Article can be accessed online at http://www.publishingindia.com

Quality Evaluation of Business Object Based Requirement Engineering Framework

Shreya Banerjee*, Anirban Sarkar, Narayan C Debnath*****

Abstract

Design and development of efficient software product depends on the quality of the requirements engineering phase. Hence it is required to devising a general purpose quality evaluation framework for that phase. In this paper, a theoretical approach of quality evaluation framework has been proposed for generic requirement engineering process. For this reason a set of quality metrics have been proposed along with their illustrations specifically based on Business Object based Requirement Engineering framework [12]. However, the proposed metrics are applicable to any other frameworks also. Finally these proposed metrics are validated theoretically using the framework defined by Briand et al.

Keywords: Object Oriented Requirement Engineering, Quality Analysis, Quality Evaluation Framework, Quality Metrics, Requirement Object, Theoritical Validation

1. Introduction

Software systems requirements engineering (RE) process encompasses the set of tasks that lead to an understanding of what the business impact of the software will be, what the customer wants, and how end-users will interact with the software. In general, Requirements Engineering is a structured process of eliciting, defining, negotiating, prioritizing and validating requirements of a system. The growing popularity of Object Oriented paradigm in the development of complex and large scale information system has led the influence of Object Oriented Requirements Engineering towards elicits and analyzes the requirements of such system. Object oriented requirements engineering is an approach to encapsulating information about the process and product, as well as functionality into a requirements object [2].

For complex and large scale information system development process, the stakeholder's requirements are often changed. Moreover, business processes of large system are dynamic in nature. Thus it is very difficult to anticipate any future requirements for such system in the scenario of evolving business processes over time. The use of object oriented paradigm to define and analyze the requirements is added advantageous in such cases. A semantic representation of software requirements is required to remove the communication gaps exist between stakeholders, software engineers and project managers [11]. The requirements objects can be represented using a common set of semantic notations which are understandable both to the stakeholders and domain engineers. At the same time, to accommodate the requirements changes, the representation of the requirements objects must be reusable. In object oriented paradigm requirements are directly represented as first-class objects and the notation of a "requirements object" is used to represent the problem domain object [4]. A first-class object has the supports of all the features of object oriented paradigm and does not exhibit any difference between objects in the domain model and requirements objects in general.

Several requirements engineering frameworks have been proposed in recent literatures to model the requirements objects. In [13] these requirement engineering frameworks are evaluated based on some common features. Quality

* Department of Computer Applications, National Institute of Technology, Durgapur, India. E-mail: shreya.banerjee85@gmail.com
** Department of Computer Applications, National Institute of Technology, Durgapur, India. E-mail: sarkar.anirban@gmail.com
*** Department of Computer Science, Winona State University, MN, USA. E-mail: ndebnath@winona.edu

Table 1: Comparison of OO Requirements Engineering Frameworks for Quality Features [13]

Model Name / Reference	*Quality Evaluation Level Features*												
	(a)	*(b)*	*I*	*(d)*	*(e)*	*(f)*	*(g)*	*(h)*	*(i)*	*(j)*	*(k)*	*(l)*	*(m)*
[5,6]	Y	-	-	-	-	-	-	-	-	-	-	-	-
MORE[8]	Y	-	P	P	P	-	-	-	-	-	-	-	-
GORE[1]	Y	Y	P	P	-	-	-	-	-	-	-	-	-
AGORA[9]	Y	Y	Y	Y	Y	P	P	-	-	-	-	-	-
BORA[7]	Y	P	P	P	P	-	-	-	-	-	-	-	-

Y: supported in the model, Hyphen: not supported or not explained how to support it, P: partially supported
Business Object Based Requirement Engineering Framework: The Basic

Evaluation Level Features are one of them. Traditionally, quality in requirement frameworks is a complex mix of factors that will vary across different applications and the customers who request requirements. However, a good requirements engineering framework must support the validation of certain quality features to evaluate the requirements objects and related models achieved from the framework.

Very few research proposals of requirements engineering frameworks are exist in the literatures which are included with quality evaluation schemes (Table I). Further, those schemes are specific to the related frameworks and also not very comprehensive. In [13], a set of generic quality factors described like, a) Ambiguity, b) Correctness, c) Completeness, d) Consistency, e) Traceability, f) Maintainability, g) Importance and stability, h) Reliability, i) Efficiency, j) Productivity k) Visibility, l) Usability, m) Interoperability. The proposed framework in [5, 6] support only the factor of ambiguity. MORE [8] supports ambiguity but partially support the quality factors like complexity, consistency and traceability. On the other hand, AGORA [9] support the quality factor of number (a), (b), (c), (d), (e) but partially support the factor of number (f) and (g). GORE[1] and BORA[7] also support ambiguity but they partially support some other qualities. Majority of the proposed frameworks in the literatures do not provide any quality evaluation scheme to assess the quality of the requirements objects and related analysis models. Few of the quality features have been defined and discussed for the framework called AGORA in [9]. However those definitions are not generic and specific for the framework itself. Also few of the quality factors can be examined at the implementation level using CASE tools for MORE [8] and BORA [7].

In this paper in section II the basic of Business Object based Requirement Engineering framework proposed in [12] has been described. In section III a theoretical framework has been proposed for the quality evaluation for the requirements engineering process with the objective to assess the qualities of the requirements objects and related analysis models. For the purpose, separate sets of construct level, semantic level and framework level quality metrics have been proposed specifically based on Business Object based Requirement Engineering framework [12]. The proposed metrics are illustrated with the case study in section IV. The proposed set of metrics can be applicable to any general purpose requirements engineering framework. In section V the proposed metrics are theoretically validated using Briand et al.

In [12], a process driven requirements analysis framework based on Common Business Objects [3] has been proposed for large scale information system. A business object (BO) captures information about a real world (business) concept, operations, constraints, and relationships between those concepts. The advantage of using this concept is that, the set of Bos can be reusable in the context of business domain and can easily be transformed into system level objects for software realization of the specific business concept. With these perspectives, the framework consists of three phases, namely, (i) Early Requirements Analysis Phase (ii) Detailed Requirements Analysis Phase and (iii) Mapping phase. The former allows for modeling and analyzing the contextual setting of the business domain, in which the system will operate. In this step the involved Entity Bos, Actor Bos, Process Bos Collaborations and Interactions, those are relevant to the functional requirements of targeted information system domain, will be identified. In later phase, the early requirements specifications are refined with the structural, functional and nonfunctional features of the domain that is relevant to the stakeholders and their roles related to the intended information system. The refinement process is largely influenced by the concepts of Feature Oriented Domain Analysis (FODA) [10].Both the phase will take the stakeholder's roles, requirements and objectives towards the domain of interest as prime inputs. Identifications, semantic representations and refinements of related Bos and their inter-relationships will be done on the basis of such inputs. Both phases may be iterative in nature. The mapping phase used to map the requirement specifications to conceptual level design model and can starts just after the early requirements analysis phase.

The proposed framework is based on concept of Common Business Objects. The set of related Bos describe the set of business processes relevant to the business itself along with the intentions that these processes are supposed to fulfill and also express an abstract view of the business's "real world". The business object is capable of performing to fulfill its purpose including the collaboration with other Bos.

In the framework several requirements modeling elements like Process BO, Entity BO, Actor BO, Event BO, Interface, Interaction Diagram, Collaboration Diagram, Interaction Collaboration Network, Feature Tree etc. have been described to express different business concepts of the domain, relevant to targeted information system and in the real business scenario. This framework is supported with object oriented features like abstraction, reusability and inheritance.

Early Requirement Analysis Phase consist of six steps- (1) Identification of Process Bos, (2) Identification of Entity Bos and separation of Actor Bos, (3) Identification of Interactions, (4) Identification of collaborations, (5) Identification of Entity Bos associated with interactions and (6) Identification of Event Bos.

In first step Process Bos related to the domain are identified with the activities and context. Entity Bos are identified in second step and Actor Bos which are users or stakeholders of the system are separated from Identified Entity Bos. In third step interactions between Process Bos and Actor Bos are identified. These interactions implement the activities of Process Bos. Fourth step identifies collaborations between Actor Bos which in turn identifies roles of participant Actor Bos concerned to the specific Process BO. This develop collaboration diagram in the context of some specific Process Bos. The fifth step is necessary to draw the association between the identified Entity Bos including Actor Bos for the design phase of the candidate domain [12]. Finally in sixth step Event Bos are generated as a result of specific set of interactions between related Entity Bos [12].

Detailed Requirement Analysis phase consists of three steps. First step is refinement of Process Bos and interactions. In this step Process Bos are refined in two levels. In first level Process Bos and related activities are refined in possible collection of related sub Process Bos. This task may be iterative. In second level each refined process Bos are further refined by adding domain level features like structural, functional and nonfunctional features along with attributes. Structural features describe the object label features of Process Bos. Functional features describe the operations or functionalities of Process Bos. Non functional features represent the expected quality standard requirements of stakeholders for Process Bos and this feature may be optional. Feature Tree Diagram represents the features of Process Bos and logical grouping like AND, OR and EXOR of the features to satisfy the stakeholder's requirement [12].

Second step is refinement of Entity Bos. This step is also consisting of two levels. In first level, known as vertical refinement, different possible hierarchical levels are identified between or within Entity Bos using inheritance and containment relationships. In second level, known as horizontal refinement, each refined Entity Bos are further refined by adding domain level features as well as in case of Process Bos. This generates the feature tree diagram for each refine Entity Bos. Here feature tree represents the constraint requirements for the specific domain.

Third step is identification of user interfaces for intended information system. An user interface is represented using interface template. The Interface Template will contain the information of interface name and ID, target Process BO, Actor BO, roles, related activities, related other Entity Bos, related Event Bos and related constraints.

In Mapping Phase the domain level requirements specifications achieved from the early requirements analysis phase and consequently from the detailed requirements analysis phase are mapped into conceptual level design model. This phase consist of two steps- Early Mapping and Detailed Mapping. In early mapping Process Bos, Actor Bos, Event Bos, Interactions etc identified in

Table 2: Formal Notations of the Framework

Framework Component	*Description*
BO=PBO ∪ EBO ∪ EVBO	The set of business objects
R=BO×BO	The relationships between the concerned Bos.
IS={BO,R}	Any information system
EBO_i = {id, D}	Any Entity BO within IS where, id refers to the identity of the entity BO and D is vector of attributes to characterize the Entity BO.
PBO={id,T}	An Process BO where, id refers to the identity of the process BO and T is vector of activities that may be performed by the process BO.
$EvBO_i$ = {id, IR_j, C}	An Event BO can be resulted from one interaction and can be represented as where id refers to the identity of the event BO and $IR_j$is the specific interaction relationship due to which the event BO resulted, *C*is the constraint specification for occurrence of event BO and *C* ≠ □.
CR_i = {{ABO_i.id, ABO_i.Rp}, {ABO_j.id, ABO_j.Rq}, C}	A collaboration can be defined within any two Actor Bos where *ABO_i, ABO_j*□*Is*and *C*is constraint on that relationship.
IR_i = {{ABO_i.id, PBO_j.id}, ABO_i.R_q, G, C} or {{PBO_j.id, ABO_i.id}, ABO_i.R_q, G,C}	An interaction relationship can be defined within an Actor BO and a Process BO based on some role of ActorBO where *ABO_i, PBO_j*□*IS, G* is the objective of goal that can be achieved on performing that activity and *C*is constraint on that relationship.[3]

Early Requirement Analysis Phase can be mapped into UML class diagrams to generate the early prototype of the system. In detailed mapping steps, the high level design schema achieved from the early mapping can be further refined to deliver the full-fledged conceptual design of targeted information system for the domain of interest [12].

Table 3: Graphical Notations for BO based Requirement Analysis Framework

Taxonomy	*Graphical Notation*
Entity BO	▭
Process BO	▢
Event BO	▭
Attribute	⬡
Actor BO	A
Encapsulation	—
Inheritance	➞
Collaboration	- - - - - -
Interaction	⟶

The formal and graphical notations of the framework have been summarized in Table II and Table III respectively. In this context, formally, a Business Information System *IS* will be represented as a pair *{BO, R}*, where *BO* represents the set of business objects concerned to the targeted system *IS.* The set of Bos can be expressed as and *BO=PBO È EBO ÈEvBO,* where *PBO* is the set of concerned Process Bos, *EBO* is the set of concerned Entity Bos including Actor Bos and *EvBO* is the set of concerned Event Bos. The *R* is the set of binary relations defined on Bos as *R Î BO ´ BO* representing the relationships between the concerned Bos.

2. Quality Evaluation Framework

Efficient software design and subsequently efficient software product development depends on the quality of the requirements engineering process. It is evident from the previous researches that effective requirements engineering process can reduce the risk of failure for development of large scale information system. To establish Business Object Requirement Engineering Framework as a good requirement engineering framework we have to require defining some basic quality metrics. Hence we define the basic quality metrics in two groups namely construct level quality metrics and semantic level quality metrics. Although these quality metrics are defined based on Business Object based Requirement Engineering Framework, it is also implied to any other framework. We also define some quality factors with the help of these basic quality metrics for the purpose of quality evaluation. Those quality factors described in [13] are as follows,

(a) Ambiguity: It is a quality factor used to validate each requirements object in the specification that those are atomic and cannot have more than one interpretation.

(b) Correctness: It is a quality factor which means how many requirements in requirement specification meet customer's need.

(c) Completeness: It is a quality factor which means necessary requirements objects are not lacking specification.

(d) Traceability: It is a quality factor used to specify the implementation of the requirements objects towards the related design components and vice versa.

(e) Consistency: It is a quality factor used to check the presence of the inconsistency among requirements objects in the specification.

(f) Maintainability: It is a quality factor which means the effort required to locate and fix an error in requirement specification.

(g) Importance and stability: This quality factor means how clearly the prioritization and stabilization of the requirements objects are described when they are necessary to be specified.

(h) Reliability: It is a quality factor which means the extent to which a software application can be expected to perform its intended functionalities as stated in requirement specification.

(i) Efficiency: This specifies the degree to which the requirement object makes optimal use of limited business resources.

(j) Productivity: It is the ratio between the functional value of software produced to the labor and expense of producing it.

(k) Visibility: It is the quality factor which is used to maintain end user access policies for the application.

(l) Interoperability: This property means how efficiently coupling can be established from one system to another. The requirements related to interlinking of two systems are complex in nature.

3. Proposed Set of Metrics

We classify the quality metrics based on Business Object based Requirement Engineering Framework in two categories. One is construct level and another is semantic level quality metrics.

a. Construct Level Quality Metrics

Business Object based Requirement Engineering Framework has been proposed using a well defined set of business object constructs. The set of construct level quality metrics are proposed below with view point and unit.

1. *Metric 1*: Number of process BO in a system (NPBO).

Description: Let consider an information system IS with Process BOs $PBO_{1,}$ PBO_2 .....,PBO_n those are specified in the system. Then,

$$NPBO = \sum_{i=1}^{n} PBO_i$$

View Point: The number of process BOs are used to identify how many requirement objects and how many interaction relationships are specified within the system. The larger number of Process BOs in a system implies the greater expressiveness of the system and lowers the abstraction of the system.

2. *Metric 2*: Number of activities or tasks may be performed by all Process BO in a system (NT).

Description: Let consider a Process BO PBO_i then the activities or tasks performed by the PBO $_i$are t_1, t_2,... t_n. Therefore:

$$NT_i = \sum_{k=1}^{n} t_k$$

Now,

$$NT = \sum_{i=1}^{n} NT_i$$

The unit is number.

View Point: The larger the number of tasks performed by one PBO, the system is more abstract and less complex.

3. *Metric 3*: Number of Entity BOs in a system (NEBO).

Description: Let consider an IS with Entity BOs EBO_1, EBO_2, ...,EBO_n those are specified in the system. Then

$$NEBO = \sum_{i=1}^{n} EBO_i$$

*View Point:*1. *NEBO* is used to identify the collaboration relationships in the IS.2. *NEBO* IS also used to identify the interaction relationships in the IS.3. *NEBO* also represents one of static structure of the system.

4. *Metric 4*: Number of attributes of all Entity BOs (ND).

Description: Let consider an Entity BO (*EBO*).Suppose the attributes of the Entity BO is d_1, $d_{2,}$..., d_n. Therefore total number of attributes of an Entity BO (EBO_i) is

$$ND = \sum_{i=1}^{n} d_k$$

Now,

$$ND = \sum_{i=1}^{n} ND_i$$

The unit is number.

View Point: The larger the number of attributes of an EBO, the EBO is more expressive.

5. *Metric 5*: Number of Event BOs in a system (NEVBO).

Description: Let consider an IS with event BOs $EVBO_1$, $EVBO_{2,}$……, $EVBO_n$ those are specified in the system. Therefore,

$$NEVBO = \sum_{i=1}^{n} EVBO_I$$

View Point: NEVBO are required to express the complexity and static structure of the system.

6. *Metric 6*: Number of Actor BOs in a system (NABO).

Description: Let consider an IS with Actor BOs ABO_1, ABO_2,…….,ABO_n, those are specified in the system. Therefore:

$$NEVBO = \sum_{i=1}^{n} ABO_i$$

View Point: NABO are used to identify roles of Actor BO and the interaction relationships between a PBO and an ABO.

7. *Metric 7*: Number of requirement objects in a system (NRO).

Description: Let consider an IS with numbers of *PBO, EVBO* and *EBO*. Therefore

$$\text{NRO} = \text{NPBO} + \text{NEBO} + \text{NEVBO}$$

The unit is number of objects.

View Point: *NRO* is used to identify how many BOs (Business Objects) are present in the system. The larger the NRO implies the static complexity is larger and abstraction is lower.

b. Semantic Level Quality Metrics.

In Business Object based Requirement Engineering Framework semantics are expressed using the set of well defined set of business object constructs. These semantics are also very important for defining quality metrics. The set of semantic level quality metrics are proposed below with view point and unit.

8. *Metric 8*: Total Number of interaction relations in a IS (NIR).

Description: Let consider an IS with a PBO *i* and an ABO *j* and IR_1, IR_2,…….,IR_n are the interaction relations between PBO_i and ABO_j. Therefore

$$NIR_{ij} = \sum_{k=1}^{n} IR_k$$

And,

$$NIR = \sum_{i=1}^{n}\sum_{i=1} NIR_{ij}$$

The unit is number.

View Point: This metric is used to identify the roles of Actor BOs when they interact with a PBO. This also helps to figure out collaborations among actor BOs.

9. *Metric 9*: Number of roles performed by all Actor BO within the system (NR).

Description: Let consider an IS with Actor BOs and roles performed by one Actor BO (ABO_i) are $r_{1,}$ $r2$,.....,r_n. Therefore number of roles performed by one *ABO* (ABO_i) is

$$NR_i = \sum_{k=1}^{n} NR_i$$

And

$$NR = \sum_{i=1}^{n} NR_i$$

The unit is number.

View Point: Roles are used to identify the collaborations among the Actor BOs.

10. *Metric 10*: Total Number of collaborations within an IS (NCR).

Description: Let consider an IS with two Actor BOs *i* and *j* and CR_1, $CR2,\ldots,CR_n$ are the collaboration relations among two Actor BOs ABO_i and ABO_j . Therefore the numbers of collaborations among two Actor BOs ABO_i and ABO_j are

$$NCR_{ij} = \sum_{k=1}^{n} CR_k$$

Now,

$$NCR = \sum_{i=1}^{n}\sum_{i=1} NCR_{ij}$$

The unit is number.

View Point: The larger the value of NCR_{ij}, the dynamic complexity value will be larger.

11. *Metric 11*: Number of total inheritance relationships in a system (*NIH)*.

Description: Let consider an IS with numbers of inheritance relationships IH_1, $IH2\ldots\ldots,IH_n$ between *ABOs* of the system.

Therefore

$$NIH = \sum_{i=1}^{n} IH_i$$

The unit is number.

View Point: From this metric we can figure out reusability of the BOs within the system.

12. *Metric 12*: Static complexity of a system (*SC*).

Description: Let consider an IS with Event BO (*EVBO*), Entity BO (*EBO*) and Process BO (*PBO*). Therefore

$$SC = NPBO + NEBO + NEVBO$$

View Point: It helps to figure out total complexity of a system.

13. *Metric 13*: Dynamic complexity of a system (DC).

Description: Let consider an *IS* with interaction relations (*NIR*) and collaboration relations (*NCR*).

Now, the Dynamic complexity of the system is,

$$DC = NIR + NCR$$

The unit is number.

View Point: It helps to measure the total complexity of the system.

14. *Metric 14*: Complexity of a system (CM).

Description: Let consider an IS with static complexity *SC* and dynamic complexity *DC*. Therefore total complexity of the system:

$$CM = DC + SC$$

The unit is number.

View Point: The larger the complexity value, abstraction of the system is lower.

c. Framework level Quality Factors

Some quality issues described in [13] has been formally described in this section with the help of some basic quality metrics.

15. *Metric 15*: Ambiguity of a system (AMB).

Description: Let consider *URF* as a function which gives user requirement as output. Suppose the list of user requirements are $UR_1, UR_2,\ldots\ldots,UR_n$. Let also consider an IS with *PBO, ABO, EVBO, EBO* and *NUM* are the summation of requirements which are not same. Therefore

$$NUM = \left(\sum_{i=1}^{n} PBO_i + \sum_{i=1}^{n} EBO_i + \sum_{i=1}^{n} EVBO_i\right)$$

Where, $PBO_i.T \neq PBO_j.T$, $EBO_i.D \neq EBO_j.D$, $EVBO_i.D \neq EVBO_j.D$and total number of user requirements can be represented as,

$$NUM = \sum_{i=1}^{n} UR_i$$

Now,

$$AMB = NUM/NUR$$

View Point: Maximum value of ambiguity can be 1.The larger the ambiguity value the system is more stable. This metric can help us to evaluate quality of a system.

16. *Metric 16*: Completeness of a system (COM).

Description: Let consider an IS with *PBO, EBO, EVBO* and *URF()* returns user requirements. Therefore total number of specified user requirements in the system

$$NSUR = NPBO + NEBO + NEVBO.$$

Therefore the completeness of the system is,

$$COM = NSUR/NUR$$

View Point: Maximum value of *COM* can be 1. This metric can help us to evaluate quality of a system.

17. *Metric 17*: Correctness of a system (COR).

Description: Let consider an IS with *PBO, EBO, EVBO*. Now one BO is correctly specified if it resembles with one user requirement. Consider numbers of correct BOs are *NCB*. Therefore,

$$NCB = \left(\sum_{i=1}^{n} PBO_i + \sum_{i=1}^{n} EBO_i + \sum_{i=1}^{n} EVBO_i \right)$$

Where, $PBO_i = UR_i$ for only one value of *i*, $EBO_i = UR_i$ for only one value of *i* and $EVBO_i = UR_i$ for only one value of *i*.

Now, correctness of the system can be

$$COR = NCB/NUR$$

View Points: Maximum value of correctness of a system can be 1.This metric can help us to evaluate quality of a system.

18. *Metric 18*: Traceability of a system (TRS).

Description: Let consider an IS with *PBO, EBO, EVBO*. And *DEO( )* is a function which returns design objects $DR_1, DR_2, \ldots, DR_n$ traceable from BOs. Therefore the total number of design objects of the IS can be:

$$NDEO = \sum_{i=1}^{n} DR_i$$

Therefore traceability of the system can be:

$$TRS = NDEO/NRO$$

View Point: Maximum value can be 1. This metric can be used to measure mapping capability of the system from the requirement specification to the conceptual level design.

19. *Metric 19*: Consistency of a system (CON).

Description: Let consider an IS with PBO, EBO, EVBO and NCO are the numbers of consistency objects within the system.

Therefore:

Where PBO_i, EBO_i, and $EVBO_i$ are in the system IS unchanged due to its life time.

Therefore *CON= NCO/NRO*.

$$NCO = \sum_{i=1}^{n} (PBO_i + EBO_i + EVBO_i)$$

View Point: The maximum value of *CON* is 1. If the system is more consistent, then the system is more acceptable.

20. *Metric 20*: Maintainability of a system (MTN).

Description: Let consider an IS with number of requirement objects *RO* and their relationships *R* (Such as interaction, collaboration, inheritance, encapsulation). Let consider *NR* the total number of relations in the system. Let also consider *NARO* are the requirement objects and *NAR* are the total number of relationships between them within the system which are added to the system. Similarly *NDRO* and *NDR* are the requirement objects and relationships between the objects respectively which are deleted from the system during maintenance.

Therefore:

$$MNT = \left(\frac{(NRO + NARO - NDRO)}{NRO} + \frac{(NR + NAR - NDR)}{NR} \right)$$

View Point: Maintainability is used for fixing errors. So for fixing error number of ROs and relations can be added or deleted.

21. *Metric 21*: Productivity of a system (PRD).

Description: Let consider an IS and *SOF ( )* is a function which returns the result of system output. Suppose *SOF ( )* gives output $SO_1, SO_2, \ldots\ldots, SO_n$.

$$NSO = \sum_{i=1}^{n} SO_i$$

Therefore total number of system output can be represented as

Similarly *NUR* gives total number of user requirements.

Therefore productivity of the system is:

$$PRD = NSO/NUR$$

View Point: The total number of user requirements can be used as indicator of total labor and expense for implementing each user requirement. The increase value of PRD indicates that the system is productive.

22. *Metric 22*: Reliability of a system (RLB).

Description: Let consider a system IS with numbers of requirements objects (*RO)* and their relationships. Let consider *NUR* is the total number of user requirements in the system.

Therefore Reliability can be expressed as

View Point: RLB can be used as indicator about the functionality of the framework. Maximum value of RLB can be 1.

23. *Metric 23*: Interoperability of a system (IOP).

Description: Let consider an *IS*is consist of different modules *M1, M2,……,Mn*. Let consider the number of *RO*s in M1 is $NRO_{M1,}$ number of *RO*s in *M2* is $NRO_{M2}$and so on. Now interoperability of two modules *M1* and *M2* depends on the inter module relationships between them i.e. outgoing relationships from every module to others modules in the system.

$$IOP = \sum_{i=1}^{n} OGR_i$$

Let consider the number of outgoing relations from module *M1* is OGR_1, from module *M2* is OGR_2 and so on.

Therefore, Interoperability of the system is

Unit is number.

View Point: If in a system the number of outgoing relations and the number of ingoing relations of every module are same, then the system is composed of disjoint modules and its interoperability is high.

24. *Metric 24*: Visibility of a System (VIB).

Description: Let consider S1, S2, …Snare the end users who can access the system at some moment according to the stakeholder. Let consider NS are the total number of end users who can access the system and who has the access permissions to the system at any moment. Let also consider NTS are the total number of end users who access the system at a moment and NPS are the total number of end users who have no access permission for the system at any moment.

Therefore at any moment visibility of the system can be expressed as

$$VIB = (NTS - NPS)/NS$$

View Point: Visibility can be used to measure the security of the system. The value of visibility up to 1 is good, but if the value is greater than 1 then the system is less secure.

25. *Metric 25*: Efficiency of a system (EFC).

Description: Let consider in a system *IS* the business resources available are BR_n and the business resources used by requirement objects (*BO)* and their relations(*R)* are BR_U.

$$EFC = BR_U/BR_N$$

Therefore efficiency of the system is

View Point: The value of EFF up to 1 is good. But if the value exceeds 1 then it is worse.

26. *Metric 26*: Importance and stability of a system (IMS).

Description: Let consider a system IS with the number of requirement objects. Let consider *NROS* are requirement objects which are specified with clear specifications.

$$IMS = NROS/NRO$$

Therefore importance and stability of system is

View Point: The maximum value of IMS can be 1 and then it is more stable.

4. Illustration of Metrics with Case Study

For the illustration of proposed metrics we consider the case study proposed in [12]. The business in a Retail Organization comprised of several interested business processes like, (i) Procurements – for procuring the products for sale, (ii) Sales with the activities like, handle the customer orders and to sale the products as per order, and (iii) Accounting – to handle the bills, order payment, salaries etc. Now, the business process will be mapped into the Process BOs. The candidate Process BOs and their activities can be follows:

(a) Procurements: Book Order, Process Order, Deliver Product, Received Product, Credit Amount, Received Amount.

(b) Accounts: Raise Procurement Bill, Payment Adjust, Raise Bill for Order, Payment Clearance for Order.

(c) Sales: Place Order, Handle Order, Deliver Order Product, Receipt of Ordered Product, Received Payment from Client, Clear Payment.

Table 4: Illustration Result of Quality metrics based on Business Object based case study

Metrics	*Value*	*Unit*	*REMARK*
NPBO	5	PBO	Consider only five Process BOs.
NT	38	Number	Consider activities of all PBOs.
NEBO	8	EBO	Consider five Actor BOs and three Entity BOs
ND	8 X a	Number	Consider that attributes of every Entity BOs are a.
NEVBO	3	EVBO	Consider the Event BOs for three PBOs Cars Issue, Book Return and Book Issue.
NABO	5	ABO	With respect the Domain Level Interaction Diagram.
NRO	16	BO	Consider all the PBOs, ABOs, EBOs in Figure 3 and the EVBOs in three afforementioned PBOs.
NIR	38	IR	Consider the interaction relations between all PBOs in the domain.
NR	5xa	Number	Consider that all ABOs perform *a* number of roles.
NCR	4X 3X e	CR	Consider that between every ABOs there are collaboration relations and consider that value of collaboration relations between every ABOs are e.
NIH	5	IH	Consider all the inheritance ralationship between ABOs.
SC	16	BO	According to the Domain Level Interaction Diagram in Figure 3.
DC	38+(4 x3xe)	Number	Considering the collaboration relation between every ABOs.
CM	54+(4x3xe)	Number	According to the value of SC and DC.

Table 5: Illustration Result of Quality Metrics Based on Business Object Based Case Study

Metrics	*Value*	*Unit*	*REMARK*
AMB	16/UR	Number	Consider that NUR is UR i.e. all the requirements proposed by the customer are UR.
COM	16/UR	Number	Consider that NUR is UR i.e. all the requirements proposed by the customer are UR.
COR	16/UR	Number	Consider that NUR is UR i.e. all the requirements proposed by the customer are UR.
TRS	DR/16	Number	Consider that NDEO is DR.
CON	16/16=1	Number	As all the BOs are unchanged due to its life time.
MNT	2	Number	Consider there no ROs or relations are added or deleted during maintenance.
PRD	SO/UR	Number	Consider that value of SOF is SO and value of NUR is UR.
RLB	16/UR	Number	Consider that NUR is UR i.e. all the requirements proposed by the customer are UR.
IOP	OGR	Number	Consider the diagram in figure 3 as a module and there are numbers of modules in the system and total outgoing relations from every module in the system are OGR.
VIB	(TS-NPS)/PS	Number	Consider TS is the value of NTS, NPS is the value of NPS and PS is the value of NS.
EFF	BR_U/BR_N	Number	Consider $BR_U$and BR_N as they are specified in the metric definition.
IMS	ROS/16	Number	Consider ROS is the value of NROS.

Figure 1: Domain Level Interaction Diagram

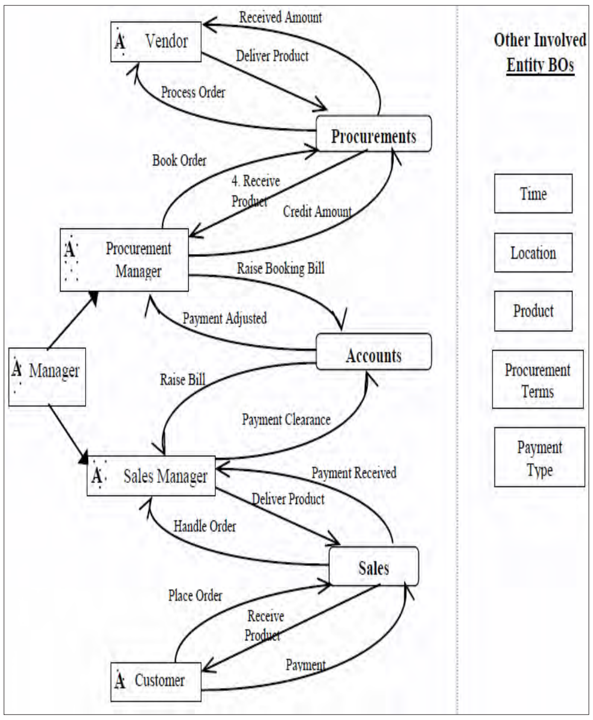
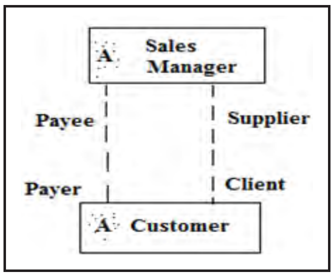


Figure 2: Collaboration Diagram



According to [12] the domain level interaction diagram of the above example has been shows in Figure 1. Here we assume the attributes of the Actor BO VENDOR are name, city, phone number, license number. Also assume in SALES PBO the possible Event BOs may be Order Status, Order Processed, and Payment Processed [12]. The collaboration relation between ABO Sales Manager and ABO Customer has been shown in Figure 2. The proposed metrics are illustrated with this case study as follows, in TABLE IV. And TABLE V.

5. Validation of Proposed Metrics Using Briand et al

In order to validate the set of metrics of object oriented requirement engineering frameworks formally the framework defined by Briand et al. [4] has been used.

In case of BO based requirement engineering framework the system IS represents the concept of the system <E, R>. *IS* is represented formally in the framework as IS={BO,R}. Therefore IS<BO,R> resembles with S<E,R> in the framework defined by Briand et al.[4] .In IS E is the set of Business Objects And R is the relations between them. Now a module M is subset of system IS where $m=<E_m,R_m>$. $E_m$consists of various business objects like PBO, ABO, EBO and EVBO and R_m represents relationships between the Business Objects like interaction, collaboration, inheritance and encapsulation. Therefore $E_m \subseteq E$, $R_m \subseteq E_m \times E_m$ and $R_m \subseteq R$ holds for the module m. Now we can say if there are two modules m1 and m2, then $E_{m1} \cap E_{m2} = \Phi$.

Now the framework is applied in order to validate the set of metrics. At first the framework is applied to validate the metric of *NPBO* i.e. number of process BO in a system. NPBO is a size measure. It follows the following size properties:

1. *Non-negativity*: The number of PBOs (One type of BO) must be always greater than or equal to zero. Without PBOs the IS has no functional express ability. As PBOs are countable objects it cannot be negative. So NPBO > = 0.
2. *Null Value*: If no BOs exist in the system IS, then NPBO=0.
3. *Modular Additivity*: Let there are two disjoint modules m1 and m2 in the system IS. Now any PBO of IS is an element of either module m1 or module m2. Now it is easy to see that NPBO(IS)=NPBO(m1)+NPBO(m2).Where m1⊆IS , m2⊆IS, NPBO(m1)⋂NPBO(m2)=Φ and NPBO(IS)=NPBO(m1)∪NPBO(m2).

So NPBO are theoretically valid. In a similar way NABO, NEBO, NEVBO, NRO, NT and ND are validated. All of these are size measures. Similarly NIR NCR, NIH and NR are also size measures. All the framework level quality factors are validated using size measure except interoperability and visibility. Interoperability is validated using coupling measure. In order to validate the interoperability (IOP), the following coupling properties must be accomplished:

1. *Non-negativity*: If a system IS consist of numbers of modules then there should be numbers of inter relations between them or not. But it cannot be negative, as number of inter relations cannot be negative. So interoperability of a system IS or a module m is [IOP (m)>=0 | IOP (IS)>=0].
2. *Null value*: If the inter relations between the modules is zero i.e. outgoing relations from the module is zero then interoperability of the system as well as of the module is zero. Therefore [IOP (m)=0 | IOP(IS)=0].
3. *Monotonicity*: Let two information system IS' =<BO, R', M'> and IS" =<BO, R",M"> be two modular systems (with the same set of elements BO) such that there exist two modules m' ∈ M', m"∈ M" such that R' - OGR(m') = R" -OGR(m"), and OGR(m') ∈ OGR(m") where OGR are the total number of outgoing relations from the module. Therefore interoperability of the module m' is less than or equal to that of module m" as value of OGR (m") is greater than or equal to OGR(m').So[IOP(m')<=IOP(m")|IOP(IS')<=IOP(IS")]. So adding inter module relationship cannot decrease the interoperability of a module as well as of a system.
4. *Merging of Modules:* Let consider IS'=<BO',R',M'> and IS" = <BO",R",M"> be two modular systems such that BO' = BO", R' =R", and M" = M' - {m'1,m'2} ∪{m"}, where $m'1 = <E_{m'1},R_{m'1}>$, $m'2 = <E_{m'2},R_{m'2}>$, and $m" = <E_{m"},R_{m"}>$, with m'1 ∈ M', m'2 ∈ M', m" ∉ M', and $E_{m"} = E_{m'1} \cup E_{m'2}$ and $R_{m"} = R_{m'1} \cup R_{m'2}$. (The two modules m'1 and m'2 are replaced by the module m", whose elements and relationships are the union of those of m'1 and m'2). As the modules m'1 and m'2 may have common inter

relationship, therefore[IOP(m'1)+IOP(m'2)>=IOP (m")| IOP(MS') >=IOP(MS")].

5. *Disjoint Module Additivity:* Let consider IS' = <BO,R,M'> and IS" = <BO,R,M"> be two modular systems (with the same underlying system <BO,R>) such that M" = M' - $\{m'_1,m'_2\} \cup \{m"\}$, with $m'_1 \in M'$, $m'_2 \in M'$, $m" \notin M'$, and $m" = m'_1 \cup m'_2$. (The two modules m'_1 and m'_2 are replaced by the module m", union of m'_1 and m'_2). If no relationships exist between the elements belonging to m'_1 and m'_2, i.e., $IGR(m'_1) \cap OGR(m'_2) = \Phi$ and $IGR(m'_2) \cap OGR(m'_1) = \Phi$, then [$IOP(m'_1) + IOP(m'_2)$ = IOP(m") | IOP(MS') = IOP(MS")]. Here IGR represents the ingoing relations from one module to another and OGR represents outgoing relations from one module to another.

So IOP is theoretically valid. Similarly visibility (VIB) is validated using length measure. Therefore for validating VIB, the following length properties must be accomplished:

1. *Non-negativity*: Visibility of a system is non- negative as at any moment the number of end users who can access the system cannot be negative. The value of VIB can be zero if the value of NPS is equal to NTS i.e. all the end users who access the system at that moment have no access permission to the system. Therefore VIB (IS)>=0.
2. *Null Value*: The visibility (VIB) of a system can be null if the numbers of BO is zero. Because if it is zero, then there are no things for accessible by the end users. Therefore VIB(S) = 0.
3. *Non-increasing Monotonicity for Connected Components*: Let IS be a system and m be a module of IS such that m is represented by a connected component of the graph representing IS. Now adding relationships between BOs of m does not increase the VIB of IS as it is predefined and depends on the value of the access permissions of the end users. Therefore if a new system IS' is generated from IS by adding relationships between the BOs of The system IS then VIB (IS)>=VIB (IS').
4. *Non-decreasing Monotonicity for Non-connected Components*: Let IS be a system and m_1 and m_2 be two modules of IS such that m_1 and m_2 are represented by two separate connected components of the graph representing IS. Adding relationships from elements of m1 to elements of m2 does not decrease the VIB of IS. Because adding the relations between the modules increase the inter relations of the modules. Therefore it increases the new relations between the modules and it can increase the end users who can access the system i.e. it can increase visibility also. Therefore if a new IS' is generated from IS by adding relations between m_1 and m_2 then VIB (IS')>=VIB (IS).
5. *Disjoint Modules*: At a moment the VIB of a system IS = <BO, R> made of two disjoint modules m_1, m_2 is equal to the maximum of the VIB of m_1 and m_2. Therefore VIB (IS) = max {VIB (m_1), VIB (m_2)}.

Therefore VIB is theoretically validated using length measure. SC, DC and CM can be theoretically validated using complexity measure. In order to validate CM the following complexity properties must be accomplished:

1. *Non-negativity*: The complexity (CM) of a system IS is non-negative as the numbers of BOs and R cannot be negative. So CM (IS)>=0.
2. *Null Value:* If the numbers of R in a IS is zero then there are no BOs and R in the system. Therefore the complexity of the system also is null. So CM (IS) = 0.
3. *Symmetry*: The complexity of a system IS = <BO, R> does not depend on the convention chosen to represent the relationships between its elements. In IS relations R can be represented in two equivalent forms. One is active form i.e. in R form and in passive form i.e. in R^{-1} form but the system and its relationship cannot be affected. So complexity of the system is same for the two equivalent representations. Therefore for IS=<BO,R>and IS^{-1}=<BO,R^{-1}> CM(IS) = Cm(IS^{-1}).
4. *Module Monotonicity*: If IS is a system which have two modules m1=<E_{m1},R_{m1}> and m2=<E_{m2},R_{m2}> and $m1 \cup m2$=IS and $R_{m1} \cap R_{m2} = \Phi$ then CM(IS)>=CM(m1)+CM(m2). Because here modules do not share the relationships but they share BOs. So there may be indirect relationships between the BOs of the IS which is considered in the computation of CM. Besides it if the modules are merged then relations are implicitly generate between the BOs of each modules.
5. *Disjoint Module Additivity*: Let IS be a system which has two disjoint modules m1 and m2. These

modules cannot be merged and the system cannot have indirect relationship. So the CM of IS Isis equal to the sum of its two module's CM. Therefore CM(IS)=CM(m1)+CM(m2).

So CM is valid. In similar way the two metrics SC and DC are also validated using complexity measure.

6. Conclusion

In this paper a theoretical framework has been proposed for quality evaluation of object oriented requirement engineering framework. For this purpose a set of quality metrics have been proposed in three different levels of perspectives namely, conceptual, semantic and framework. The proposed quality metrics are also illustrated with case study based on Business Object based Requirement Engineering framework proposed in [12]. Proposed quality factors in [13] for requirements engineering phase have been formally described in this paper. Also the proposed quality metrics are useful for any general purpose requirements engineering framework. Finally the matrices are theoretically validated using Briand et al.

Future work includes empirical validations of the proposed set of metrics in order to prove their practical utility.

References

1. Lamsweerde, A. V. (2001). *Goal-oriented Requirements Engineering: A Guided Tour*. 5th IEEE International Symposium on Requirements Engineering, 249-263.
2. Kasser, J. E. (2003). *Object-Oriented Requirements Engineering and Management*. Systems Engineering Test and Evaluation (SETE) Conference.
3. OMG, Business Object DTF – Common Business Objects. (1997). OMG Document bom/97-11-11. Retrieved from ftp://ftp.omg.org/pub/docs/bom/97-11-11.pdf.
4. Kaindl, H. (1997). A practical approach to combining requirements definition and object-oriented analysis. *Annals of Software Engineering*, 3(1), 319-343.
5. Vidgen, R. (2003). Requirements analysis and UML - Interaction diagrams and state transition diagrams. *IET Journal of Computing & Control Engineering*, 14(3), 7-11.
6. Vidgen, R. (2003). Requirements analysis and UML use cases and class diagrams. *IET Journal of Computing & Control Engineering*, 14(2), 12-17.
7. Gan, Z. B., Wei, D. W., Zhang, J. L. & Varadharajan, V. (2005). *Business-Process-Oriented Software Requirements Automatic Generator*. 3rd International Conference on Information Technology and Applications, (pp. 1, 95-98).
8. Lu, C. W., Chu, W. C., Chang, C. H. & Wang, C. H. (2007). *A Model-based Object-oriented Approach to Requirement Engineering (MORE).* 31st Annual International Computer Software and Applications Conference, (1, 153-156).
9. Kaiya, H., Horai, H. & Saeki, M. (2002). *AGORA: Attributed Goal-Oriented Requirements Analysis Method. IEEE* Joint International Conference on Requirements Engineering, (pp. 13-22).
10. Kang, K. C., Cohen, S. G., Hess, J. A., Novak, W. E. & Peterson, A. S. (1990). *Feature-Oriented Domain Analysis (FODA) Feasibility Study*. Technical Report, Software Engineering Institute. USA: Carnegie Mellon University.
11. Yang, Y., Xia, F., Zhang, W., Xiao, X., Li, Y. & Li, X. (2008). *Towards Semantic Requirement Engineering*. IEEE International Workshop on Semantic Computing and Systems, 67-71.
12. Sarkar, A. & Debnath, N. C. (2012). Business-object oriented requirements engineering. *Journal of Computational Methods in Sciences and Engineering (JCMSE),* 12(3), 39-51. The Netherlands: IOS Press.
13. Banejee, S. & Sarkar, A. (2012). Evaluation of Object Oriented Requirements Engineering Frameworks: A Study. *International Journal of Computer Applications, Foundation of Computer Science*, 51(10).
14. Briand, L., Morasca, S. & Basili, V. (1996). *Property-Based Software Engineering Measurement*. IEEE Transactions on Software Engineering, (22, 68-86).
15. Sarkar, A. & Debnath, N. C. (2011). *Business Object Oriented Requirements Analysis for Large Scale Information System*. 20th International Conference on Software Engineering and Data Engineering, (SEDE 2011), 103-108.
16. Banerjee, S., Sarkar, A. & Debnath, N. C. (2013). *Quality Evaluation of Requirement Engineering Framework: Business Object Based Approach*. Proceedings of the International Conference on Computing Management & Telecommunications (ComManTel-13).