Softw Syst Model (2015) 14:1505–1526
DOI 10.1007/s10270-013-0387-8

REGULAR PAPER

A framework for qualitative assessment of domain-specific languages

Gökhan Kahraman · Semih Bilgen

Received: 5 March 2013 / Revised: 24 October 2013 / Accepted: 1 November 2013 / Published online: 23 November 2013

Abstract Domain-specific languages (DSLs) are used for improving many facets of software development, but whether and to what extent this aim is achieved is an important issue that must be addressed. This paper presents a proposal for a Framework for Qualitative Assessment of DSLs (FQAD). FQAD is used for determining the perspective of the evaluator, understanding the goal of the assessment and selecting fundamental DSL quality characteristics to guide the evaluator in the process. This framework adapts and integrates the ISO/IEC 25010:2011 standard, CMMI maturity level evaluation approach and the scaling approach used in DESMET into a perspective-based assessment. A detailed list of domain-specific language quality characteristics is elaborated, and a novel assessment method is proposed. Two case studies through which FQAD is matured and evaluated are reported. The case studies have shown that stakeholders find the FQAD process beneficial.

Keywords Domain-specific languages · Quality measures · Qualitative assessment · ISO/IEC 25010 · CMMI

1 Introduction

Domain-specific languages (DSL) are used for improving many facets of software development, but the measurement of those improvements and whether and to what extent DSLs provide desired benefits are important issues that must be addressed [11]. Gabriel [6] has compiled a survey on the assessment of DSLs and points out the absence of a systematic assessment of the languages. The growing number and complexity of DSLs raise the necessity of a systematic approach for assessment of DSLs [13,29].

This paper focuses on the important challenge within DSLs of how to determine the specific quality characteristics relevant to the success of a DSL 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 paper from various others [7,8,11,14,17–19,21,32] 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 framework for decision support. The aim of this study is to assess the DSL, in order that the DSL

Communicated by Dr. Oystein Haugen.

G. Kahraman (✉) · S. Bilgen
Department of Electrical and Electronics Engineering,
Middle East Technical University, Ankara, Turkey
e-mail: gokhank@aselsan.com.tr

S. Bilgen
e-mail: semih-bilgen@metu.edu.tr

meet stakeholders' goals, by increasing their satisfaction. 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.

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 field of real-time software engineering and hardware/software systems integration.

The paper is structured as follows. Section 2 sets the theoretical background. Section 3 describes FQAD and outlines the DSL assessment process. The case studies are presented in Sect. 4. The main findings from the questionnaire-based evaluation of the FQAD case studies are summarized in Sect. 5. Finally, Sect. 6 presents a conclusion on the findings and identifies issues for further research.

2 Theoretical background

In this section, we summarize previous approaches, compare them to our framework and detail the concepts that form the foundation of FQAD.

2.1 Related work

A number of studies have addressed specific quality characteristics for DSLs. Haugen et al. [7] present a structured questionnaire based on three dimensions of a DSL—expressiveness, transparency, formalization. Merilinna and Pärssinen [21] investigate the benefits of using DSLs by making comparisons between the DSL approach and traditional approaches experimentally. Kosar et al. [18] 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. [8] 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. [32] 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. [17] list the core quality requirements for a DSL. Kahlaoui et al. [10] emphasize domain specificity and code generation ability as specific requirements on DSLs. Based on real DSL development cases, Kelly and Pohjonen [14] discuss worst practices for creating domain-specific modeling (DSM) languages which developers should avoid. Kärna et al. [11] 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 [22] present criteria that will help in selecting a DSM or any other approach to use in system development. Gabriel [6] 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. Frank [5] presents a study to support the process of designing modeling languages by analyzing requirements to specify and evaluate a DSL.

Comparison type studies [11,18,21] 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 for creating modeling tools are called metaCASE tools and popular ones include: MetaEdit+, Obeo Designer, GMF [19]. 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 [30,31] created a quality framework based on Bunge's [2] ontology and it is known as the BungeWandWeber (BWW) framework. An ontological evaluation is based on two mappings [30]: First, a representation mapping describes whether and how the constructs of the BWW-model are mapped onto the grammatical constructs. Second, the interpretation mapping describes whether and how the grammatical constructs are mapped onto the constructs of the BWW-model. Four ontological deficiencies can be identified: incompleteness, redundancy, excess and overload [30].

The studies [7,8,11,14,17–19,21,32] 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 is on specific technical issues, whereas we aim to assess what is relevant for different stakeholders.

2.2 Foundations

In this paper, 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 [28] and the scaling

approach used in DESMET [16] into a perspective-based assessment.

Strembeck and Zdun [29] 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.

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 and tailored for DSLs. To determine DSL quality characteristics, ISO/IEC 25010:2011 standard is used as a major reference, together with the computer language assessment literature. ISO/IEC 25010:2011 standard revises ISO/IEC 9126:2001 and provides a model for the quality of software systems 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. Based on the hierarchical model of ISO 25010, quality of a DSL is expressed in terms of DSL quality characteristics and sub-characteristics.

The maturity level evaluation approach used in CMMI [28] framework is adapted to assess DSLs in this study. Capability maturity models (CMMs) focus on improving processes in an organization. They 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 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.

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.

Measures of DSL assessment can be gathered by objective means, such as the measurement of the frequency of occurrence of particular events. Alternatively, data may be gathered from the subjective responses of the users. Objective measures provide direct indications of effectiveness and efficiency, while subjective measures can be related to satisfaction.

While defining DSL success levels, the DESMET approach is used and levels defined in [16] are adapted to cover the concepts in the DSL assessment domain.

The effectiveness assessment approach presented in [3] is adopted in the present study. In Cameron [3], 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.

3 The FQAD

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. We are making use of ISO/IEC 25010 here outside its original scope of reference, and with changes, deletions and additions. However, the framework we thus construct is, as verified by the case studies reported in Sec. 4, promising and useful. Furthermore, our choice of assessment questions is largely subjective, as will be the answers to those questions. There is, however, as also shown explicitly via numerous references to the relevant literature, a broad measure of agreement with the experiences and publications of many authors. 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 [12,25]. We use this characteristic to cover correctness, completeness [23], lack of domain understanding [14], incompleteness [30] and domain appropriateness [20].
2. Usability: Usability of a DSL is the degree to which a DSL can be used by specified users to achieve speci-

fied goals. A DSL should be as simple as possible in order to express the domain concepts and to support its users [12,15,17,25]. Using symbols that are too simple or similar or unappealing should be avoided [14]. We consider understandability [1], semantic transparency, cognitive fit, complexity management, perceptual discriminability, visual expressiveness [24], comprehensibility, appropriateness [20], learnability [11], effort for adoption, effort required to build models [22], transparency [7], space economy [25], writability and readability [15] 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 [4,25]. DSL's support for error prevention and model checking is pointed out as a significant quality by Kärna et al. [11].
4. Maintainability: The degree to which a language is easy to maintain. DSLs can be altered and new concepts and concept extensions can be added [14]. Maintainability covers understandability and modifiability in this study [4,7]. Modifiability can be described as the amount of effort required for modifying the DSL to provide different or additional functionality. We consider modularity [1] 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. Extensibility: The degree to which a language has general mechanisms for users to add features [4,7,17]. Scalability [7] is another sub-characteristic that is handled in extensibility.
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 [15]. Expressiveness is pointed out as one of the main characteristics of DSLs in [7]. 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 [15,25,30]. There must be a one-to-one correspondence between concepts and their representation in the language [24]. 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 [14]. 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 [17,30]. Graphic economy [24], conformity [17] and consistency [1] 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: The degree to which a language is amenable to integration with other languages. DSL can be integrated with other languages used in development process [7].

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. Usability can either be specified or measured as a product quality characteristic in terms of its sub-characteristics or specified or measured directly by measures that are a subset of quality in use (ISO/IEC 25010:2011). Satisfaction is referred as a characteristic 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. To be successful, 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, extensibility and 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 [9]. 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 [9]. 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 [11,13,14]. 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 [7,15]. Extensibility and integrability are not handled as a characteristic in ISO/IEC 25010:2011, but they are among the referred characteristics in the literature [4,7,17]. For this reason, extensibility 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 1.

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 be deemed 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.

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.

3.2 Assessment in FQAD

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 framework 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 [16] are taken as reference and via significant modifications, success level determination strategy is defined.

3.2.1 Assessment components

The assessment 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.

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

The assessment components are summarized in Fig. 1 to illustrate their relationships.

The 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 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 component.

In FQAD, the logical order used for assessing a DSL is as follows, evaluator:

1. make an evaluator profile:
   (a) choose which characteristics are to be evaluated
   (b) choose the importance degree of each characteristic
   (c) this gives the minimum support level required for all its sub-characteristics
2. evaluate the DSL
   (a) evaluate each sub-characteristic
   (b) check all sub-characteristics' ratings against their minimum level to determine the DSL success level:
      i. if any sub-characteristic is rated below the minimum support level DSL success level = Incomplete
      ii. if all sub-characteristics are rated exactly equal to the minimum support level DSL success level = Satisfactory

Table 1 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)
	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, language elements are practicable for the specific task and specific language developers)
	Attractiveness	DSL has symbols that are good-looking/attractive (attractive interaction, attractive appearance)
	Compactness	The language provides mechanisms for compactness of the representation of the program
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 reduces user error rates
	Correctness	DSL includes right elements and correct relations between them (DSL prevents unexpected interactions between its elements)
Maintainability		The degree to which a language is easy to maintain
	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
	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

Table 1 continued

Quality characteristics	Sub-characteristics	Description
Extensibility		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 with 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 with 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 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.)
Integrability		The degree to which a language is amenable to integration with other languages
	Integrability	DSL can be integrated with other languages used in development process. (e.g., language integrability with other languages)

iii. if all sub-characteristics are rated above or equal to the minimum support level DSL success level = Effective

3.2.2 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 Fig. 2.

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

3.2.2.2 Step 2 The evaluator gives feedback on the determined quality characteristics as defined in the FQAD. Part II of the 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.

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

3.2.3 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. The details of the levels are given in "Appendix A."

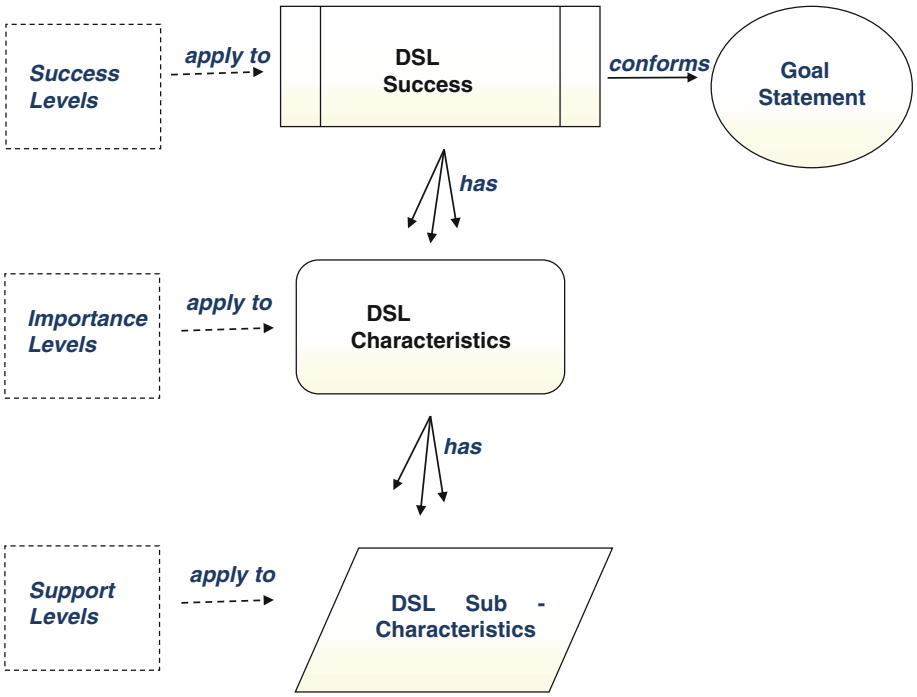

Fig. 1 FQAD assessment components

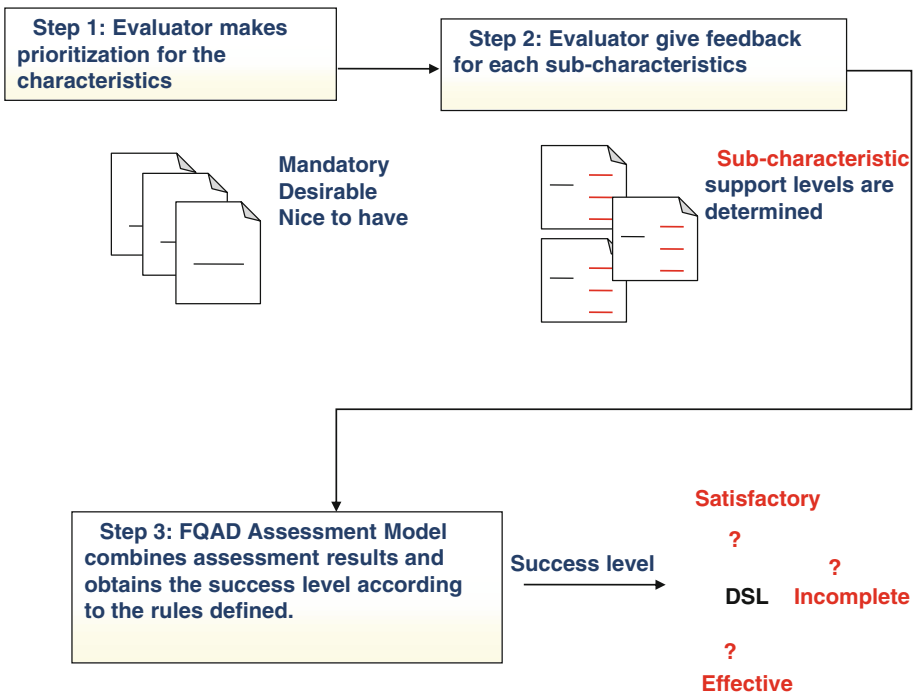

Fig. 2 Assessment steps in FQAD

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.

3.2.4 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 is 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. [16] 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-characteristic that does not possess a mandatory characteristic is, by definition, unacceptable. For this reason, in FQAD assessment, a mandatory sub-characteristic corresponds to having at least strong support level.

The evaluator constructs the meaning of DSL success from his perspective. A one-to-one correspondence is setup between the importance degrees and sub-characteristics support levels as shown in Table 2. 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 rated as some support as minimum level 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.

Table 2 Importance degree versus support level

Importance degree	Minimum required sub-characteristic support level
Nice to have	No support
Desirable	Some support
Mandatory	Strong support
–	Full support

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.

A higher number of grading scales could have been used to reflect the collected information. In the presented three-grade classification, nearly all languages will be graded as incomplete or effective. The satisfactory grading level was added to state simply that the language meets exactly what the evaluator expected. As these three grades simply aim to compartmentalize assessment results in broad categories, all grades will have to be reported together with the details of assessment, and as such, loss of information will be avoided.

An FQAD questionnaire that guides the evaluator in the assessment process and provides a template is developed using the assessment in FQAD ("Appendix"). 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 rated 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.

3.2.5 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.

3.2.5.1 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. Tangible articles are simply something tangible, coming from having the sub-characteristics performed. Sometimes these 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.

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

4 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 method. 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.

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 [27]. Below, first, the goal of the study is stated, and then the selection of the case and the subjects is described. Finally, how the data are collected and analyzed are explained before the discussion on the validity of the process.

4.1 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 method 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.

4.2 Case and subjects selection

In order to answer the case study research question, 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.

The context is considered as 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 3. 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

Table 3 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

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.

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 military projects developing products with high-end software development techniques such as 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 are issued with updates of the DSL. Thus, at the time of assessment, 3 releases had been issued.

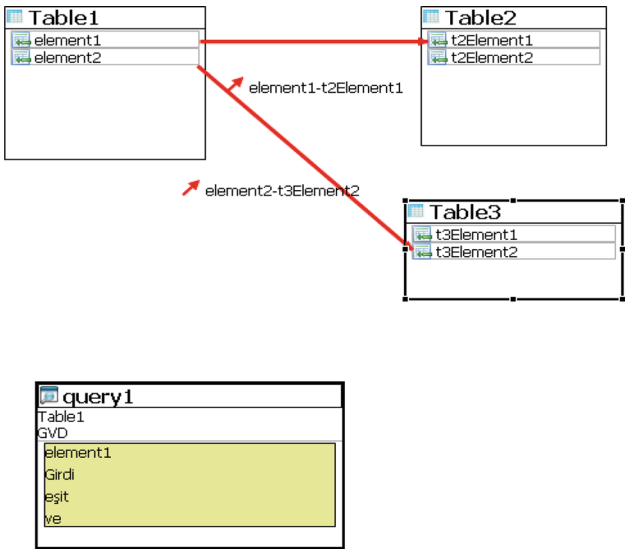

Fig. 3 An example view of the model developed using the DSL in case study 1

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 Fig. 3. 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.

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 This case aims to validate our framework as a generalizable DSL success assessment framework.

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 such as 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

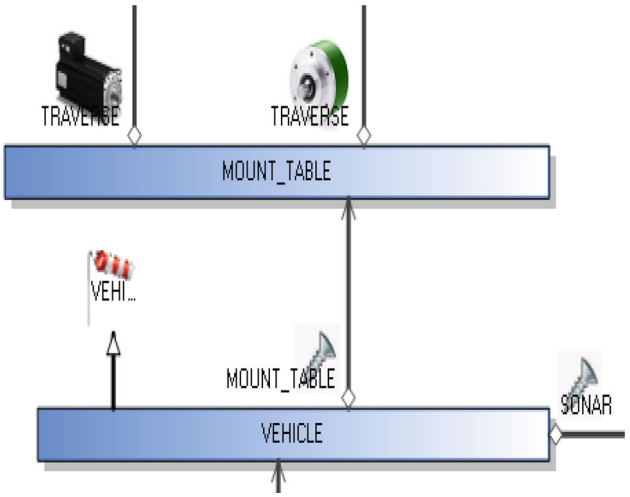

Fig. 4 An example view of the model developed using the DSL in case study 2

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 Fig. 4. In this figure, platforms are expressed using Mount_Table and Vehicle, and relations between these platforms are shown with the connectors between them.

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.

4.3 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 were 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 organization—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 2 h. 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" were submitted to the stakeholders. The material of our framework, quality characteristics importance ranking questionnaire ("Appendix I-Form I"), success assessment questionnaire ("Appendix I-Form II") and template to store domain experts' opinion on the FQAD (Sect. 5.1) were 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 the 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 Sect. 5.

Case study 1 All quality characteristics were described in a Word document and sent out via e-mail to the five stakeholders.

Case study 2 The results of the first case study were carefully analyzed, and the critiques received were reflected to the 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 Sect. 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 three stakeholders.

4.4 Evaluation of validity

Four important aspects of the quality of an empirical work are recognized as construct validity, internal validity, external validity, reliability [27]. 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 could improve 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 the 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 [26]. 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 were done and documented systematically so that replicability has been ensured.

5 Evaluation of FQAD

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

5.1 Evaluation strategy

An evaluation strategy and a standard questionnaire (called Evaluation Template) were developed for evaluating FQAD. Evaluation criteria determined in [16] were adopted to evaluate FQAD.

The Evaluation Template is used to provide a context for planning an evaluation in which the methods and procedures described in the 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 the FQAD method 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, organization, 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 Case study 1 evaluation results and discussion

Evaluation of the participants of case study 1 is summarized in Table 4.

According to the answers to the first part, the stakeholders thought that it was easy to understand what to do and not difficult at all to carry out the FQAD process due to its guidance given by the framework. 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, how-

Table 4 Case study 1 evaluation template and results

Level of validation	Evaluation criteria	Evaluation results
Basic	Complete	Some 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 organized	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

ever, be improved regardless of the method used and possibly improve confidence in the interpretation. Participants stated a need for an automatic results generation structure.

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 the FQAD method. One stakeholder thought the 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 5 discusses the improvements obtained from the first case study.

5.3 Case study 2 evaluation results and discussion

Evaluation of the participants of the second case study is summarized in Table 6.

We received responses from all three participants. The questions can be found in Table 6. 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.

The stakeholders found that the results closely reflected their opinion on what was important. But there was a strong 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 the 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

Table 5 Case study 1 improvements

Level of validation	Evaluation criteria	Evaluation results
Basic	Complete	One more sub-characteristic is added to the Expressiveness characteristic
	Understandable	Functional suitability, Reliability, Extensibility characteristics and sub-characteristics descriptions are detailed. Performance efficiency characteristic renamed as Productivity and description is changed
	Appropriate for audience	It is taken into consideration while detailing the characteristic descriptions
Use	Produced expected results	POOR effective 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

Table 6 Case study 2 evaluation template and results

Level of validation	Evaluation criteria	Evaluation results
Basic	Complete	Yes
	Understandable	Yes./But some characteristics may cause misunderstanding
	Internally consistent	Yes
	Well organized	Yes
	Appropriate for audience	Yes
	Well written (readable)	Yes
Use	Produced expected results	No, "Reusability" measurement should not be in "Maintainability" characteristic. Integrability shouldnt be in "Extensibility 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

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." The validatory case indicates the resulting DSL quality characteristics and assessment framework.

6 Conclusions

The purpose of the research was to work out a practical, consistent and systematic framework for assessing DSL using quality characteristics balanced according to an evaluator perspective. These characteristics had a direct link to the

goals and hence also addressed the concerns of specific stakeholders. An assessment framework was developed from the quality characteristics and assessment method constructed in the literature review and then validated in two case studies. The outcomes reported in Sect. 5 provide a clear summary of capability for the assessment framework.

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

6.1 Contributions

The most important contribution of this research has been the presentation of a comprehensive framework for DSL success assessment. The research has pointed out the need for a comprehensive framework for DSL assessment, and quality characteristics using contextual information to align them up to the highest level of goals. Thus, we have attempted to unify and bring together the experiences from the fields 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 framework for qualitative assessment of DSLs and in the introduction of the assessment in FQAD (see Sect. 3). 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 quality field.

For the managers, domain-specific language developers and decision makers who often face domain-specific language success assessment, a number of practical contributions are presented. In general, they would benefit from the research deliverables through a deep understanding of the quality characteristics related to domain-specific language.

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

6.2 Limitations

The interpretive research philosophy has its own drawbacks. All the weaknesses of the used research philosophy were known in advance, and experience and recommendations from previous interpretive studies [16,26,27] were taken into consideration to overcome them. In addition, the research design (see Sect. 4) was carefully elaborated.

The interviews and the findings show that due to the nature of 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 a framework that captures different viewpoints.

Although the cases—the organizations and the subjects 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 considered to be sufficient for generalizing the findings further.

6.3 Future Work

Application of the proposed assessment framework on other domain-specific languages would be an obvious first step toward accumulating experience with and evaluating FQAD.

In FQAD, DSL quality characteristics are determined by examining studies on both graphical and textual DSLs. Since in the case studies the used DSLs are graphical languages, we do not have enough evidence that our framework applies to textual languages too. But we believe that our framework is also applicable to textual languages and further case studies would be performed to validate the applicability of the framework on textual languages.

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. For example, qualitative assessment could be combined with some quantitative steps such as assessing the DSLs to evaluate completeness.

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 toward strengthening the proposed assessment framework.

Acknowledgments We would like to thank especially the members of the ASELSAN Company REHIS and SST Group software development departments, where we conducted the case studies. The authors are deeply grateful for the thorough evaluation and constructive criticism provided by the anonymous reviewers whose suggestions have been crucial in improving the quality of this paper.

7 Appendix: Assessment levels in FQAD

7.1 DSL quality sub-characteristics support levels

Sub-characteristics support levels apply to a DSLs 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 [16] and depicted in Table 7 are designated in an ordinal scale in which "full support" corresponds to the highest level. According to Kitchenham et al. [16], granularity of the support levels depends upon the feature that is assessed and requirements. Based on this, we defined four support levels for DSLs.

7.2 DSL success levels

Success levels apply to a DSLs quality achievement. These levels are a means of assessing a DSL. The three success levels are presented in Table 8.

Table 7 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

Table 8 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

8 Form I

See Table 9.

Table 9 Characteristics importance ranking form

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 is easy to maintain.	**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.	Extensibility	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 is amenable to integration with other languages	**Please make a choice**

9 Form II

See Table 10.

10 Form III

See Table 11.

Table 10 Sub-characteristic assessment statements form

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 language provides mechanisms for compactness of the representation of the program. (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**
	Extensibility	
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 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.	**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**

Table 11 Results are generated automatically in form III

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
Extensibility		
	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**

References

1. Amstel, M.F., Lange, C.F., Brand, M.G.: Using Metrics for Assessing the Quality of ASF+SDF Model Transformations, ICMT. Springer, Berlin (2009)
2. Bunge, M.A., Ontology, I.: The Furniture of the World (vol. 3). Reidel, Dordrecht (1977)
3. Cameron, K.S.: Critical questions in assessing organizational effectiveness. Organ. Dyn. **9**(2), 66–80 (1980)
4. 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)
5. Frank, U.: Domain-specific modeling languages-requirements analysis and design guidelines. In: Reinhartz-Berger, I., Sturm, A., Clark, T., Wand, Y., Cohen, S., Bettin, J. (eds.) Domain Engineering: Product Lines, Conceptual Models, and Languages. Springer, Berlin (2013)
6. 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)
7. Haugen, O., Mohagheghi, P.: A multi-dimensional framework for characterizing domain specific languages. In: Proceeding of the 7th OOPSLA Workshop on Domain Specific Modeling (2007)
8. Hermans, F., Pinzger, M., Deursen, A.V.: Domain specific langages in practice: a user study on the success factors. In: MODELS09, pp. 423–437. Springer, Berlin (2009)
9. 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)
10. Kahlaoui, A., Abran, A., Lefebvre, E.: DSML success factors and their assessment criteria. Metrics News **13**(1), 43–51 (2008)
11. Kärna, J., Tolvanen, J.P., Kelly, S.: Evaluating the use of domain-specific modeling in practice. In: Proceedings of DSM09 (2009)
12. Karsai, G., Krahn, H., Pinkernell, C., Rumpe, B., Schindler, M., Volkel, S.: Design guidelines for domain specific languages. In: Proceedings of the 9th OOPSLA Workshop on Domain-Specific Modeling (DSM' 09), Orlando, FL, USA (October 2009)
13. Kelly, S., Tolvanen, J.P.: Domain Specific Modeling Enabling Full Code Generation. Wiley, New York (2008)
14. Kelly, S., Pohjonen, R.: Worst practices for domain-specific modeling. IEEE Softw. **26**(4), 22–29, (July/August 2009)
15. Khedker, U.P.: What Makes a Good Programming Language? Technical Report TR-97-upk-1, Department of Computer Science University of Pune (1997)
16. Kitchenham, B.A., Linkman, S., Law, D.: DESMET: a methodology for evaluating software engineering methods and tools. Comput. Control Eng. J. **8**(3), 120–126 (1997)
17. 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 ECOOP06, Nantes, France (2006)
18. 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 **7**(2), 247–264 (Special Issue) (2010)
19. Kouhen, A.E., Dumoulin, C., Gerard, S., Boulet, P.: Evaluation of Modeling Tools Adaptation, hal-00706701, version 2 (2012)
20. Krogstie, J.: Evaluating UML Using a Generic Quality Framework Chapter in UML and the Unified Process. Idea Group Publishing, Hershey (2003)
21. Merilinna, J., Pärssinen, J.: Comparison between different abstraction level programming: experiment definition and initial results. In: OOPSLA Workshop on Domain-Specific Modeling, Montreal, Canada (2007)
22. McKean, D., Sprinkle, J.: Heterogeneous multi-core systems: UML profiles vs. DSM Approaches. In: Proceedings of the 2012 Workshop on Domain Specific Modeling (2012)
23. Mohagheghi, P., Dehlen, V., Neple, T.: Definitions and approaches to model quality in model based software development a review of literature. Inf. Softw. Technol. **51**, 1646–1669 (2009)
24. Moody, D.L.: The physics of notations: toward a scientific basis for constructing visual notations in software engineering. IEEE Trans. Softw. Eng. **35**, 756–779 (2009)
25. Paige, R., Ostroff, J., Brooke, P.: Principles for Modeling Language Design, Techical Report CS-1999-08, York University (1999)
26. Pfleeger, S.L., Kitchenham, B.A.: Principles of survey research. ACM SIGSOFT Softw. Eng. Notes **26**, 16–18 (2001)
27. Runeson, P., Host, M.: Guidelines for conducting and reporting case study research in software engineering. Empir. Softw. Eng. **14–2**, 131–164 (2009)
28. Software Engineering Institute, CMMI for Acquisition, Version 1.3, Technical Report, CMU/SEI-2010-TR-032, Carnegie Mellon (November 2012)
29. Strembeck, M., Zdun, U.: An approach for the systematic development of domain specific languages. Softw. Pract. Exp. **39**, 1253–1292 (2009); John Wiley & Sons Ltd
30. Wand, Y., Weber, R.: On the ontological expressiveness of information systems analysis and design grammars. J. Inf. Syst. **3**, 217–237 (1993)
31. Wand, Y., Weber, R.: On the deep structure of information systems. Inf. Syst. J. **5**, 203–223 (1995)
32. 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)

Author Biographies

Gökhan Kahraman holds Electrical and Electronics Engineering M.Sc. degree from Hacettepe University and PhD. degree from Middle East Technical University (METU), Ankara, Turkey. He is currently working as a senior expert software engineer at ASELSAN A.Ş. in Turkey. His current research interests are Domain Specific Languages, Code Generation and Model Driven Software Development.

Semih Bilgen received the B.S. degree from the Department of Electrical and Electronics Engineering, Middle East Technical University (METU), Ankara, Turkey, in 1973, the M.Sc. degree from Rensselaer Polytechnic Institute, Troy, NY, in 1976, and the Ph.D. degree in electrical and electronics engineering from the University of Manitoba, Winnipeg, MB, Canada, in 1982. He is currently a professor of electrical and electronics engineering with METU. His current research interests include information systems, software engineering, and computer networking.