries as the language should be avoided and the right abstraction level must be selected so as not to use too generic or too specific concepts. (Kelly and Pohjonen 2009) We use this characteristic to imply orthogonality as well, which means that each language construct is used to represent exactly one distinct concept in the domain (Kolovos et al. 2006; Wand and Weber 1993). Conformity (Kolovos et al. 2006) and consistency (Amstel et al. 2009) are other sub-characteristics that are handled in expressiveness.
9. Reusability: The degree to which a language construct can be used in more than one language. Reusability refers to what parts of a DSL are reused from or by other DSLs.
10. Integrability: DSL can be integrated with other languages used in development process (Haugen and Mohagheghi 2007).

It is stated in ISO/IEC 25022:2012 that the term usability, which is also used to refer to product quality characteristics, has a similar meaning to quality in use. Both usability characteristic under product quality model and usability characteristic under quality in use model can be used to specify and measure usability(ISO/IEC 25010:2011). Both effectiveness and satisfaction are referred as characteristics in the quality in use model (ISO/IEC 25010:2011).

We propose usability as one of the DSL characteristics. The importance of this characteristic depends on the viewpoint of the evaluator and expectations related to the DSL goal. We use the term success in a broader sense. Success levels are defined to determine the DSL's quality achievement. To be effective, a DSL needs to satisfy the characteristics which are also related to the sub-goals of the evaluator. For a DSL to be effective it may possess different characteristics which depend on the evaluator perspective. Those characteristics also form the sub-goals of the assessment. Hence, we measure satisfaction of stakeholders of a DSL according to their goal and expectations from a DSL.

We focused on eight software product quality characteristics of the standard. However, because the standard is not specifically designed to assess DSL, we included three additional characteristics: expressiveness, extendability, integrability, and removed two characteristics: security and portability as they are not considered to be primary concerns for DSL success. Security of a software product refers to the information and data protection degree that persons or other products have the right for data access

(ISO/IEC 25010:2011). Since everyone can access the language without any limitation, security characteristic is not suitable for DSLs. Portability of a software product refers to the level that enables product to be transferred from one operational or usage environment to another (ISO/IEC 25010:2011). Since a DSL cannot be transferred from one usage to another, this characteristic is also omitted. The productivity of the DSLs is handled in several studies (Kelly and Tolvanen 2008; Kelly and Pohjonen 2009; Karna et al. 2009). To link this characteristic with the relevant studies we replaced performance efficiency characteristic with productivity and redefined this characteristic from the DSL point of view. Performance efficiency characteristic in ISO/IEC 25010:2011 refers to performance of a product/system relative to the amount of resources. A major change is applied and productivity is defined from a quality in use perspective. DSLs emphasize domain expressiveness, so it is meaningful to define expressiveness as a characteristic (Khedker 1997; Haugen and Mohagheghi 2007). Extendability and integrability are not handled as a characteristic in ISO/IEC 25010:2011 but they are among the referred characteristics in the literature (Chen et al. 2005; Kolovos et al. 2006; Haugen and Mohagheghi 2007). For this reason extendability and integrability are proposed as separate characteristics in our study. We also shifted the reusability sub-characteristic to the characteristics level. We distinguished reusability sub-characteristic from the maintainability characteristic and redefined it as a characteristic because according to the feedback we obtained from the case studies it needs special attention in the assessment process. As a result FQAD consists of 10 characteristics and 26 sub characteristics, which are obtained from ISO/IEC 25010:2011 standard and literature on language evaluation. The characteristics, sub-characteristics and their descriptions are presented in Table 4 below.

Table 4. FQAD DSL Quality Characteristics and Sub-characteristics Descriptions

Quality Characteristics	**Sub-characteristics**	**Description**
Functional Suitability		Functional suitability of a DSL refers to the degree to which a DSL supports developing solutions to meet stated needs of the application domain.
	Completeness	All concepts and scenarios of the domain can be expressed in the DSL.
	Appropriateness	DSL is appropriate for the specific applications of the domain (e.g. to express an algorithm).
Usability		Usability of a DSL is the degree to which a DSL can be used by specified users to achieve specified goals.
	Comprehensibility	DSL language elements are understandable (e.g. language elements can be understood after reading their descriptions; such descriptions or tutorials of the DSL are available.)

Table 4 (Continued)

Quality Characteristics	**Sub-characteristics**	**Description**
	Learnability	The concepts and symbols of the language are learnable and rememberable (e.g.ease of learning DSL language elements, ease of learning to develop a program, effective DSL documentation.)
	Number of activities for task achievement	Language has capability to help users achieve their tasks in an acceptable number of program development activities.
	Likeability,user perception	Users can recognize whether the DSL is appropriate for their needs.
	Operability	DSL has language elements that facilitate to operate and control the language (e.g. language elements can be selected and put into practice easily, actions are undoable, error messages that explain recovery methods are available.)
	Attractiveness	DSL has symbols that are good-looking/attractive (attractive interaction, attractive appearance.)
	Compactness	The printed page representing a program developed using DSL, takes up not more than a single page.
Reliability		Reliability of a DSL is defined as the property of a language that aids producing reliable programs (model checking ability/preventing unexpected relations.)
	Model checking	DSL protects users against making errors. The DSL avoids the user to make mistakes.
	Correctness	DSL includes right elements and correct relations between them (DSL prevents unexpected interactions between its elements).
Maintainability		The degree to which a language promotes ease of program maintenance.
	Modifiability	DSL is designed such that it can provide different or additional functionality by modifying it, without degrading existing DSL functionality.
	Low coupling	DSL is composed of discrete components such that a change to one component has minimal impact on other components its elements.
Productivity		Productivity of a DSL refers to the degree to which a language promotes programming productivity. Productivity is a characteristic related to the amount of resources expended by the user to achieve specified goals.

Table 4 (Continued)

Quality Characteristics	**Sub-characteristics**	**Description**
	The development time	The development time of a program to meet the needs is improved.
	The amount of human resource	The amount of human resource used to develop the program is improved.
Extendability		The degree to which a language has general mechanisms for users to add new features.
	Mechanisms for users to add new features	DSL has general mechanisms for users to add new features.
Compatibility		The degree to which a DSL is compatible with the domain and development process.
	Compatibility to the domain	DSL is compatible with the domain. DSL has capability to operate with other elements of the domain with no modification required to perform a specific application in the domain.
	Compatibility to the development process	Using DSL to develop models fits in the development process, since it is used as part of a development process with phases and roles.
Expressiveness		The degree to which a problem solving strategy can be mapped into a program naturally.
	Mind to program mapping	A problem solving strategy can be mapped into a program easily.
	Uniqueness	The DSL that provides one and only one good way to express every concept of interest.
	Orthogonality	Each DSL construct is used to represent exactly one distinct concept in the domain.
	Correspondence to important domain concepts	The language constructs correspond to important domain concepts. DSL does not include domain concepts that are not important.
	Conflicting elements	DSL does not contain conflicting elements.
	Right abstraction level	DSL is at the right abstraction level such that it is not more complex or detailed than necessary.
Reusability		The degree to which a language constructs can be used in more than one language.
	Reusability	The symbols and other elements of the DSL can be used in more than one DSL, or in building other language elements. (e.g. using the definition of a language as a beginning to develop a new one.)

Table 4 (Continued)

Integrability		DSL can be integrated with other languages used in development process.
	Integrability	DSL can be integrated with other languages used in development process. (e.g. language integrability with other languages).

These aspects of assessment are inevitably open to subjective appraisal, as they must reflect the characteristics and abilities of the organization in which assessment is taking place. (For example, what is considered "easy" in one setting may deem cumbersome in another, etc.) This, we consider, does not reflect a shortcoming of FQAD but simply constitutes the organizational value of any assessment that will be carried out.

Example properties of the sub-characteristics are provided together with their descriptions to enhance the understandability of the characteristics and to avoid misinterpretation of the sub-characteristics by different evaluators.

DSL characteristics consist of self-contained sub-characteristics and the definitions do not overlap. In practice, DSL characteristics interact, as all goals of the DSL evaluators do, because a DSL must meet all goals of the evaluator. We handle this interaction between the characteristics by referring explicitly to DSL characteristics while determining the evaluator goals. Goal determination in FQAD enables evaluators to determine the focus of their DSL assessment efforts by choosing the DSL characteristics that best fit evaluators' objectives. Although there are some dependencies among DSL characteristics, the evaluators have freedom in their selection.

The proposed characteristics interact with each other in a manner that has an effect on the overall DSL success. It may be quite challenging to achieve multiple characteristics simultaneously. This shows that the evaluator's point of view is critical for assessment.

3.3.2. FQAD Assessment Model

Standard quality models define characteristics, sub-characteristics and the relationships between them but they explain the relationship between them without considering their value. However, not all characteristics equally influence success. To address this problem, for different DSLs, the relations and impacts of different quality characteristics are distinguished in FQAD.

The proposed assessment model evaluates the success of DSLs for compliance with the goals of the evaluator and qualitatively assesses the level of success using FQAD questionnaires with ordinal scales. In this study, the maturity level evaluation approach used in CMMI and feature analysis method developed in DESMET (Kitchenham et al. 1997) are taken as reference and via significant modifications, success level determination strategy is defined.

Assessment Model Components

The assessment model specifies that a DSL should have characteristics that address evaluator goals. To determine whether a DSL is effective, the evaluator maps his goals to the characteristics in this model.

Mapping of the evaluator goals to DSL characteristics enables the evaluator to track his goals as he assesses the DSL.

The assessment model components are summarized in Figure 2 below to illustrate their relationships.

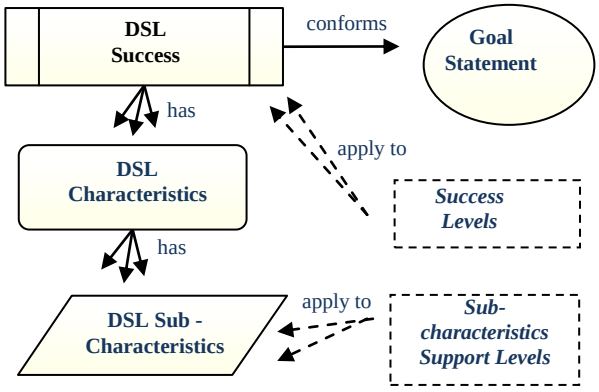

Figure 2. FQAD Assessment Model Components

The model components are described below:

- DSL Success: The success of a DSL is a cluster of related characteristics in a DSL, which, when owned collectively, satisfy a goal considered important for the DSL.

- Goal Statement: Statement of the goal describes the purpose of the assessment and is an informative component.

- DSL Characteristics: A DSL characteristic describes the unique characteristic that must be present in a high-quality DSL. A characteristic is a required assessment model component.

- DSL Sub-characteristics: A sub-characteristic is used to describe a quality measure that is considered important in achieving the associated characteristic. The sub-characteristics reflect the properties that are expected to result in achievement of the characteristics. A DSL sub-characteristic is an expected model component.

Assessment Levels

FQAD presents two assessment paths in terms of levels. One path enables the evaluator to assess directly the success level of a DSL. The other path enables evaluators to evaluate a characteristic of a DSL. These two assessment paths are associated with the two types of levels: sub-characteristics support levels and success levels.

Sub-characteristics support levels apply to a DSL's quality achievement in individual characteristics. These levels are a means for understanding the state of the quality corresponding to a given characteristic. Support levels adapted from (Kitchenham et al. 1997) and depicted in Table 5 are designated in an ordinal scale in which "full support" corresponds to the highest level. According to Kitchenham et al. (1997) granularity of the support levels depends upon the feature that is assessed and requirements. Based on this we defined four support levels for DSLs.

Table 5. DSL Quality Sub-characteristics Support Levels

Support level	Definition of Sub-characteristic Support level
No support	Fails to recognize the sub-characteristic. The sub-characteristic is not supported nor referred to in the DSL
Some support	The sub-characteristic is supported but not satisfactorily. It needs improvement.
Strong Support	The DSL meets the sub-characteristic.
Full support	All aspects of the sub-characteristic are covered and the DSL provides beyond the sub-characteristic requirements.

Success levels apply to a DSL's quality achievement. These levels are a means of assessing a DSL. The three success levels are presented in Table 6.

Table 6. DSL Success Levels

Success level	Definition of Success level
Incomplete	DSL is incomplete in satisfying its intended purpose and it needs improvements.
Satisfactory	DSL satisfies its intended purpose on average, yet it can be further improved.
Effective	DSL satisfies its intended purpose.

For a DSL to reach a particular success level, it must satisfy all of the sub-characteristics of the characteristics.

Success levels are used to characterize the success of a DSL as a whole, whereas the sub-characteristics support levels are used to characterize the state of a DSL relative to a DSL characteristic.

The weights for different sub-characteristics of the characteristics are not used in our method. Like CMMI rating elements (Specific and generic goals) we defined DSL characteristics as the rating elements. Like CMMI maturity level assessment approach DSL must satisfy all of the sub-characteristics of the characteristics to reach a particular success level.

Evaluator Profile and Success Level Determination Rules

The decision on the success level of a DSL is described in an "evaluator profile". An evaluator profile defines all of the characteristics to be addressed and targeted sub-characteristic support level for each. This profile guides which goals a DSL should address. The "evaluator profile" of a DSL is determined using the importance ranking of the evaluator made for each DSL characteristic.

The evaluation profile determines the characteristics that are most relevant to the evaluation goal. This allows handling the relation between the characteristics using the importance rankings of the characteristics which are determined according to the evaluation goals.

Importance degrees of DSL quality characteristics are used to determine the expectations of an evaluator from an effective DSL. The importance of characteristics are designated in an ordinal scale with the following scale levels: mandatory, desirable and nice to have, in which, "mandatory" represents a higher desire compared with the next importance degree "desirable". Kitchenham et al. (1997) recommends using at most three "desirability" gradations for practical purposes. For this reason we preferred to use two gradations for "desirability" importance degrees. A DSL sub-characteristics that does not possess a mandatory characteristic is, by definition, unacceptable. For this reason in FQAD assessment model, a mandatory sub-characteristic corresponds to having at least strong support support level.

Table 7. Importance degree vs. Support Level

Importance Degree	Sub-characteristic Support Level
Nice to have	No support
Desirable	Some support
Mandatory	Strong support
	Full support

The evaluator constructs the meaning of DSL success from his perspective. Table 7 presents the defining sub-characteristics support levels for each of the importance degrees on the left column. Any importance degree on the left column can be accepted as fulfilled if the sub-characteristic support level corresponds to the one at the same row or below it in this table. For example, if an evaluator chooses the importance degree of a characteristic as "desirable" this means that the sub-characteristics of that characteristic

must be assessed as sub-characteristic support level of "some support" as minimum in the questionnaire to satisfy the evaluator's expectations.

An effective DSL is one that possesses the characteristics that are most important to its evaluators. The importance of a characteristic can be assessed by considering whether it is mandatory or only desirable. If a DSL does not possess a mandatory characteristic described in an "evaluator profile", by definition, it is rated as incomplete.

The rules of the success level determination process are summarized as follows:

- Incomplete: For any characteristic importance degree if any sub-characteristic that is contained in that characteristic is rated below the correspondent sub-characteristic support level then the DSL success level is incomplete.

- Satisfactory: For any characteristic importance degree if all sub-characteristics that are contained in that characteristic are rated exactly same with the correspondent sub-characteristic support level then the DSL success level is satisfactory.

- Effective: If all sub-characteristics that are contained in a characteristic meet their required importance degree and any of those sub-characteristics is rated above the correspondent sub-characteristic support level then the DSL success level is effective.

An FQAD questionnaire that guides the evaluator in the assessment process and provides a template is developed using the assessment model (Appendix A). The first part of the questionnaire aims to get DSL quality characteristics importance ranking from the evaluator (Appendix-Form I). The rankings of the characteristics given by the evaluator represent the goal of the DSL success assessment.

The second part of the questionnaire presents the characteristics and sub-characteristics that a DSL may possess (Appendix-Form II). It provides a template for and guides the evaluator in the assessment process. Sub-characteristics are ranked according to a qualitative approach in which the evaluator may state his opinion and also may provide documents or show evidences like published papers on the focused DSL.

The third part of the questionnaire presents the assessment results (Appendix-Form III). The results are obtained according to the rules defined in this section. When the evaluator fills the forms I and II the results are automatically reflected on Form III.

Assessment Evidences

There are two types of evidence in FQAD: Tangible articles and supporting statements. For each sub-characteristic in the scope of a characteristic, the requirement for evidence requires either tangible articles or supporting statements, as a function of the sub-characteristic being assessed. The maturity of these evidences determines the support level of the sub-characteristic.

Tangible articles: These are direct or supporting outcomes of a characteristic. If, for example, the way DSL is implemented provides reusability, then the usage of some parts of the DSL in another DSL constitutes a tangible article of the sub-characteristic. Sometimes tangible articles are design documents, which possess a sub-characteristic. Project documents (internal reports, technical reports, etc.), and measurement records (e.g. cost, performance reports) can also be used. In addition, electronically documented publications (e.g. DSL development team intranet website, other available data via organization's intranet) provide extra sources of information.

Supporting statements: These are the opinions of the evaluators performing DSL assessment. These can be obtained through interviews and demonstrations.

3.3.3. A DSL Success Assessment Process

To assess the success of a DSL, a baseline sub-process for assessing DSL success was formulated where each stakeholder can contribute with their perspective. The sequencing of tasks in the process is illustrated in Figure 3.

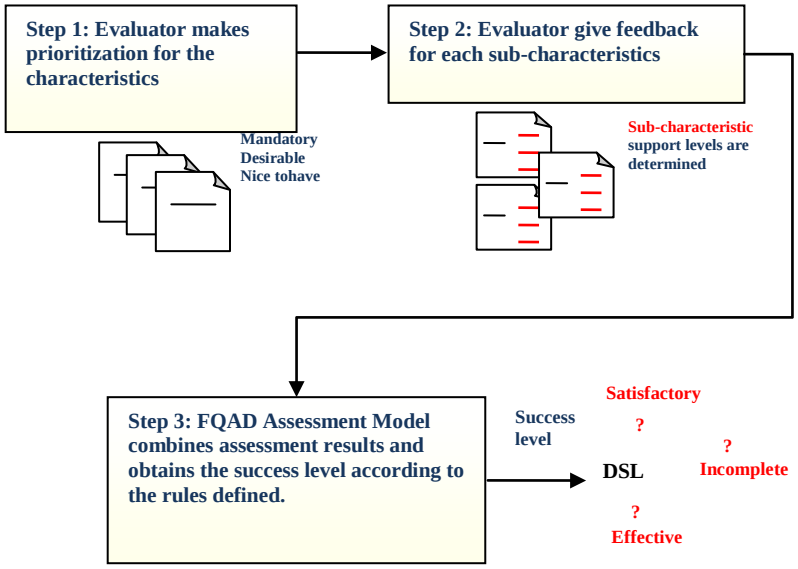

Figure 3. Assessment Steps in FQAD

Step 1. The DSL success evaluator starts the process by selecting importance rankings of the quality characteristics that are given in Part I of FQAD Questionnaire. The evaluator determines the importance of the characteristics aligned with his goal as defined in FQAD.

Step 2. The evaluator gives feedback on the determined quality characteristics as defined in FQAD. Part II of FQAD Questionnaire is used for this purpose. Sub-characteristic support levels are determined by the evaluator, according to the tangible articles or opinions of the evaluator.

Step 3. The assessment result is obtained according to the rules defined in FQAD assessment model.

CHAPTER 4

RESEARCH METHODOLOGY

The previous chapters reviewed the selected literature on PL assessment and DSL development particularly focusing on success measures of DSLs. In depth reviews of success measures related themes in these two domains of knowledge has located similarities and differences, and a foundation on which to propose the development of new knowledge in the area of DSL success measurement.

It was hypothesized that FQAD could be used to effectively measure from a theoretical perspective the success of DSLs using measures well aligned with the DSL development goals. The effectiveness of the theoretical framework requires testing and the assertions operationalised so that empirical data may be collected. This chapter develops a research methodology that addresses the problem area of DSL success measurement. The research methodology follows a qualitative approach to undertake multiple case studies in two organizations to evaluate the proposed framework.

In this chapter the research methodology will be presented. The chapter is structured into five major sections starting with a review of the problem area and questions arising from the literature review. This is followed with a discussion of the research approach where the basic assumptions underlying the research are discussed. The next section specifies the research design where the steps in the research process are outlined and the justifications for choosing a case study method is discussed. Section 4 gives the case study plan and finally the limitations and the expected problems are explored in the "Threats to Empirical Validity" section.

The presentation of the research methodology will be followed by the application of the methodology to evaluate FQAD. Two case studies shall be conducted to explore the applicability of FQAD. The case studies will be presented in the next chapter.

4.1. The Research Question

In any organisation, a DSL may be implemented in the software department under the responsibility of the assigned team leader. Hence the theoretical framework developed in Chapter 3 (FQAD) needs to be evaluated and tested in the software department, and with the stakeholders who are involved with DSL development or management. In Chapter 3 the theoretical answers to the question of what 'knowledge' may be found by applying the framework have been explored. The next phase is to test the hypotheses in

practice. The research question that has been derived from the review of literature of the DSL tasks is “How can an evaluator measure the success of a DSL using measures aligned with stakeholder perspectives?”.

Even though Chapter 3 (FQAD) theoretically answers the research question, empirical research is required to validate the claims of the framework.

4.2. Research Approach

There can be three approaches to research methods, namely, quantitative research, qualitative research and a mixed (both quantitative and qualitative) approach (Cresswell 2003; Runeson and Höst 2009).

Quantitative research methods were developed to study natural phenomena which include survey methods, laboratory experiments, formal methods (e.g. econometrics) and numerical methods (including laboratory modeling). Qualitative methods were developed to enable researchers to study social and cultural phenomena using methods like action research, case study, and ethnography (Myers 1997). Qualitative data sources may include observation, interviews and questionnaires, documents and texts, and the researcher’s impressions and reactions (Tokdemir 2006). In quantitative research the data are in the form of numbers while in qualitative research the data are non-numeric (Punch 2008).

In electrical and computer engineering, it is widespread to use quantitative methods. However, quantitative methods do not take into consideration environmental effects, and the social interactions in the environment. As a result, some of the studies combine both qualitative and quantitative methods (Kaplan and Duchon 1988).

In this study, the proposed framework considers the relationships of success of DSLs with stakeholders and environmental factors, which includes both qualitative and quantitative aspects. For this reason, both qualitative and quantitative methods are employed in FQAD process. Accordingly, in order to explore whether the proposed framework elements are valid or not, interviews were held with DSL stakeholders. Then, for success assessment, a questionnaire was applied to the stakeholders of DSLs to rate the success of their DSLs. The interviews were recorded for analysis of data.

After the proposed framework structure was verified through the case studies, the results were evaluated and the framework was updated with the missing elements. In order to mature FQAD process one case has been used as the preliminary test case. The preliminary case was used as a verification of FQAD elements. After updating and verifying FQAD elements in the preliminary case study, it was applied to a case which is actively used in an organization and has a more sophisticated DSL than the first case study. The interviews and questionnaires were applied to two different software departments of an organization where these departments develop software for different domains.

4.3. Research Design

Blaikie (2000) points out that a research design should help in providing answers for some basic questions namely ‘what’, ‘why’ and ‘how’ of the study. This serves as a very useful starting point to explain and define the proposed research. While the ‘what’ and ‘why’ components have been answered in the literature review section, the ‘research methodology’ chapter addresses the latter question by explaining the method of acquiring knowledge and the underlying philosophy undertaken for the design. Blaikie (2000) has further subdivided this question into four components namely the type of research strategy that will be used, the source of data, how the data will be collected and analysed and finally when will each of these stage will be carried out.

A research method is defined in (Myers and Avison 2002) as a technique for collecting data. Both qualitative and quantitative research is conducted in this research because the data that is to be collected is qualitative in nature. Data are collected from various sources and this requires in depth interviews. Myers and Avison(2002), categorised qualitative research into four main methods namely action research, case study research, ethnographic research, and grounded theory. In attempting to study the phenomena of measuring DSL success in an organisation from a stakeholder’s perspective, it was found appropriate to use the case study method due to the need for a deep inquiry into the problem and the novelty involved.

4.3.1. Case Study Method

The usage of empirical research methods and their contributions to improving knowledge is continuously growing in software engineering (Runeson and Host 2009). Case Study is an empirical research method (Robson 2002; Yin 2003; Benbasat et al.1987) which investigates contemporary phenomena in their context. Yin (2003) describes it as an empirical inquiry and remarks that the boundaries between phenomenon and context may not be clearly evident.

Runeson&Host (2009)discuss the appropriateness of the case study methodology for software engineering research as: “The case study methodology is well suited for many kinds of software engineering research, as the objects of study are contemporary phenomena, which are hard to study in isolation” and for the generalizability it is stated that “Case studies do not generate the same results on e.g. causal relationships as controlled experiments do, but they provide deeper understanding of the phenomena under study. As they are different from analytical and controlled empirical studies, case studies have been criticized for being of less value, impossible to generalize from, being biased by researchers etc. This critique can be met by applying proper research methodology practices as well as reconsidering that knowledge is more than statistical significance (Flyvbjerg 2007; Lee 1989)“. In the present study proper research methodology practices are applied, as described in this chapter, to support the analytical generalizability of the study.

Case studies have often been used as the preliminary exploratory stage of a research project (Rowley 2002). However they are also used for descriptive purposes, if the

generality of the phenomenon is of secondary importance. Case studies may also be used in explanatory purposes although the isolation of factors may be a problem (Runeson and Host 2009).

Yin (2003) identifies the following three factors that determine the best research methodology:

- Research question type,
- Control capability over behavioral events,
- The degree of focus on contemporary as opposed to historical events.

Case studies are useful in answering the questions "How?" and "Why?", and can be used for exploratory, descriptive and explanatory research (Rowley 2002). The research questions of this dissertation are of the types of "why?" and "how?". For such questions, which are explanatory, the use of case studies is appropriate.

The unique strength of the case study approach is its ability to deal with a full variety of evidence (Yin 2003). This is the approach needed to exploit the rich source of information available about FQAD and to come to an understanding as to why and how FQAD will effectively be used as a tool for assessing DSLs.

Yin (2003) states that if the phenomenon and the context cannot be separated and it is not possible to stabilize certain number of variables a case study research should be preferred to other research methods. As there is no directly comparable FQAD method to ours, instead of experiments, we have decided that case study research would be appropriate to investigate the applicability of FQAD approach. On the other hand in order to improve the validity of the empirical conclusions, we have performed multiple case studies.

Table 8. Analysis for choosing the case study research method (adapted from (Benbasat et al. 2002))

Questions	Analysis	Relation to this Study	The Choice
Can the phenomenon of interest be studied outside its natural setting?	If No, then Case Study	This is a study where a framework is tested in an organisational context and thus cannot be studied outside its natural setting.	Case Study Method
Must the study focus on contemporary events?	If yes, then Case Study	The topic is contemporary and not historic as it involves studying contemporary phenomena.	Case Study Method
Is control or manipulation of subjects or events necessary?	If No, then Case Study	There is absolutely no need to control or manipulate the subjects or events, as the researcher is not present during the implementation.	Case Study Method
Does the phenomenon of interest enjoy an established theoretical database?	If No, then Case Study	There is little academic literature in this topic and the framework, being innovative has not been the subject of prior research.	Case Study Method

To further identify the appropriateness of the case study approach for this research, we did a self analysis by asking the four questions recommended by (Benbasat et al. 2002) given in Table 8, that directed us to the case study method.

Special attention is taken during the conclusion phase of the research , especially on the capacity in the conclusion for the generalization of the individual cases. This was an analytical generalization (Yin 2003) to the theory of DSL assessment. As Vaus (2001) states, "theoretical generalization involves generalizing from a study to a theory. Rather than asking what a study tells us about the wider population (statistical generalization), we ask what the study tells us about a specific theory. Case study designs are fundamentally theoretical. They are designed to help develop and refine theories". Runeson&Host (2009) emphasize that "for case studies, the intention is to enable analytical generalization where the results are extended to cases which have common characteristics and hence for which the findings are relevant, i.e. defining a theory". Yin (2003) also supports theoretical generalization, stating that "in analytical generalization, the investigator is striving to generalize a particular set of results to some broader theory". In chapter 6 of this thesis it is shown that case studies' results allowed analytical generalizations to be made that contribute to the advancement of the theory of DSL assessment.

4.3.2. Multiple Case Study

Benbasat et. al. (1987) argues that multiple case studies can be used when the aim of the research is description, theory building or theory testing and states that most research studies require multiple case studies. In addition, multiple case studies help cross analysis and extension of theory.

Multiple case study approach used in the current research is illustrated in Figure 4. This figure has been adapted from (Yin 2003). The figure indicates the steps performed as follows:

- **Develop Theory**: Determine the questions, theoretical propositions (pointing attention, limiting scope, possible links between phenomena) and units of analysis (units must be at the same level as study questions) of the study.
- **Design Data Collection Process**: Link the data to propositions. State the criteria for interpreting the findings by iterating between propositions and data.
- **Select Cases**: Select the cases in the same way as the topic of an experiment is selected. Selected cases reflect characteristics and problems identified in the underlying conceptual framework.
- **Conduct Case Study**: Each individual case study consists of a "whole" study, in which the results are dealt within its own context. Both the individual case and multiple case results are important in the overall study.
- For each individual case, the findings are reported. Following that, cross analysis and discussion of the findings are elaborated. Cross analysis of the cases helps in understanding why cases have different results and their reasons.

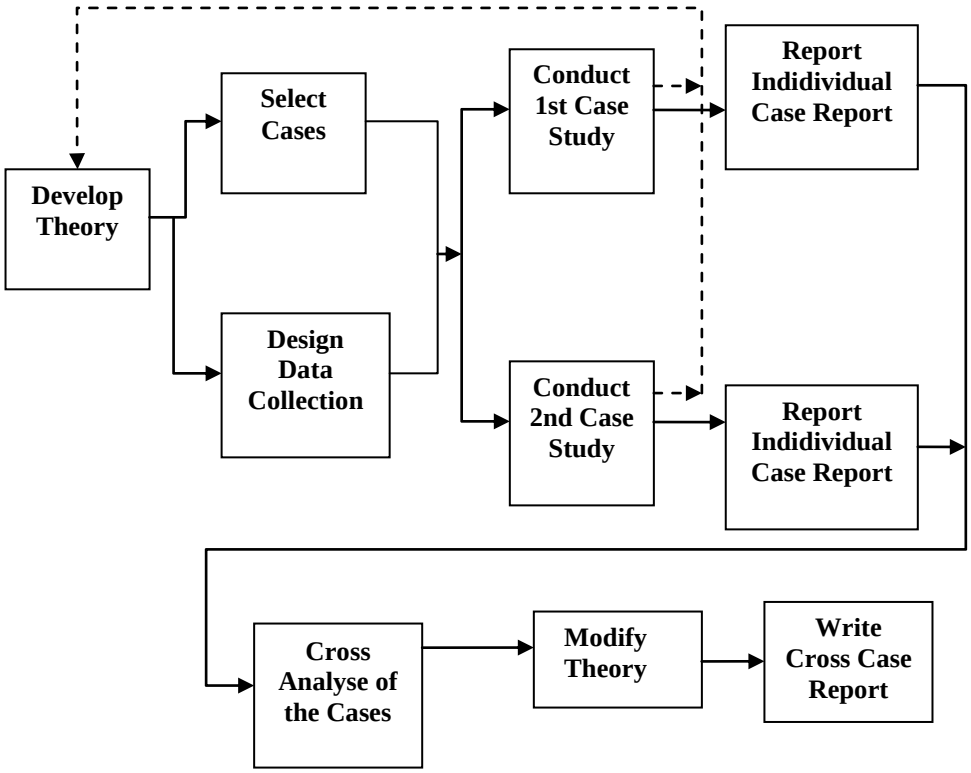

Figure 4. Case Study Method

We conducted two case studies to validate the proposed FQAD approach. The first case study was performed with a DSL developed in a software development department of an organization. A DSL developed by another software development department of an organization was used in the second case study. The two cases have aimed to address different research questions.

The first case study has been mainly conducted for exploration purposes. We aimed to improve and refine the framework we had initially formulated by applying it to a real environment. The second case study has been conducted to verify and validate the framework before the application of inspection. Accordingly, the aim of the second case study has been to validate the effectiveness of FQAD.

4.3.3. Data Collection and Analysis

There are multiple methods used for data collection in the case study method (Benbasat et al. 2002). In the current research study, interviews are mainly used to get the personal experience of the stakeholders. Prior to organisation visits, the data to be gathered is outlined in detail. The templates of the questionnaire like lists are given in Appendices. The duration of each interview and data gathering process lasted approximately two hours in each case study. If necessary a further interview could be arranged to resolve any unclear matter. Feedback was posted via e-mail to the person upon request. The selection of the interviewees was based on their direct involvement with the DSL Development phases.

In order to increase the consistency of the information gathered, in addition to the interviews, project documents (written material ranging from internal reports, technical reports, training materials and related publications), and measurement records (e.g. cost, workmanship reports, performance reports) were used extensively. In addition, electronically documented publications (e.g. DSL development team intranet website, other available data via organization's intranet) provided an extra source of information. Moreover direct observation helped to understand and capture details.

Having multiple data sources and cross checking of the data gathered has provided greater support in order to achieve more robust conclusions.

4.3.4. Questionnaires

Questionnaires are used to gather data from a population or a sample of a population in a structured way (Robson 2002). A questionnaire was conducted to fulfill the data collection need of this research. The questionnaire survey process used in this study is depicted in the Figure 5 below.

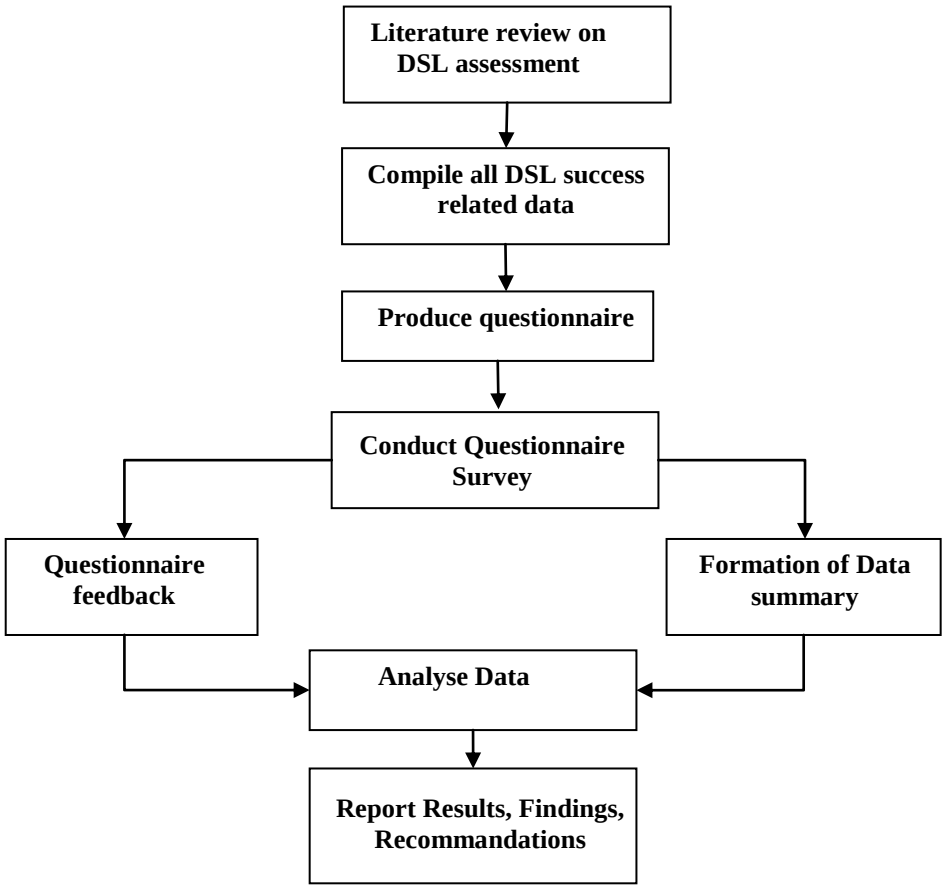

Figure 5. Questionnaire Process

Questionnaire is the main data collection source of this study. Questionnaire facilitates the collection of related information.

The objective of the questionnaire is to assess the success of a DSL. The stakeholders of the DSL and the ones who want to assess it use the questionnaire to evaluate the DSL's success.

According to (Robson 2002), two primary types of questionnaire are open-ended or closed-ended. In order for respondents to answer the questionnaire easier and simpler, closed form questions are applied in this study. Closed-ended questions limit the scope of responses but it yields quantifiable data more easily than open-ended questions. Keeping in mind that, excessively long questionnaire will affect the respondents' answers (Yin 2003) and will also discourage the participants, clear and straightforward questionnaire is targeted to gather the broad information on the subject. In the beginning of each questionnaire section, an introductory statement which briefly defines the survey's purpose and addresses confidentiality is included. Ranking questions are chosen as the scaling strategy. Response options are ranked according to their importance. Ordinal scales are used to gather respondents' selection.

The questionnaire is divided into two main sections. The first section is used to gather the information on the characteristics of the DSL and this section provides the links between the DSL characteristics and evaluator goal. The second and last section provides sub-characteristics related statements to gather the evaluator's evaluation on the DSL. This last part of the questionnaire presents the sub-characteristics of each characteristic which are determined from the relevant literature and ISO/IEC 25010 quality model.

Questions are prepared paying attention to the following rules (Robson 2002):

- be clear and unambiguous and not use language that is inappropriate for the participants
- be simple rather than complex
- ensure that rating scales are mutually exclusive (if a single response is required)
- avoid questions that may irritate participants or could be perceived to be threatening.

Instructions on how to conduct the questionnaire are provided explicitly, clearly and politely.

Questionnaires are administered via the email. The questionnaire covering information and face to face interviews are supplied to outline the aims of the research and assure participants of confidentiality. Questionnaires are returned electronically online in which data entry is automated.

Steps in the data collection process identified for the proposed study is as follows:

- Give the participant information and the consent form for approval from their side.
- Brief the participants regarding the assessment model and state the purpose.
- Conduct a questionnaire and collect them.
- Conduct a detailed interview to collect any related document like DSL development team intranet website, other available data via organization's intranet.
- Get back to the participants with the transcripted report to validate the response.

4.4. Case Study Plan

The case studies were chosen from two different domains. Command and control systems and electronic warfare systems departments of a defense systems company ASELSAN were chosen because reference data for software performance evaluation is available in those domains. We have chosen two different domains (organizational sections) in one company for reliability. In the defense domain, ASELSAN is the leading company in Turkey, with four different sections for different sub-domains and having approximately 4000 personnel in 2013. It has different software departments in each section with approximately 250 software engineers. Both of the software departments studied in the organization develops their software in-house and the sections are currently acquired CMMI level 3 certification.

In the interviews, all the stakeholders knowledgeable in the DSL used in the department were selected to gather the data related to FQAD. Similarly, management was interviewed to determine the decision making process to develop a DSL, since management knows and decides on the organizational strategies. We planned to perform the interviews with a single interviewee in each case to gather the necessary data in depth.

In order to collect the questionnaire answers, the questionnaires were sent to the participants who were available for the study.

4.4.1. About ASELSAN

ASELSAN is a high technology defense industry company, with its most distinctive expertise in the fields of design, development, production and system integration of hardware/software systems. The company also has major business in custom manufacturing involving electronic assembly, communications hardware, radar data integration, real time command, control, and communications (C3) and avionic systems, data fusion, and command center design and installation.

4.4.2. Case A

The first case study was performed for a DSL developed in electronic warfare mission software department (ASELSAN REHİS Group) in a defense systems organization to test the framework elements and assessment process. The software department is specialized in electronic warfare systems software and has been in this sector for 20

years. It has 62 employees responsible for development and maintenance of the software for different projects.

4.4.3. Case B

The second case study was performed for a DSL developed in the command and control systems domain software engineering department (ASELSAN SST Group) in the same company but in a different section from that considered in case study A to test the framework elements and assessment process. The software department is specialized in command and control systems software and has been in this sector for, again, 20 years. It has 74 employees responsible for development and maintenance of the software for different projects.

4.5. Threats to Empirical Validity

When undertaking qualitative research, validity and reliability of the research are important factors that must be taken into consideration. Quality checks to ensure that the case study is done in a proper manner need to be performed to prevent subjective interpretations.

Four important aspects of the quality of an empirical work are recognized as “construct validity”, “internal validity”, “external validity”, “reliability” (Runeson and Host 2009). In our context the following points apply regarding validity:

- Construct Validity focuses on the correctness of the interpretation and measurement of the theoretical constructs.

- Internal Validity focuses on the design of the study and controls whether the results are consistent with the data.

- External Validity focuses on whether claims for the generalizability of the results are justified. For case studies, there is no population from which statistically representative samples could be analyzed. However, analytical generalization can be drawn to define a theory by extending results for cases which have common characteristics.

- Reliability focuses on replicability of the study results by other researchers.

CHAPTER 5

CASE STUDIES AND RESULTS

5.1. Case Study Planning and Operation

In the previous sections we proposed a framework to help DSL stakeholders to assess the success of a DSL, by providing an evaluation roadmap to follow. In this section, we report the case studies conducted to explore and mature various aspects of our framework. In these case studies, DSL stakeholders assessed their languages according to our framework and the findings have been analyzed with the aim of maturing and enhancing our framework.

The framework proposed in this thesis is independent from the domain of the DSL assessed. The framework guides the evaluator to assess the DSLs in its context. For this reason the framework described in this study will also scale to other DSL solutions.

We were aware of the high level of confidentiality stipulations of the military industry, and our request to hold interviews with the DSL stakeholders with high-level security measures were accepted. The reader may find the details of cases and the findings in the following sections.

The case studies are planned and operated in accordance with the principles described by Runeson and Host (2009), Yin (2003), Benbasat et al. (1987). Below, first, the goal of the study is stated, and then the selection of the case and the subjects are described. Finally, how the data is collected and analyzed is explained before the discussion on the validity of the process.

5.1.1. Goals of the Case Studies

Case Study 1 (Exploratory Study).

The significance of this preliminary case is to be exploratory so that we could finalize the list of DSL assessment characteristics for our FQAD. We aimed to improve and refine the process we had presented in the previous chapters by applying it to a real environment.

Case Study 2 (Validatory Study).

Following the exploratory case study, we planned one more case study to validate and test the finalized framework in order to come up with a solid set of DSL quality

characteristics and assessment model for the assessment of a typical DSL. In the second case, we aimed to assess the success of a DSL used in a large software project where the focus of the DSL was to implement a highly error prone part of the system software.

5.1.2. Case and Subjects Selection

In order to answer the case study research questions, we needed to measure the success of our FQAD method and to finalize the individual constituents of the method, specifically in terms of the DSL quality characteristics and their operational definitions, as well as the measures to be applied for those characteristics. For this purpose, we began by selecting two groups of DSL stakeholders to assess their DSLs based on our framework. In the end of their DSL evaluation we asked them to answer a questionnaire to evaluate our method. Before the application of our framework they had not applied any systematic approach for DSL success assessment, but did resort to ongoing assessment based only on expert opinions.

For the purposes of this thesis, we selected cases that are actually in professional use. Although the cases are chosen from the same company, the domains where DSLs are developed and used are completely different. So this means that every DSL is different. Command and control domain software and electronic warfare software differs in the both design and application phases. This is the reason why there are different departments in the same company with different names and physical locations. These two cases cover two different and important problem domains and application areas.

The context is considered being the specific application domain, so, according to Runeson and Host's terminology, the cases are holistic case studies with one unit of analysis. The companies were selected based on existing academia-industry relations and the units of analysis were determined firstly based on availability, but more significantly, according to the case study purposes.

When developing DSLs, potential stakeholders may be all those persons involved in the development process. The organizations, in which the presented case study has been conducted, and the corresponding roles are outlined in Table 9. The DSL development team is responsible for performing all of the tasks needed to build a DSL.

We formed two groups of participants in the two different case studies, as study subjects who are experts from ASELSAN and responsible for developing and using the DSL under consideration. All involved experts had extensive experience in software engineering in general, and concerning DSLs, in particular. The selection of interviewees was based on their direct involvement with DSL development phases. They represented different viewpoints regarding the assessment of the DSL, e.g., managers, developers. During the study, the time of the DSL stakeholders was limited, so we tried to make optimal use of this limited resource. Therefore, we distributed all relevant steps for the assessment process that require the involvement of the domain experts to one presentation and two questionnaires.

Table 9. Stakeholders and Tasks for DSL Development Process in the Organization

Role	Tasks
Manager	- perceive the organization/process - decide the investment in DSL - derive product roadmaps
Domain Expert	- gather the domain knowledge - specify the functional and non-functional requirements in an abstract way - model variability
Language Developer	- specify the language in a complete and consistent way - formalize the specification into metamodel
DSL Implementer	- construct a library that implements the semantic notions in the DSL - implement a compiler that translates DSL programs to a sequence of library calls(code generation framework) - know using code generation tools
DSL User	- write DSL programs for all desired applications and compile them to use

Each participant acted as an evaluator and assessed the DSL from his/her perspective for different roles. During the evaluation process, we never gave any kind of advice, but we did answer any questions about our framework. We consider this approach to have helped to avoid the contamination of the results according to our expectations.

Case Study 1

The investigated DSL was one that was developed in the software development department of ASELSAN REHIS Group, where radar and electronic warfare systems software is being developed and tested. Having a CMMI-3 certification, ASELSAN-REHIS group is mainly specialized in defense industry projects developing products with high-end software development techniques like agile programming, software product lines, model driven software development, and reusable components. The software development project team uses model driven development practices to develop the DSL. The DSL for which the assessment process was applied is used to generate one of the software modules validated by the department. The DSL was released in 2011 and has been continuously maintained and extended with new features since then. Each year multiple releases take place with updates of the DSL. Thus, at the time the assessment was done, 3 releases had been announced.

The developed DSL aims to support the rapid development and evolution of data intensive modules (called MDF (Mission Data File)) of the embedded software used. The approach includes the automated generation of MDF from a conceptual model, and the automated generation of a data inquiry API providing functions with a conceptual view of the MDF. An example view of the small part of a model developed using the DSL is given in Figure 6 below. In this figure MDF structure is expressed using Table1, Table2 and Table3 and elements in these tables. An inquiry is defined using Query1 and rules of the inquiry are defined within Query1.

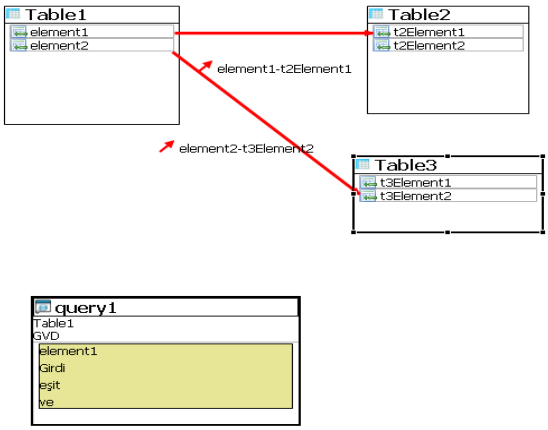

Figure 6. An example view of the model developed using the DSL in Case Study 1

In this case study, there were 5 stakeholders in the team involved in the DSL development process, with different roles in different stages. One stakeholder only has manager role, 2 stakeholders share the Domain Expert, Language Developer, DSL Implementer roles, and 2 stakeholders share the Domain Expert, Language Developer, DSL User roles.

Case Study 2

The case focuses on a DSL developed in a software development department of ASELSAN SST Group, where defense systems software is being developed and tested. Having a CMMI-3 certification, ASELSAN-SST group is mainly specialized in military projects, developing products with high-end software techniques like software product lines, component based software development and model driven software development. The software development project team uses component based software technologies heavily in their projects. The DSL for which the assessment process was applied is used to generate one of the software modules validated by the department. The DSL was released in 2010 and has been continuously maintained and extended with new features since then. Each year multiple releases are issued with updates of the DSL. .Thus, at the time of assessment, 7 releases had been issued.

The developed DSL aims to support a specific part of the fire control systems domain which models the sensors and drivers of the system. The functionality includes the automated generation of executable codes from a conceptual model. Wind sensor, navigation device, global positioning system antenna and connectors can be given as some examples of the concepts used in the language. The DSL generates the code that performs the transformation between the platforms in the fire control systems. An example view of a small part of a model developed using the DSL is given in Figure 7 below. In this figure platforms are expressed using Mount_Table and Vehicle, and relations between these platforms are shown with the connectors between them.

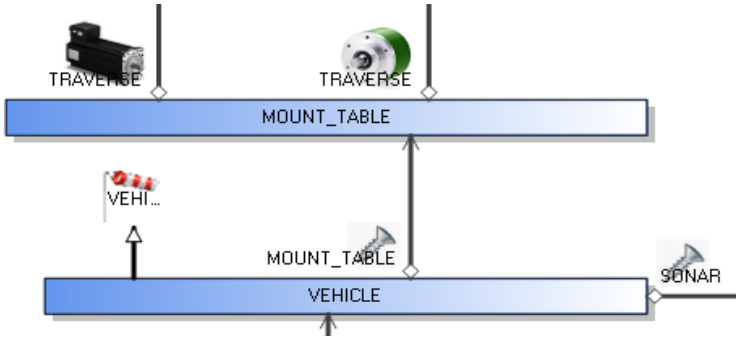

Figure 7. An example view of the model developed using the DSL in Case Study 2

In the second case study, there were 3 stakeholders in the team involved in the DSL development process. One stakeholder only has manager role, 1 stakeholder performs the Domain Expert, Language Developer, DSL Implementer, DSL User roles and 1 stakeholder performs the Language Developer, DSL User roles.

5.1.3. Data Collection and Analysis

Multiple data collection techniques were applied within this case study. We applied interviews and questionnaires to investigate personal experiences of stakeholders. Interviews were fully structured so taking notes was sufficient to record the extra opinions of the participants.

FQAD was explained to the participants in a meeting. All the meetings took place on the premises of the studied organisation - one was in the office of the software department manager; the others were held in the meeting room of the Software Department of the company. In these meetings, we presented the details on the aims and the steps of each stage of the evaluation process, and explained the tasks of the evaluators. The interviews with the participants were performed one by one (directed by the researcher). The duration of each interview was two hours. First, the purpose of the study was stated to the stakeholders. FQAD structure and components were explained in detail. Then the quality characteristics that were represented on a set of sub-characteristics and represented by statements as shown in Appendix A were submitted to the stakeholders. The material of our framework, quality characteristics importance ranking questionnaire (Appendix A- Form I), success assessment questionnaire (Appendix A- Form II) and template to store domain experts' opinion on FQAD (Section 5.1) was also made available to them.

The document that describes the quality characteristics/sub-characteristics was sent out via e-mail to the stakeholders. Each stakeholder indicated his/her specific importance rankings. During the importance ranking, the stakeholders were given the possibility to add new quality characteristics and sub-characteristics. The evaluator gave feedback on the determined quality characteristics defined in FQAD. The second part of the FQAD questionnaire was used for this purpose. Sub-characteristic support levels were determined by the evaluator, according to the tangible articles of the DSL or opinions of the evaluator. Lastly, results were gathered, analyzed and assessed.

After the DSL assessment, an FQAD evaluation template was sent out to each stakeholder, with the objective of receiving feedback on the FQAD process. The results from the analysis of the evaluation template answers are summarized in Section 5.

Case Study 1

All quality characteristics were described in a word document and sent out via e-mail to the 5 stakeholders.

Case Study 2

The results of the first case study were carefully analyzed and the critiques received were reflected to FQAD before the second case study was performed. The details of the improvements made based on the results of the first case study are presented in Section 5. One of the criticisms made for the documentation was that using word documents for the assessment was time consuming. For this reason, excel spreadsheets are used for the assessment and a new form is added in which the results of the assessment are automatically expressed with the help of excel formulas.

All quality characteristics were described in a spreadsheet and sent out via e-mail to the 3 stakeholders.

Gathering Multiple Sources of Evidences

The interviews in the case studies are performed without any voice recorder because of confidentiality regulations of the company. All participants were informed about the research details before the interviews with the aim of maintaining initial trust, avoiding unethical issues, and preparing them for the interviews.

The participants were happy to provide the researcher with all relevant information through interviews, with their notes and reports. In Case Study 1, a paper published in a national conference was used, and also official reports on DSL development and usage and intranet website were available to the researcher. The same was true with Case Study 2. Hence multiple sources of evidence was used namely the interviews, written reports on DSL and conference papers. In case study 1 and case study 2 when the researcher studied the DSL documents it was evident that the same was given during the interview. For example “productivity” increase that was reported in case study 1 documents (report and conference paper) was also evaluated as “full support“ or “strong support” by the participants during the evaluation of the DSL. This shows consistency between the evaluations and the reported documents of the DSL.

The entire process of performing the two case studies took two months. In case study 1 four weeks were spent for performing the interviews and one week for analyzing the results and case study related documents. In case study 2 two weeks were spent for performing the interviews and one week for analyzing the results and case study related documents. Although the interviews took 2 hours totally, the time was mostly spent for planning the meetings and getting the forms from the participants after the interviews. This was because of the time limits of the participants due to their project delivery conditions.

5.1.4. Evaluation of Validity

Four important aspects of the quality of an empirical work are recognized as “construct validity”, “internal validity”, “external validity”, “reliability” (Runeson and Host 2009). In our context the following points apply regarding validity:

- Construct Validity focuses on the correctness of the interpretation and measurement of the theoretical constructs. In this study, multiple sources of evidence were used with case studies. The evidence is collected in the form of questionnaires, written notes and documents. The evidence is followed from the goal to the case study report and traced back to the research goal. The research supervisor which is an external observer had the role for this purpose. There remains the threat that the used questionnaire might not adequately represent the research goal. Although the questionnaire was developed by an intensive literature research, it cannot be ensured that there are no topics missing. In addition, the answers of the participants are inherently subjective. To overcome this threat, using the assessment evidences collected during the interviews we improved their objectivity.

- Internal Validity focuses on the design of the study and controls whether the results are consistent with the data. Our empirical study cannot be considered as a controlled experiment, since all subjects took part in the development of two case studies. However any bias that may have remained can be eliminated by applying FQAD in further case studies in the future. In addition, when conducting the survey, we avoided any sharing of information between subjects. By doing this we prevented answers of a respondent to be influenced by other replies (Pfleeger and Kitchenham 2001). In order to deal with this issue we made sure that no respondent had access to the responses of the others.

- External Validity focuses on whether claims for the generalizability of the results are justified. The limited size and complexity of the case studies do restrict the generalizability of our results. The team in the case studies was selected largely by the researcher, according to the eagerness expressed by candidates to participate in the DSL assessment process. Since no formal selection of the case study team took place, it cannot be stated whether the team was representative of other DSL developers. However, by applying FQAD in more case studies and projects, generalizability of the results may be enhanced.

- Reliability focuses on replicability of the study results by other researchers. Planning and operation of the case study was done and documented systematically so that replicability has been ensured.

5.2. Results From The Assessment of FQAD

After the description of the cases and subjects, this section briefly describes the evaluation of FQAD. The evaluation process that is used in the case studies and its results regarding FQAD components are outlined.

5.2.1. Evaluation Strategy

An evaluation strategy and a standard questionnaire (called Evaluation Template) were developed for evaluating FQAD. Evaluation criteria determined in (Kitchenham et al. 1997) are adopted for the evaluation of FQAD.

The Evaluation Template is used to provide a context for planning an evaluation in which the methods and procedures described in FQAD were related to the evaluation criteria.

Evaluation criteria details:

- Basic Validation – This criterion deals with the opinion of the potential users as to whether they could use FQAD for real or not. It is concerned with the quality of the FQAD structure. Sub-features identified for the evaluation were: completeness, understandability, internal consistency, organisation, appropriateness for audience, readability

- Use Validation – This criterion investigates whether the method is helpful or not. Sub-features identified for the evaluation were: understandability, ease of implementation, completeness, ability to produce expected results, ability to produce relevant results, ability to produce usable results

- Gain Validation - This criterion investigates whether the method is better than what was available previously or not. It is concerned with the benefits delivered by the component. Sub-features identified for evaluation exercise were: appropriateness for task, comparison with alternative approaches, support for decision-making, cost effectiveness

5.2.2. Case Study 1 Evaluation Results and Discussion

Evaluations of the participants of case study 1 are summarized in Table 10.

According to the answers to the first part, the stakeholders thought that it was easy to understand the framework and easy to carry out the FQAD process due to its guidance given by the assessment model. The comments indicate that there were some difficulties interpreting the sub-characteristics meaning, although most stakeholders did not come up against any problems. Also, some answers in this part show that the stakeholders had some question marks on the interpretation of those sub-characteristics by other stakeholders. The sub-characteristics descriptions can, however, be improved regardless of the method used and possibly improve confidence in the interpretation. Participants stated a need on an automatic results generation structure.

Table 10. Case study evaluation template and results

Level of Validation	Evaluation Criteria	Evaluation Results
Basic	Complete	Functional suitability, Reliability, Extendability characteristics need to be detailed with more sub-characteristics
	Understandable	More details on sub-characteristics are needed. Generally, the framework is understandable.
	Internally consistent	Yes
	Well organised	Yes
	Appropriate for audience	Technical terms not sufficiently described.
	Well written (readable)	Yes
Use	Produced expected results	The results are very strict. Some tolerance can be provided.
	Produced relevant results	Yes
	Produces usable results	Yes. The results point out the missing parts in the DSL.
	Self contained	Suggestion: The assessment can be made in *Excel*, and the results can be shown automatically.
	Procedures understandable	Yes
	Procedures easily implementable	Yes
Gain	Appropriate for task	Yes
	Better than other available guidance	Not answered
	Good support for decision making	Guidance to understand the steps for improvement of the DSL is good. The framework provides a general view to comprehend improvement titles.
	Cost effective	Yes

According to the answers to the second part of the evaluation template, the stakeholders seem to have good confidence in the usefulness of the assessment of the DSL using FQAD. One stakeholder thought FQAD was “very useful”. The other stakeholders thought it was “rather useful”.

In this study, the stakeholders were asked to assess a DSL which is used actively in a project. As third part of the evaluation suggests, another way would be to continue with the existing evaluation method within the department. FQAD was supported by all of the stakeholders.

Although the answers indicate some difficulties with the sub-characteristics interpretations, the stakeholders' response to evaluation template shows that they largely agreed with the resulting DSL assessment and application of the method by them.

Table 11 discusses the improvements obtained from the first case study.

Table 11. Case study 1 improvements

Level of Validation	Evaluation Criteria	Improvements
Basic	Complete	One more sub-characteristic is added to the Expressiveness characteristic.
	Understandable	Functional suitability, Reliability, Extendability characteristics and sub-characteristics descriptions are detailed. Performance efficiency characteristic replaced with Productivity and description is changed.
	Appropriate for audience	It is taken into consideration while detailing the characteristic descriptions
Use	Produced expected results	POOR success level is renamed as INCOMPLETE level.
	Self contained	The assessment forms are transferred to an *Excel* tool and formulas are defined to automate the assessment process.

5.2.3. Case Study 2 Evaluation Results and Discussion

Evaluation of the participants of the case study 2 is summarized in Table 12.

We received responses from all 3 participants. The questions can be found in Table 12. The interpretation of the results is presented below.

The participants explained that FQAD defines a clear process. The comments indicate that there were no difficulties interpreting the sub-characteristics meaning, which shows that improvements made as a result of the first case study were useful.

Table 12. Case study evaluation template and results

Level of Validation	Evaluation Criteria	Results
Basic	Complete	*Yes*
	Understandable	*Yes. /But some characteristics may cause misunderstanding.*
	Internally consistent	*Yes.*
	Well organised	*Yes.*
	Appropriate for audience	*Yes.*
	Well written (readable)	*Yes.*
Use	Produced expected results	*No, "Reusability" measurement shouldn't be in "Maintainability" characteristic. Integrability shouldn't be in "Extendability characteristic" /Some misunderstood characteristics may lead to unexpected results*
	Produced relevant results	*Yes.*
	Produces usable results	*No, results are some kind of late feedbacks for DSL implementers /Yes, it gives useful results*
	Self contained	*Yes*
	Procedures understandable	*Yes*
	Procedures easily implementable	*Yes, but it is sometimes hard to implement procedure for DSL not for DSL outputs (i.e code etc.)*
Gain	Appropriate for task	*Yes.*
	Better than other available guidance	*Not answered*
	Good support for decision making	*No, there is decision after implementing DSL Yes*
	Cost effective	*Yes*

The stakeholders found that the results closely reflected their opinion on what was important. But there was a strict criticism on the reusability sub-characteristic of the DSL where it was handled under the Maintainability characteristic. The participant suggested the reusability sub-characteristic should be defined as a separate characteristic. This improvement suggestion is found meaningful by the researcher and

the related improvement is made in the final FQAD. The other stakeholders thought FQAD was “very useful”.

The new method FQAD was supported by all of the stakeholders. But an important criticism was that the results constituted late feedback. That is, rather than a-posteriori assessment of an implemented DSL, designated quality characteristics would be very useful in the beginning of the DSL development process, that is a-priori, and would guide the DSL developers during the process. The DSL developer can use the characteristics and related support levels to develop the DSL. In this way the developers focus on the needed characteristics instead of developing a perfect DSL. Since FQAD can be used in the beginning of the DSL development process this comment is well appreciated.

Although the answers indicate some difficulties with the timing of the application of the framework, the stakeholders’ response to evaluation template shows that they largely agreed with the resulting DSL assessment and their application of it.

The DSL characteristics improved with the first case study were also improved according to the validatory case study. Reusability and integrability sub-characteristics were defined as characteristics and the resulting evaluation form is presented in Appendix A. The validatory case indicates the resulting DSL quality characteristics and assessment model.

Final note: We have started with case study 1 with exploratory purposes and updated and matured FQAD based on this case study. Although we have performed one more case study for validatory purposes, we believe that the number of case studies could have been increased if time and resources would permit because there may be some other factors still waiting to be explored. Despite these limitations, we are confident that this study has met our research objectives and it presents a comprehensive framework for DSL assessment.

CHAPTER 6

CONCLUSION

6.1. Introduction

The objective of this study was to propose a framework that provides quality characteristics and sub-characteristics for DSLs that conform with ISO/IEC 25010 standard and literature, and provides a qualitative method for the measurement of the success level of the DSLs according to different perspectives. The two research questions were:

1. What are the quality characteristics of DSL success?
2. How can an evaluator measure the success of a DSL using measures aligned with stakeholder perspectives?

The goals of the study have been achieved, as follows:

- A framework for qualitative assessment of DSLs is developed (Chapter 3). This framework provides quality characteristics and sub-characteristics for DSLs that conform with ISO/IEC 25010 standard and literature and provides a qualitative method for the measurement of the success level of the DSLs according to different perspectives.

- Even though Chapter 3 theoretically answers the research questions, empirical research was performed to validate the claims of the framework. Two case studies were conducted to explore and mature various aspects of the framework (Chapter 5). In these case studies, the findings were analyzed with the aim of maturing and enhancing the framework.

Both research questions have been answered in the form of a framework that consists of ten characteristics and 26 sub-characteristics of DSL quality as well as a model for DSL success assessment using quality characteristics balanced according to an evaluator perspective.

The following sections are structured to conclude the thesis. In section 2 the contributions of this research to the body of DSL assessment knowledge is summarized and limitations of the study are discussed in section 3. In section 4 areas for further research are elaborated.

6.2. Research Contributions

The current study provides a deep consideration of the conceptual and operational issues of domain specific languages assessment. Both the academic community and practitioners will benefit from the research contributions.

To evaluate the contributions of this study to the domain specific languages domain, two types of approach is taken in this section. One is to look at the findings from a theoretical perspective and the other through a practical perspective and both of these are evident in these sub-sections.

The exploratory case 1 has contributed to our research from two perspectives: (a) initial infrastructure of our assessment framework is verified, (b) the quality characteristics of the success of DSLs, initially created through an extensive literature review, are updated to be used for the subsequent validatory case. This initial framework can be used for other researchers since it actually is a summary of the DSL assessment literature.

The validatory case 2 has contributed to our research from two perspectives: (a) one of the largest defense industry company's opinions on FQAD was quite important since their systems design and implementation work is heavily software-intensive (b) we realize that a-priori usage of FQAD before the DSL development can be emphasized in our framework. On the other hand, it is confirmed that for DSL assessment FQAD has a visible advantage on the present, essentially ad-hoc assessment methods used.

6.2.1. Theoretical Contribution

The main theoretical contributions lie in the understanding of the conceptual foundation of DSL assessment.

The most important contribution of the research is the presentation of a comprehensive framework for DSL assessment. The research pointed out the need for a comprehensive model that can do DSL assessment, and quality characteristics using contextual information where the quality characteristics can be aligned up to the highest level of goals. Thus this is an attempt to unify and bring together the exercises done in the field of DSL assessment, information systems assessment, and software product quality. The contribution of this research to concept development can be seen in the introduction of the "a framework for qualitative assessment of domain specific language" (see chapter 3), and in the introduction of the "assessment methodology" (see chapter 4). A detailed list of domain specific language quality characteristics was elaborated and a novel assessment method was proposed. The two in-depth case studies provided rich insight into the DSL assessment field.

Critical review of the literature:

The existing literature has been reviewed from a different perspective emphasizing on the need for "understanding" and "comparing" the three contexts of success: (1) domain

specific languages, (2) software product quality, and (3) information systems. This has provided a solid starting point for new researchers in the area.

Development of theoretical constructs:

The conceptual framework proposed in this study has provided a solid basis for the DSL assessment. In addition, the proposed conceptual approach extends previous work on domain specific language success assessment by focusing on the ISO/IEC 25010 standard software product characteristics and extending and adapting those characteristics for DSL assessment. The experience acquired can assist in the understanding of DSL assessment and the understanding of the processes related with DSL success as well as understanding of the assessment roles assigned and performed by different stakeholders.

This research can further facilitate the design and implementation of methodological approaches on software development to fill in the gaps in the assessment of these approaches in general.

6.2.2. Practical Contribution

For the managers and domain specific language stakeholders who often face domain specific language assessment, practical contributions are presented. In general, they would benefit from the FQAD through a deep understanding of the quality characteristics related with domain specific language success.

Indications of the positive practical contributions derived from the feedback received from the case study participants. For the Case I, the findings were perceived as complementary to their evaluation methods performed. In the second case, it is stated that this study will be beneficial to the DSL developers a-priori in future DSL development attempts such that goals and critical characteristics addressed shall be taken into account.

6.3. Limitations

6.3.1. Limitations of the Research Paradigm

The suitability of this research philosophy as well as the suitability of qualitative research for the investigation of software systems evaluation has been justified extensively in Chapter 4. However, the interpretive research philosophy has its own drawbacks.

This study focused particularly in the success of the domain specific languages. Responsibilities of the case study participants are inherently intangible, and the information they provide in the interviews may have been influenced by other factors for which the researcher was uninformed. Objectivity may be affected because of this.

All the weaknesses of the used research philosophy were known in advance, and experience and recommendations from previous interpretive studies (Kitchenham et al.

1997; Pfleeger and Kitchenham 2001; Runeson and Host 2009) were taken into consideration to overcome them. In addition, the research design (see section 4.3) was carefully elaborated.

6.3.2. Limitations of the Work Done

We have started our research with formulating our research objective and research questions. Next, an extensive literature review has been performed. The case selections were carefully made and a great amount of qualitative data was collected through forms and interviews. The research findings, as a result of these case studies, are supported by the literature review for validity purposes.

On the other hand, the interviews and the findings show that due to the nature of the success assessment, the existence of different perspectives is the major limitation for all similar types of research. Human factors cause and form the level of uncertainty. We have tried to handle this uncertainty by using an assessment model that captures different viewpoints.

Although the cases - the organizations and the subjects we interviewed - were designed after a careful selection process, the results may still contain some level of bias since all of the cases were success cases. The interviewees may have had a positive interpretation of the DSL experiences in their department.

The final framework is the result of two case studies. These cases aided in exploring and evaluating the approach proposed based on the literature review. However the results of these two case studies are not intended to generalize the findings further.

It is important to re-visit at this point the subject of generalization of the results of the case studies reported in the present study, which has already been discussed in chapter 4. Generalization of the case study results does not aim to generalize from one case to another but to generalize a particular set of results to some broader theory (Runeson and Host 2009;Yin 2003). In this sense, the aim of the present study was to identify from the case studies investigated the issues that seemed to effect DSL assessment, so that new DSL characteristics and an improved assessment model could be contributed to enrich relevant theory. The conclusions made in this section are the result of the attempt to generalize the findings of the case studies undertaken for this thesis into the broader theory of DSL assessment. Broadly, the issues deal with two research questions: on the one hand they concern specific difficulties to determine DSL quality characteristics; on the other hand they aim to address the question of how to use those characteristics in the assessment of a DSL. The discussion is structured under FQAD that is proposed in this thesis in an attempt to begin a process of refinement of the concepts of DSL assessment that will shed light on their future application.

6.4. Future Work

The limited number (two) of case studies restricted the general validity of the conclusions. This research study can be enriched with application of the framework on

other domain specific languages. This could be the beginning of a stream for further research that would concentrate on exploring the situation of the success of domain specific languages. Such a further work could strengthen the findings of the domain specific languages assessment framework proposed in this study.

The automated assessment model described in this study can be sent to several organizations covering more countries along with a survey. A more convenient methodology is to create a website for the purpose, with comprehensive details of the framework and its purpose, along with a downloadable link to the automated assessment model, with an online survey form incorporated into the website.

Based on the proposed DSL quality characteristics, alternative assessment approaches may be attempted and evaluated. For example a positivistic quantitative research methodology may be applied, examining the correlations of the quality characteristics in similar organizations and comparing the findings with the present results. Such a study may possibly point out issues of generalizability of our findings and enable remedies.

In addition, further development of the idea of "quality assessment of DSLs" would be a possible research area. Such research may be focused specifically on "DSL quality characteristics". This could include the enhancement of the quality characteristics of the framework investigated here.

While a number of artefacts that may be investigated in the course of the assessment process have been suggested and their use demonstrated in the case studies, an explicit list has not been enunciated. Further work on establishing objective measures based on well-defined artefacts would definitely be beneficial towards strengthening the proposed assessment framework.

6.5. Conclusion

This thesis has investigated the subject of assessment of domain specific language qualitatively. However, DSL success assessment is complicated with many conceptual and operational difficulties.

The research and the subsequent framework that have emerged shows that the framework will aid an evaluator's ability to perform a DSL assessment. This research leads towards a comprehensive framework for continuous improvement and alignment of the DSL with the software development goals.

To conclude, this study was designed to be only a step in the field of ever evolving domain specific language success research.

REFERENCES

Allen N. A., Shaffer C. A., Watson L. T., *Building modeling tools that support verification, validation, and testing for the domain expert*, in WSC '05: Proceedings of the 37th conference on Winter simulation, Orlando, Florida, pp. 419-426, 2005

Amstel, M.,F., Lange, C.F., Brand, M.G., *Using Metrics for Assessing The Quality of ASF+SDF Model Transformations*, ICMT 2009, pp.239-248, Springer, 2009

Benbasat, I., Goldstein, D.,K., Mead, M., *The Case Research Strategy in Studies of Information Systems*, MISQ 11(3):369-386, doi:10.2307/248684, 1987

Benyon, D., *Information and Data Modelling*, McGraw-Hill, second edition, Wokingham, 1997

Blaikie, N., *Designing Social Research*, Malden:Blackwell Publishers Ltd.,2000

Boehm B.W., *Software Engineering Economics*, Englewood Cliffs, NJ:Prentice Hall, 1981

Brooks, R., *Studying Programmer Behaviour Experimentally: The Problems of Proper Methodology*, Communications of the ACM, Vol 23, Issue 4, pp207-213, 1980

Bryman, A., *Social Research Methods*, New York:OxfordUniversity Press, 2004

Cameron K.S., *Critical Questions in Assessing Organizational Effectiveness*. Organizational Dynamics,. 9(2) 66-80, 1980.

Cameron, K. S.*The effectiveness of ineffectiveness*. Research in Organizational Behavior, 6, 235-285, 1984

Cameron, K. S., Whetten, D. A., *Some Conclusions about Organisational Effectiveness*, Organisational Effectiveness: A Comparison of Multiple Models, pp. 261-277, New York: Academic Press, 1983

Caputo K.,*CMM Implementation Guide, Choreographing Software Process Improvement*, Addison Wesley, 1998.

Chen Y., Dios R., Mili A., Wu L.,*An Empirical Study of Programming Language Trends*, New Jersey Institute of Technology, Kefei Wang, State University of New York, Albany ,IEEE- 2005

Cresswell, J. W., *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches*, Thousand Oaks: Sage Publications, 2003

Curtis, B., Soloway. E., Brooks, R., Black, J., Ehrlich, K., Ramsey, H.R., *Software Psychology: The Need for an Interdisciplinary Program*, Proceedings of the IEEE, 1986

de Vaus, D.A., *Research Design in Social Research*, Thousand Oaks, CA:SAGE, 2001

Easterbrook, S., Singer, J., Storey, M.A., Damian, D., *Selecting Empirical Methods for Software Engineering Research, Guide to Advanced Empirical Software Engineering*, Springer, 2007

Flyvbjerg, B., *Five misunderstandings about case-study research*, In Qualitative Research Practice: Concise Paperback Edition. Sage, pp 390–404, 2007

Fowler, M., *Domain Specific Languages, Addison Wesley Professional*, 2010

Freeze J. *Creating DSLs with Ruby*. *Ruby Code & Style*. Artima, Inc.: Mountain View, CA,Published online:http://www.artima.com/rubycs/articles/ruby as dsl.html, 2006

Furuta R., Kemp P.M., *Experimental Evaluation of Programming Language Features:Implications for Introductory Programming Languages*, ACM SIGCSE Bulletin-Proceedings of the 10th SIGSCE symposium on Computer Science Education, vol 11,issue 1, 1979

Gabriel, P.H.N., *Software Langauges Engineering: Experimental Evaluation*, Dissertacao apresentada na Faculdade Ciencias e Tecnologia da Universidade Nova de Lisboa para obtencao do grau de Mestre em Engenharia Informatica, Lisboa, 2010

Haugen, O., Mohagheghi, P., *A Multi-dimensional Framework for Characterizing Domain Specific Languages*, Proceeding of the 7th OOPSLA Workshop on Domain Specific Modeling, 2007

Hermans, F., Pinzger, M., Deursen, A.v., *Domain Specific langages in Practice: A User Study on the Success Factors*, In MODELS'09, pp 423-437, Berlin,Springer-Verlag, 2009

Hoare, C. A. R.,*Hints on programming language design*. Technical Report, Stanford University,Stanford, CA, USA, 1973

Howatt, J.W., *A Project Based Approach to Programming Language Evaluation*, ACM SIGPLAN Notices, 30(7), 37-40, 1995

Hudak P. *Building domain-specific embedded languages*. ACM Computing Surveys, 28,1996

ISO/IEC 25010:2011: Systems and Software Engineering – Systems and Software Quality Requirements and Evaluation (SQuaRE) – System and Software Quality Models, International Standards Organization/International Electrotechnical Commission, 2011

ISO/IEC 25022:2012: Systems and Software Engineering – Systems and Software Quality Requirements and Evaluation (SQuaRE) – Measurement of Quality in Use, International Standards Organization/International Electrotechnical Commission, 2012

Kahlaoui, A., Abran, A., Lefebvre, E., *DSML Success Factors and Their Assessment Criteria* , Metrics News, vol. 13, no 1, p. 43-51, 2008

Kaplan, B., Duchon, D., *Combining Qualitative and Quantitative Methods in Information Systems Research: A Case Study*, MIS Quarterly, 12(4), 571-587, 1988

Karna, J., Tolvanen, J.P., Kelly, S., *Evaluating the Use of Domain-Specific Modeling in Practice*, In Proceedings of DSM09, 2009

Karsai, G., Krahn, H., Pinkernell, C., Rumpe, B., Schindler, M., Völkel, S.:*Design Guidelines for Domain Specific Languages.*In: Proceedings of the 9th OOPSLA

Workshop on Domain-Specific Modeling (DSM' 09), Orlando, Florida, USA, 2009

Kelly S., Pohjonen R., *Worst Practices for Domain-Specific Modeling*,IEEE Software, vol. 26, pp. 22-29, July/August, 2009.

Kelly S, Tolvanen J. *Domain-specific Modeling*: *Enabling Full Code Generation*. Wiley: New York, 2008.

Khedker, U.,P., *What Makes a Good Programming Language?*, Technical Report TR-97-upk-1,Department of Computer Science University of Pune, 1997

Kitchenham, B.A., Linkman, S., Law, D., DESMET: *A Methodology for Evaluating Software Engineering Methods and Tools*, Computing and Control Engineering Journal, 120-126, 1997

Kitchenham, B., Pfleeger, S., L., Principles of Survey Research Part 6: Data Analysis, Software Engineering Notes ,vol 28, no 2, ACM Sigsoft, 2003

Kolovos, D.S., Paige, R.F. Kelly, T.P., Polack, F.A.C., *Requirements for Domain-Specific Languages*, In Proceedings of the First ECOOP Workshop on Domain-Specific Program Development, co-located with ECOOP'06, Nantes, France, 2006.

Kosar, T.,Mernik, M., Crepinsek, M., Henriques, P.R., Cruz, D., Pereira, M.J.V., Oliviera, N.,*Influence of Domain Specific Notation to Program Understanding*, Proceedings of the International Multiconference on Computer Science and Information Technology, 2009

Kosar, T., Oliviera, N., Mernik, M., Pereira,V.M.J., Crepinsek, M., Cruz, D., Henriques, P.R., *Comparing General Purpose and Domain Specific Languages: An Empirical Study*, ComSIS, Vol.7, No.2, Special Issue, April 2010

Krakower J.Y., *Assessing Organizational Effectiveness: Considerations and Procedures*, Boulder, CO : National Center for Higher Education Management Systems, 1985

Kouhen, A.E., Dumoulin, C., Gerard, S., Boulet, P., Evaluation of Modeling Tools Adaptation, hal-00706701, version 2, 2012

Krogstie, J., *Evaluating UML Using a Generic Quality Framework*,Chapter in UML and the Unified Process, Idea Group Publishing, pp.1-22, 2003

Kulkarni P., Kailash H.D., Shankar V., Nagarashan S., N.L. Goutham, *Programming Languages: A Comparative Study*, 2008

Lee, A.S., *A scientific methodology for MIS case studies*, MIS Q 13(1):33–54 doi:10.2307/248698, 1989

Mellor, S. J., Scott, K., Uhl, A., Weise, D.,*MDA Distilled*, Addison Wesley, 2004

Mernik, M., Heering, J., Sloane, A., *When and How to Develop Domain-Specific Languages*, ACM Computing Surveys, 2005

Microsoft. Microsoft modeling platform (code named Oslo). Available at: http://msdn.microsoft.com/enus/library/cc709420.aspx, 31 October 2008

Miller L.A., *Programming by Non-programmers*, Int.J. Man-Machine Studies, vol 6, pp 237-260, 1974

Merilinna, J., Parsinnen, J., *Comparison Between Different Abstraction Level Programming: Experiment Definition and Initial Results*, In OOPSLA Workshop on Domain-Specific Modeling, Montreal, Canada, 2007

McKean, D., Sprinkle, J., *Heterogeneous Multi-core Systems: UML Profiles vs. DSM Approaches*, Proceedings of the 2012 workshop on Domain Specific Modeling, 2012

Mohagheghi, P., Dehlen, V., Neple, T., *Definitions and Approaches to Model Quality in Model Based Software Development – A Review of Literature*, Information and Software Technology, vol. 51, pp.1646-1669, 2009

Myers B. L.., *Information Systems Assessment:Development of a Comprehensive and Contingency Theory to Assess The Effectiveness Of the Information Systems Function*, Dissertation, University of North Texas, August 2003

Myers, M., *Qualitative Research in Information Systems*, MIS Quarterly, 21(2), pp.241-242, 1997,

Myers, M.D., Avison, D.E., *An Introduction to Qualitative Research in Information Systems, In M.D.Myers,D.E.Avison(Eds), Qualitative Research in Information Systems –A Reader*, London:Sage Publications, 2002

Naiditch D., *Selecting a Programming Language for Your Project*,Raytheon Systems Company, IEEE- 1999

Oliveira, N., Pereira, M.,J.,V., Henriques, P., R., Cruz., D, *Domain specific languages: A theoretical survey*, INForum'09 - Proceedings, Lisbon, Portugal, The University of Lisboa Faculty of Sciences, 2009

ORMSC, *Model Driven Architecture (MDA)*, OMG, 2001

Overbeek, J., *Meta Object Facility (MOF) - investigation of the state of the art*, Master's thesis, University of Twente, 2006

Özkan S., *PB-ISAM: A Process Based Framework for Information Systems Effectiveness Assessment in Organisational Contexts*, Dissertation, The Department of Information Systems, January 2006

Paige, R., Ostroff, J., Brooke, P., *Principles for Modeling Language Design*, Techical Report CS-1999-08, York University, 1999

Parker K.R., Chao J.T., Ottaway T.A., Chang J., *A Formal Language Selection Process for Introductory Programming Courses*, Journal of Information Technology Education, vol. 5, 2006

Parker K.R., Ottoway T.A.., Chao J.T., *Criteria for the Selection of a Programming Language for Introductory Courses*, International Journal of Knowledge and Learning, 2006

Pfleeger, S.L., Kitchenham, B. A., *Principles of survey research*, ACM SIGSOFT Software Engineering Notes, vol. 26, pp. 16-18, 2001

Punch, K.F., *Introduction to Social Research: Quantitative and Qualitative Approaches*, London:Sage Publication,2008

Robson, C., *Real World Research*, Blackwell, (2nd Edition), 2002

Rombach, H.D., *A Framework for Assessing Process Representations*, Proceedings of the 6th International Software Process Workshop, IEEE Computer Society Press, Japan, 1990

Rowley, J., *Using Case Studies in Research*, Management Research News, vol.25, no.1, pp.16-27, 2002

Runeson, P., Höst, M., *Guidelines for Conducting and Reporting Case Study Research in Software Engineering*, Empirical Software Engineering, Springerlink, p131-164, doi:10.1007/s10664-008-9102-8, 2009

Sackman H., Erickson W.J., Grant E.E.,*Exploratory and Experimental Studies in Comparing Online and Offline Programming Performance*, Communication of the ACM, vol 11, pp 3-11, 1968

Sánchez-Ruíz, A. J., Saeki M., Langlois B., *Domain-Specific Software Development Terminology: Do We All Speak the Same Language?*, in OOPSLA Workshop on Domain-Specific Modeling, Jacksonville, Florida, USA, 2006.

Strembeck, M., Zdun, U., *An Approach For The Systematic Development of Domain Specific Languages*, Software-Practice and Experience 39:1253-1292, John Wiley & Sons,Ltd, 2009

Software Engineering Institute, *CMMI for Acquisition, Version 1.3*, Technical Report, CMU/SEI-2010-TR-032, Carnegie Mellon, November 2012

Somekh, B., Lewin, C., Elementary Quantitative Methods, In;Research Methods in the Social Sciences, Chapter 25, SAGE Publications, 2005

Tokdemir, G., *An Assessment Model For Web-Based Information System Effectiveness*, Dissertation, The Department of Information Systems, METU, January 2009

Wand, Y., Weber, R., *On the Ontological Expressiveness of Information Systems Analysis and Design Grammars*, Journal of Information Systems, 3:217-237, 1993

Wand, Y.; Weber, R. *On the deep structure of information systems*, Information SystemsJournal (5), pp. 203-223, 1995

Watt, D.A., *Programming Language Design Concepts.* John Wiley & Sons, 2004

Wirth N.,*On the Design of Programming Languages*.In IFIP Congress, pages 386–393, 1974.

Wu, Y., Hernandez, F., Ortega, F., Clarke, P.J., France, R., *Measuring the Effort for Creating and Using Domain Specific Models*, In: Proceedings of 10th Domain Specific Modeling Workshop, 2010

Yin, R.,K., *Case Study Research Design and Methods*, 3rd edition, London, Sage, 2003

APPENDIX A

FQAD FORMS

A.1. Form I – Characteristics Importance Ranking

DSL Success Assessment Questionnaire

This study is designed to assess the success of a DSL.

DSL Quality Characteristics Importance Ranking

Choose your choice of DSL quality characteristics importance *(Mandatory, Desirable, Nice to have)* according to your goal for success assessment

	DSL Quality Characteristics	Description	Importance Ranking
1.	Functional suitability	Functional suitability of a DSL refers to the degree to which a DSL supports developing solutions to meet stated needs of the application domain.	Please make a choice
2.	Usability	Usability of a DSL is the degree to which a DSL can be used by specified users to achieve specified goals	Please make a choice
3.	Reliability	Reliability of a DSL is defined as the property of a language that aids producing reliable programs (model checking ability/preventing unexpected relations)	Please make a choice
4.	Maintainability	The degree to which a language promotes ease of program maintenance	Please make a choice
5.	Productivity	Productivity of a DSL refers to the degree to which a language promotes programming productivity Productivity is a characteristic related to the amount of resources expended by the user to achieve specified goals	Please make a choice
6.	Extendability	The degree to which a language has general mechanisms for users to add new features	Please make a choice
7.	Compatibility	The degree to which a DSL is compatible to the domain and development process.	Please make a choice
8.	Expressiveness	The degree to which a problem solving strategy can be mapped into a program naturally	Please make a choice
9.	Reusability	The degree to which a language constructs can be used in more than one language	Please make a choice
10.	Integrability	The degree to which a language constructs can be used in more than one language	Please make a choice

Figure 8. Characteristics Important Ranking Form

A.2. Form II – Sub-characteristic Assessment Statements

DSL Success Assessment Questionnaire

Please give marks to every sentence below for the assessment of the DSL.

State how much you agree to each sentence by ticking the appropriate choice.

	DSL Success Quality Measures	Support Level
	Functional Suitability	
1.	All concepts and scenarios of the domain can be expressed in the DSL (completeness)	Make A Choice
2.	DSL is appropriate for the specific applications of the domain (e.g. to express an algorithm) (appropriateness)	Make A Choice
	Usability	
3.	The required amount of effort for understanding the language is small (comprehensibility)	Make A Choice
4.	The concepts and symbols of the language are easy to learn and remember (learnability)	Make A Choice
5.	Language has capability to help users achieve their tasks in a minimum number of steps	Make A Choice
6.	Users can recognize whether the DSL is appropriate for their needs (likeability,user perception)	Make A Choice
7.	DSL has attributes that make it easy to operate and control the language (operability)	Make A Choice
8.	DSL has symbols that are good-looking (attractiveness)	Make A Choice
9.	The printed page representing a DSL model takes up little space (compactness)	Make A Choice
	Reliability	
10.	DSL protects users against making errors. The DSL avoids the user to make mistakes. (model checking)	Make A Choice
11.	DSL includes right elements and correct relations between them(DSL prevents the unexpected interactions between its elements) (correctness)	Make A Choice
	Maintainability	
12.	The amount of effort required for modifying the DSL to provide different or additional functionality is small (modifiability)	Make A Choice
13.	DSL is composed of discrete components such that a change to one component has minimal impact on other components (Low coupling)	Make A Choice
	Productivity	
14.	The development time of a program to meet the needs is improved	Make A Choice
15.	The amount of human resource used to develop the program is improved	Make A Choice
	Extendability	
16.	DSL has general mechanisms for users to add new features (adding new features without changing the original language)	Make A Choice
	Compatibility	
17.	DSL is compatible to the domain. DSL has capability to operate with other elements of the domain with no modification required to perform a specific application in the domain.	Make A Choice
18.	Using DSL to develop models fits in the development process, since it is used as part of a development process with phases and roles.	Make A Choice
	Expressiveness	
19.	A problem solving strategy can be mapped into a program easily	Make A Choice
20.	The DSL that provides one and only one good way to express every concept of interest (unique)	Make A Choice
21.	Each DSL construct is used to represent exactly one distinct concept in the domain (orthogonal)	Make A Choice
22.	The language constructs correspond to important domain concepts. DSL does not include domain concepts that are not important.	Make A Choice
23.	DSL does not contain conflicting elements.	Make A Choice
24.	DSL is at the right abstraction level such that it is not more complex or detailed than necessary	Make A Choice
	Reusability	
25.	The symbols and other elements of the DSL can be used in more than one DSL, or in building other language elements. Using the definition of a language as a beginning to develop a new one (Reusability)	Make A Choice
	Integrability	
26.	DSL can be integrated with other language used in development process. (language integrability with other languages)	Make A Choice

Figure 9. Sub-characteristic Assessment Statements Form

A.3. Form III – Automatically Generated Assessment Results

DSL Success Assessment Results		
Functional Suitability		
	Completeness	INCOMPLETE
	Appropriateness	INCOMPLETE
Usability		
	Comprehensibility	INCOMPLETE
	Learnability	INCOMPLETE
	Language helps users achieve their tasks in a minimum number of steps	INCOMPLETE
	Likeability,user perception	INCOMPLETE
	Operability	INCOMPLETE
	Attractiveness	INCOMPLETE
	Compactness	INCOMPLETE
Reliability		
	Model checking ability	INCOMPLETE
	Correctness	INCOMPLETE
Maintainability		
	Modifiability	INCOMPLETE
	Low coupling	INCOMPLETE
Productivity		
	The development time improvement	INCOMPLETE
	The amount of human resource used improvement	INCOMPLETE
Extendability		
	Mechanisms for users to add new features	INCOMPLETE
Compatibility		
	DSL is compatible to the domain	INCOMPLETE
	Using DSL to develop models fits in the development process	INCOMPLETE
Expressiveness		
	A problem solving strategy can be mapped into a program easily	INCOMPLETE
	Uniqueness	INCOMPLETE
	Orthogonality	INCOMPLETE
	The language constructs correspond to important domain concepts.	INCOMPLETE
	DSL does not contain conflicting elements.	INCOMPLETE
	DSL is at the right abstraction level	INCOMPLETE
Reusability		
	Reusability	INCOMPLETE
Integrability		
	Integrability	INCOMPLETE
OVERALL SUCCESS OF THE DSL		INCOMPLETE

Figure 10. Automatically Generated Assessment Results Form

CURRICULUM VITAE

PERSONAL INFORMATION

Surname, Name: Kahraman, Gökhan

Nationality: Turkish (TC)

Date and Place of Birth: 21 January 1978, Ordu

Marital Status: Married

Phone: +90 537 499 85 22

Email: gokhank@aselsan.com.tr

EDUCATION

Degree	Institution	Year of Graduation
MS	H. U. Electrical and Electronics Engineering	2000
BS	H. U. Electrical and Electronics Engineering	2004
High School	Trabzon Yomra Fen Lisesi	1996

WORK EXPERIENCE

Year	Place	Enrollment
2000- Present	ASELSAN	Senior Expert Engineer

FOREIGN LANGUAGES

Advanced English

PUBLICATIONS

1. Kahraman, G., Bilgen, S., “Alana Özgü Diller için Bir Etkinlik Değerlendirme Çerçevesi”, 6. Ulusal Yazılım Mühendisliği Sempozyumu, Ankara 2012
2. Kahraman, G., “Gömülü Kontrol Yazılımlarında İşlev Katmanının Referans Model Oluşturmak için Yeniden Yapılandırılması Yaklaşımı”, 4. Ulusal Yazılım Mühendisliği Sempozyumu, İstanbul 2009
3. Kahraman, G., Bilgen, S., “Alana Özgü Diller için Bir Etkinlik Değerlendirme Çerçevesi”, MASFOR Poster Session 1, İstanbul 2012

4. Tekinerdoğan, B., Seker, Ş., Kahraman, G., Tekkalmaz, M., Erdoğan, Ö.Ö., "Gömülü Sistemler için Yazılım Mimari Çerçevesi", 7. Ulusal Yazılım Mühendisliği Sempozyumu, İzmir 2013