Received March 13, 2019, accepted April 1, 2019, date of publication April 9, 2019, date of current version April 24, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2909839

A Domain Ontology for Software Requirements Change Management in Global Software Development Environment

ABEER ABDULAZIZ ALSANAD[1,2], AZEDDINE CHIKH[3], AND ABDULRAHMAN MIRZA[2]
[1]College of Computer and Information Sciences/Information Systems, Imam Mohammad Ibn Saud Islamic University, Riyadh 11432, Saudi Arabia
[2]College of Computer and Information Sciences/Information Systems, King Saud University, Riyadh 11362, Saudi Arabia
[3]Department of Computer Sciences, University of Tlemcen, Tlemcen 13000, Algeria

Corresponding author: Abeer Abdulaziz Alsanad (aaasanad@imamu.edu.sa)

ABSTRACT Ontology and its role in both software engineering and knowledge management fields have been widely reported in the literature. Numerous studies have proven the effectiveness of using ontologies to support the process of requirements engineering. One of the main problems facing requirements engineering is that the requirements are exposed to change. In fact, this problem swells even more in global software development (GSD) environment. Given that this problem is unavoidable, we need to improve the requirements change management (RCM) process. In this paper, a systematic domain ontology for RCM especially in GSD environment is proposed. A hybrid method combining Methontology and the 101 method is used for its development; Web Ontology Language (OWL) for its representation; and Protégé for its implementation. It was validated using ontology content evaluation and competency question evaluation methods. Also, it was verified using both ontology taxonomy evaluation and FOCA evaluation methods. Validation and verification results showed that the proposed ontology was successfully built. Building this ontology can be considered as a base contribution in supporting the RCM process in GSD.

INDEX TERMS Requirement engineering, global software development, requirement change management, ontology.

I. INTRODUCTION

At the beginning of the 21st century, the field of ontology, known as ontological engineering, was widely used by computer science and information systems researchers. It can be defined as "the set of activities that concern the ontology development process, the ontology life cycle, and the methodologies, tools and languages for building ontologies" [1]. Generally speaking, ontologies offer a for-mal representation of knowledge. They assist in checking for inconsistency and incompleteness, as well as define a common vocabulary in a specific domain with the purpose of sharing information via the inclusion of basic domain concepts and also the relations between these concepts [2]. There are some motivations for building an ontology, some of which are sharing a common understanding of the structure of information between stakeholders, allowing the reuse of domain knowledge, and making explicit domain assumptions that will allow assumptions to be changed easily when the domain knowledge is changed [3]. In addition, separating domain knowledge from operational knowledge will allow the use of operational knowledge in different domains. Building an ontology will also facilitate the analysis of domain knowledge, which in turn will greatly help in reusing existing ontologies [4]. Particularly for software engineering, ontologies have many advantages such as using the same terminology between software developers for different software applications [5] and, more specifically in software requirements engineering, ensure requirements consistency and facilitate communication between requirements engineers [6]. Actually, requirement change management (RCM) is suffering from a lack of an integrated set of related knowledge [7], [8]. Various studies have proven the effectiveness of using ontologies to support the requirements engineering process [9]–[13]. In addition, global business is the key interest of many organizations, recently due to some economic and strategic

The associate editor coordinating the review of this manuscript and approving it for publication was Mansoor Ahmed.

2169-3536 © 2019 IEEE. Translations and content mining are permitted for academic research only.
Personal use is also permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

gains [14], [15]. Unfortunately, however, applying software development in global software development (GSD) is not so straightforward; there are various challenges mainly related to RCM that are considered as a very demanding activity that needs rich communications [16], [17]. Moreover, as software engineering continued its growth, its projects became more complex and required high quality software [18]. All of these were encouraged us to build an ontology for RCM in GSD. This ontology, that in turn, can be a base to be used in different methods to increase the effectiveness of the RCM process by improving the coordination, communication, and control between stakeholders. More specifically, the ontology can be used with other ontologies as a tool to ensure the semantic correctness of the change request. This paper aims to build a domain ontology for RCM in GSD, which is called a requirement change ontology (RCO). A hybrid method combining Methontology and the 101 method was used for its development; OWL (Web Ontology Language) for its representation; and Protégé for its implementation. Finally, the proposed ontology was validated using ontology content evaluation and competency question evaluation methods. Also, it was verified using both ontology taxonomy evaluation and FOCA evaluation methods. Validation and verification results showed that the proposed ontology was successfully evaluated.

The rest of this paper is divided into the following sections: Related works is presented in section 2. Section 3, describes the method used to build an RCO. The RCO evaluation is presented in section 4. Section 5 presents a discussion about the evaluation results. Finally, section 6, shows the conclusion and future works.

II. RELATED WORKS

In this section, we analyzed literature on covering building ontologies for RCM related issues in GSD on two sides. One side is related to searching through ontology libraries for RCM ontology, especially in the GSD field. The other side is related to finding research papers that are concerned with building an RCM ontology, especially for the GSD environment. We found that the ontology for RCM in a GSD environment is given little importance in both sides.

Before starting the process of ontology modeling and building, it is important to check whether any ontology for the same domain we intend to build exists. Existing ontologies can be found in ontology libraries or by using semantic search engines. The following ontology libraries and search engines were examined, though none contained an RCM ontology for GSD: DAML [19], protege [20] ontology libraries, and Swoogle semantic web search engine [21].

On the other hand, the literature review revealed the scarcity of the number of works showing the building of an RCM ontology in GSD. We found only two works: Khatoon *et al.* [12] and Khatoon *et al.* [22]. These two works proposed an RCM ontology in GSD. Khatoon *et al.* [12] addressed the problem of RCM in GSD by ensuring knowledge management and shared understanding through a software engineering ontology. They used a case study to evaluate their framework, but their ontology is not complete. It missed many important domain concepts. While Khatoon *et al.* [22] developed an RCM ontology for GSD using Protégé, their proposed ontology addressed the issues of RCM during a GSD environment by providing a common understanding to overcome ambiguities in knowledge sharing and management. They used a case study to evaluate their ontology. Their ontology is very abstract since many important concepts are not included in the ontology.

From these works, we noticed that shared knowledge of the RCM process in GSD is lacking. Otherwise, using ontologies in this field was scarcely studied in the associated literature and is also not available in ontology libraries. Accordingly, there is a need to build an RCM ontology in GSD with more details.

III. BUILDING AN RCO: AN RCM ONTOLOGY FOR GSD

Actually, there are many methods used for developing ontologies. Brusa *et al.* [8] divided these methods into two groups: experience-based methods such as Tove and enterprise model approach, and evaluative prototypes models such as Methontology [23]. In fact, there is no one correct method used for developing ontologies. However, diverse methods can be merged together. In this paper, we adopted the same hybrid method that was used in [8] and shown in Figure 1. The first method is Methontology, which enables the construction of ontologies at the knowledge level. In addition, it is considered as the most mature if we compare it with the IEEE standard. Moreover, the Foundation for Intelligent Physical Agents (FIPA) proposed this for ontology construction [24]. The second method is the 101 method, which gives clear steps for building an ontology. The 101 method is described in [5].

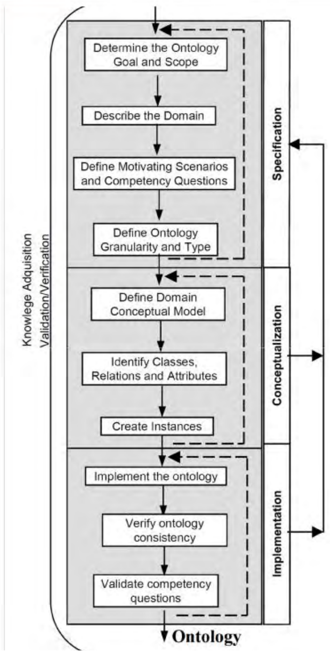

Determine the Ontology Goal and Scope
Describe the Domain
Define Motivating Scenarios and Competency Questions
Define Ontology Granularity and Type
Define Domain Conceptual Model
Identify Classes, Relations and Attributes
Create Instances
Implement the ontology
Verify ontology consistency
Validate competency questions
Ontology
Specification
Conceptualization
Implementation
Knowlege Adquisition
Validation/Verification

FIGURE 1. A domain ontology development process, [8].

The domain ontology building process is composed of three main steps: (1) specification; (2) conceptualization; and (3) implementation. The specification phase gains knowledge about the domain whereas the conceptualization phase organizes and structures this knowledge using external representations. Finally, the implementation phase builds the ontology under some technological environment.

A. SPECIFICATION

1) THE ONTOLOGY GOAL AND SCOPE

The scope binds the ontology by deciding what or what not to include. An RCO covers RCM in GSD environment. It is dedicated for general concepts about RCM but more specifically in the GSD environment.

2) DOMAIN DESCRIPTION

The domain of interest needs to be described and analyzed to obtain the most needed knowledge to build an RCO. We studied and revised the RCM process in the literature, as well as, extracted core knowledge about that domain.

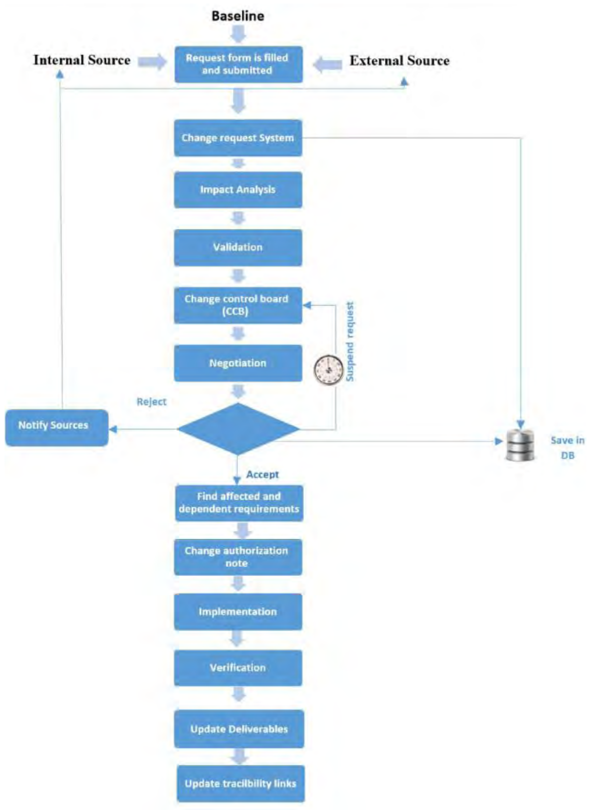


FIGURE 2. RCM life cycle steps, [13].

Moreover, we conducted interviews with a group of experts. As a result of these actions, RCM life cycle steps are briefly described and presented next in Figure 2. The life cycle starts with the change request form that is requested by either internal or external sources. Then the request is recorded. After that, an impact analysis is made. Next, the request is checked for validity, and the change control board(s) (CCBs) negotiate and make a decision on the requested change. If the request is accepted, the affected and dependent requirement is found, a change authorization note is generated, and the change is implemented. The change is then verified to ensure that it has been implemented correctly. The project deliverables, as well as the configuration documents, are updated to accommodate the change. Finally, traceability links are updated as well. If the request is rejected, on the other hand, the change requester is notified. In the case of a request suspension, the request is returned back to the CCB for taking a decision. In the three cases, the request and the action taken to the request were saved in a database.

3) COMPETENCY QUESTIONS

Specifying competency questions in the specification phase is vital since it allows us to determine the ontology scope. Furthermore, it is used in the validation of the ontology by checking if it can answer these questions [12]. Literally, there should be a competency question should be included for each concept in the ontology [8]. A subset of RCM competency questions that covers the main concepts of an RCO is shown in Table 1.

TABLE 1. A subset of competency questions.

CQ	CQ Text
CQ 1	Why requirements are changed?
CQ 2	Who does request the change?
CQ 3	Who is responsible to manage the change?
CQ 4	What is mostly changed?
CQ 5	When is the change requested?
CQ 6	Who does decide on accepting or rejecting the change?
CQ 7	What is the action if the request is accepted or rejected?
CQ 8	How is the negotiation process completed?
CQ 9	How is the change verified and validated?
CQ 10	Where are the change requests saved?

4) ONTOLOGY GRANULARITY AND TYPE

The levels of ontology granularity can be distinguished as upper ontologies, middle ontologies, domain ontologies, and application ontologies [25]. In general, the domain ontology describes the vocabulary related to a specific domain. The RCO ontology specifically describes knowledge related to RCM in GSD. It can be used to facilitate communication among the software project team stakeholders.

B. CONCEPTUALIZATION

1) DOMAIN CONCEPTUAL MODEL

In domain conceptual model step, the basic terms were acquired from an extensive study of literature in the field of RCM in GSD. In addition, RCM processes were reviewed to identify all parties involved in RCM. After that, the partial or total overlapping of concepts, synonyms, properties, relations, and attributes were omitted from the list of terms to formulate the key term list. Some of these key terms include the following: change request, change reason, change type, CCB, priority, traceability, dependability, change closure,

TABLE 2. An extract of the data dictionary of RCO.

Name	Description	Type
Change request	One single change request (The core concept in the ontology)	Concept
Change initiator	Person who initiates the change (Could be internally from the development team or externally from the users or customers).	Concept
Change Request Form	Official form to be filled.	Concept
Impact Analysis	Measurement of the change impact.	Concept
Change Validation	Whether the change is valid or not.	Concept
Change Implementation	Fact to implement the change.	Concept
Initiate	Association between the change initiator and the change request.	Relation
Submitted using	Association between the change request and the form.	Relation
Performed by	Association between the change implementation and the change builder.	Relation

updated deliverables, change acceptance or rejection, negotiation, change request log, change implementation, change validation, change verification, change report.

Afterward, a data dictionary for the key terms was created to provide an intermediate representation. The data dictionary gives descriptions of concepts and relations in the domain. An extract of the data dictionary is shown in Table 2. After that, a UML (Unified Modeling Language) class diagram was used to provide a better understanding of the domain conceptual model through classes, attributes, and relations. It provides a base for building the ontology term glossary while including other concepts by means of generalization and specialization techniques. Figure 3 shows the UML domain model.

2) INSTANCE DEFINITION

At this stage, the instances of each concept were defined in the instance table. The instance table includes the name of the instance, the concept it belongs to, the attributes, and their values. Table 3 provides an extract of the instance table.

C. IMPLEMENTATION

The RCO was represented using OWL and implemented using the Protégé editor. OWL was used to explicitly describe the knowledge in the ontology and will determine how the knowledge will be stored. OWL was chosen since it is recommended for its ability to represent ontologies and for its expressiveness and powerful representation. The Protégé editor was used to implement the RCO [26] because it is free, open-source, and extensible. Moreover, it can be exported to RDF Schema (RDFS) and OWL. Furthermore, it can be used to validate and verify the ontology [27]. A Protégé OWL API [26] was used to develop and manipulate OWL files. Figure 4 shows a snapshot, taken from Protégé that contains the RCO's entities.

D. THE PROPOSED RCM IN GSD ONTOLOGY (RCO)

The proposed ontology (RCO) contains 129 declared axioms distributed between 79 classes, 43 object properties, and 7 data properties. The classes include concepts about request change, as shown in Figure 5, and concepts about GSD, as shown in Figure 6.

Indeed, using RCO will unify the language used by all stakeholders involved in the change request process, from the change requester to the change builder. This can improve communication, which is the greatest problem faced in a GSD environment, between them and reduce potential misunderstandings. Moreover, RCO could be involved in other methods to ensure the correctness of the change request. The following list provides not only examples of some common mistakes found in a requirement change request in a GSD environment but also justification for how an RCO can avoid or limit such mistakes:

- A common mistake is requesting the same change that has been requested before. With an RCO any requested change is checked first to see if it was requested before and the action taken to that request. All previous request changes are saved in the RCM database.
- Any new requests for change are sent to the change manager for validation. With an RCO, a request is checked first in the expected change repository. If it is available, making a decision on the request will be easier since the request was expected before requesting it.
- The change request is not approved by all CCBs located in different cities. With an RCO, all CCBs' opinions should be taken in consideration.
- A change requester requests a change without determining the origin of the change. When the CCB, wants to make a decision he or she needs to know in which part of the development the request is granted. With an RCO, this will be avoided since the origin of each request needs to be specified.

By looking at these examples, we can say that an RCO has many advantages for the requirement engineering field. An RCO can be reused in different ways. One of the possible uses of an RCO is the participation of this ontology with other ontologies to improve the semantic correctness of change requests. This in turn improves the coordination, communication, and control of RCM in GSD.

IV. RCO EVALUATION

Ontology evaluation plays a major role for ontology engineers in ensuring that their ontology is correct so that they can publish it for public use [28]. Ontology evaluation can be defined as "a technical judgment of the content of the ontology with respect to a frame of reference during every phase and between phases of their life cycle" [24]. To evaluate the RCO, the criteria-based evaluation approach was used.

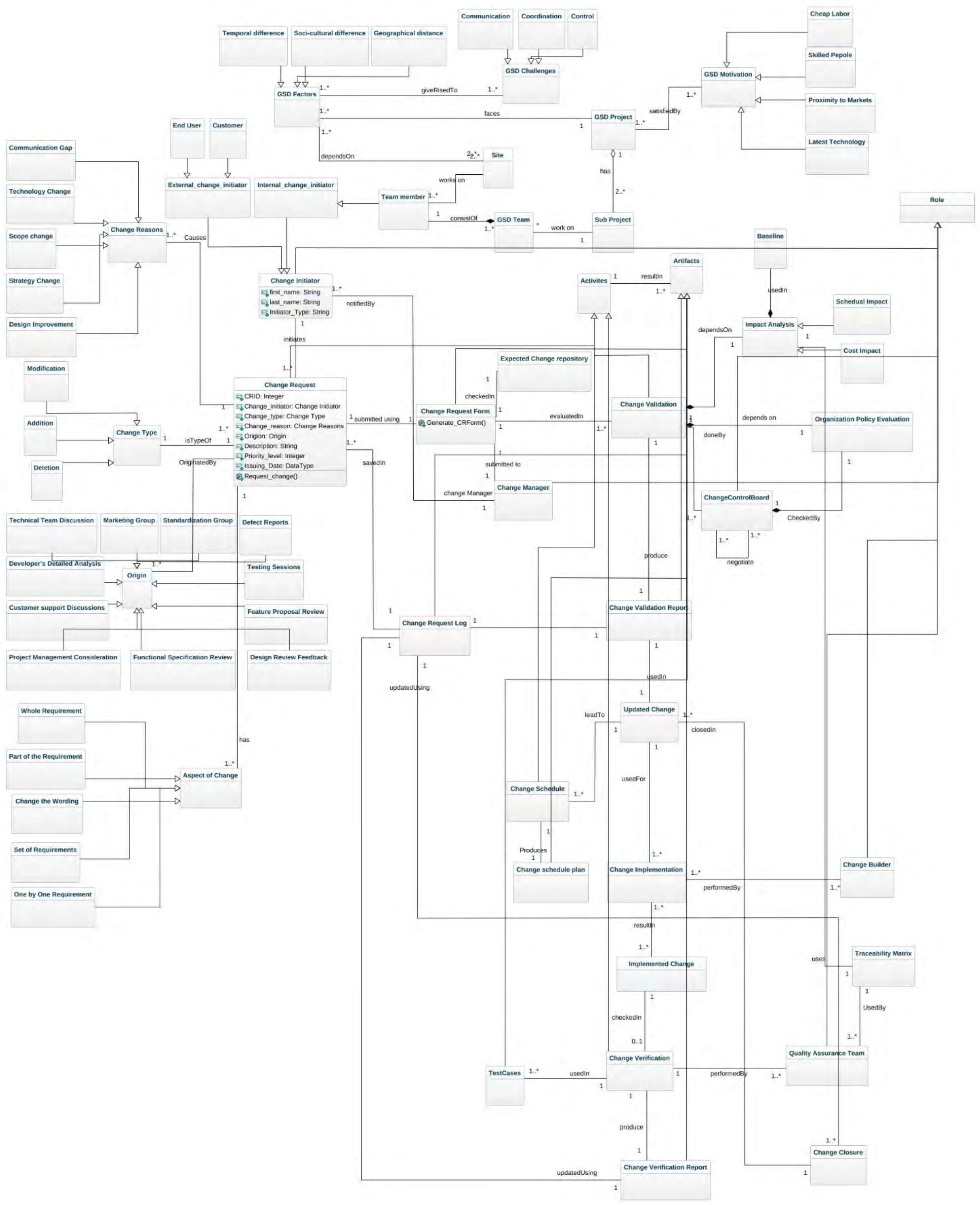
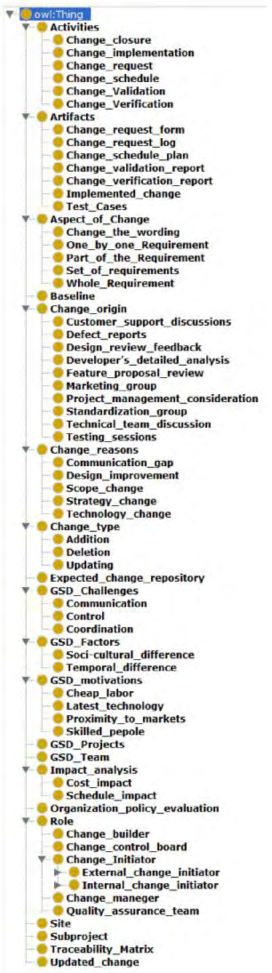


FIGURE 3. **Class diagram for RCM.**

An ontology evaluation consists of two parts: ontology validation and ontology verification [29]. Ontology validation checks if the correct ontology has been built, whereas ontology verification checks if the ontology has been built correctly. Ontology verification confirms that the ontology has been built according to certain specified ontology quality criteria. Validation is achieved by applying two validation methods. The first is the ontology content evaluation, and the second is answering competency questions that were outlined in the specification phase prior to designing the RCO.

TABLE 3. An extract of the instance table.

Instance	Concept	Attribute	Value
Change request 18	Change request	CRID	18
		Change initiator	Change initiator 29
		Change reason	Technology Change
		Change type	Modification
		Origin	Marketing Group
		Description	Change the marketing tool from Facebook to Instagram and Snapchat
		Priority level	Medium
		Issuing date	20-11-2017
Change initiator 29	Change initiator	First name	Ahmed
		Last name	AlOmar
		Initiator type	External



- owl:Thing
  - Activities
    - Change_closure
    - Change_implementation
    - Change_request
    - Change_schedule
    - Change_Validation
    - Change_Verification
  - Artifacts
    - Change_request_form
    - Change_request_log
    - Change_schedule_plan
    - Change_validation_report
    - Change_verification_report
    - Implemented_change
    - Test_Cases
  - Aspect_of_Change
    - Change_the_wording
    - One_by_one_Requirement
    - Part_of_the_Requirement
    - Set_of_requirements
    - Whole_Requirement
  - Baseline
  - Change_origin
    - Customer_support_discussions
    - Defect_reports
    - Design_review_feedback
    - Developer's_detailed_analysis
    - Feature_proposal_review
    - Marketing_group
    - Project_management_consideration
    - Standardization_group
    - Technical_team_discussion
    - Testing_sessions
  - Change_reasons
    - Communication_gap
    - Design_improvement
    - Scope_change
    - Strategy_change
    - Technology_change
  - Change_type
    - Addition
    - Deletion
    - Updating
  - Expected_change_repository
  - GSD_Challenges
    - Communication
    - Control
    - Coordination
  - GSD_Factors
    - Soci-cultural_difference
    - Temporal_difference
  - GSD_motivations
    - Cheap_labor
    - Latest_technology
    - Proximity_to_markets
    - Skilled_pepole
  - GSD_Projects
  - GSD_Team
  - Impact_analysis
    - Cost_impact
    - Schedule_impact
  - Organization_policy_evaluation
  - Role
    - Change_builder
    - Change_control_board
    - Change_Initiator
      - External_change_initiator
      - Internal_change_initiator
    - Change_maneger
    - Quality_assurance_team
  - Site
  - Subproject
  - Traceability_Matrix
  - Updated_change

FIGURE 4. Protégé snapshot of RCO.

Verification is achieved using two methods, also. The first is the ontology taxonomy evaluation, and the second is the FOCA methodology. FOCA introduces several measures, as well as a framework for evaluating the measures, making it possible for an ontology developer of any experience level to evaluate their design [30]. The details of applying the two parts of the RCO evaluation, which are validation and verification, are stated in the following section.

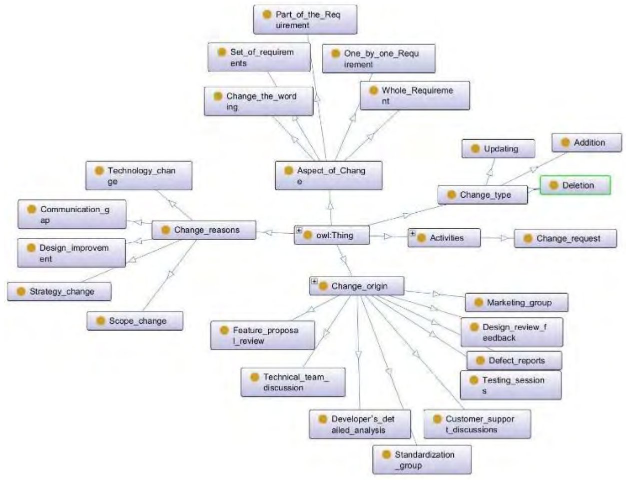


FIGURE 5. Request change concepts.

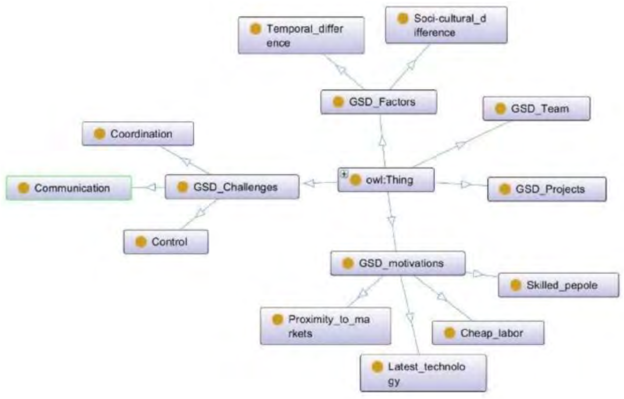


FIGURE 6. GSD concepts.

TABLE 4. Ontology content evaluation.

Criteria	Satisfaction
Consistency	Yes, since no contradictory knowledge can be inferred from all definitions and axioms. Also, reasoner shows no errors.
Completeness	Yes, it is complete based on specifications determined in the design phase of the ontology.
Conciseness	Yes, the ontology is concise since does not contain any unnecessary concepts.
Expandability	Yes, it is easily expanded since there is no need to make big changes in a set of well defined definitions when adding new definitions.
Sensitiveness	The ontology is not sensitive since small changes in definition will not alter a set of well-defined concepts.

A. RCO VALIDATION

1) ONTOLOGY CONTENT EVALUATION

This method checks the content of the ontology based on the following main criteria [29], [31]: consistency, completeness and conciseness, and sensitiveness. The criteria and their compatibility to RCO are shown in Table 4.

2) COMPETENCY QUESTIONS EVALUATION

The competency questions that were stated in Table 1 for determining the RCO's scope and designing purposes are

TABLE 5. Competency question answers and justifications.

CQ #	CQ	Justification
CQ 1	Why requirements are changed?	Requirements are changed because of several reasons. Most famous reasons that are found in literature are stated as subclasses of Change reasons class. Which are: communication gap, design improvement, scope change, strategy change and technology change.
CQ 2	Who does request the change?	The change is requested by change initiator. Stated that the change initiator can be one of two kinds: 1.Internal: which can be one of the team members. 2. External: which can either the customer or the user.
CQ 3	Who is responsible to manage the change?	Managing the change is the responsibility of Change manager, who manages details of all the changes with their status.
CQ 4	What is mostly changed?	There are many aspects of change. Most famous aspects of change found in literature are stated as subclasses of Aspect of change class. They are: change the wording, one by one change, part of requirement, set of requirements or whole requirement.
CQ 5	When is the change requested?	Changes can be requested during development or after development completion.
CQ 6	Who does decide on accepting or rejecting the change?	The CCB is responsible of accepting or rejecting the change. In the case of GSD environment, there could be more than one CCB. A negotiating process is then necessary for making a decision.
CQ 7	What is the action if the request is accepted or rejected?	In the case that the request is accepted. The change is sent for implementation and verification. In the case that the request is rejected, the requester is notified. In both cases, the change request log is updated with the request status.
CQ 8	How is the negotiation process completed?	The negotiation process between CCBs has a threshold time. Once the time is over the decisions are collected and the final decision is implemented.
CQ 9	How is the change verified and validated?	Validating the change is done by CCB. They check whether the change is valid and could be made or not based on organization policy evaluation and impact analysis. Verifying the change is done by quality assurance team, using test cases.
CQ 10	Where are the change requests saved?	Any request for a change is saved in change request log and the status of that change is updated to the log.

used here for the evaluation. Each competency question was answered and justified based on the RCO components. Answers and justifications are shown in Table 5. Competency questions ensure that the ontology implementation fulfills the RCO scope.

B. RCO VERIFICATION

1) ONTOLOGY TAXONOMY EVALUATION

The taxonomy evaluation method is used for checking the taxonomy of the ontology based on main criteria mentioned in [31]. These criteria and their compatibility to RCO are shown in Table 6.

TABLE 6. Ontology taxonomy evaluation.

Criteria		Satisfaction
Inconsistency	Circularity errors	No error, reasoner shows no errors
	Partition errors	No error, reasoner shows no errors
	Semantic errors	No error, reasoner shows no errors
Incompleteness	Incomplete concept classification	No error, all concepts of the knowledge specified in the design phase are included.
	Partition errors	No error, because all the instances of the base classes belong to the sub classes.
Redundancy	Grammatical redundancy	No error, each class has only one definition.
	Identical formal definition of some classes	No error, there is no two classes with the same definition.
	Identical formal definition of some instances	No error, in domain ontology there are no instances.

2) FOCA EVALUATION

FOCA is a method that can be used for evaluating the quality of an ontology. FOCA includes determining the type of ontology, a questionnaire to evaluate the components, a framework to follow, and finally, a statistical model that calculates the quality of the ontology [30]. FOCA goes through three verification steps, as shown in Figure 7.

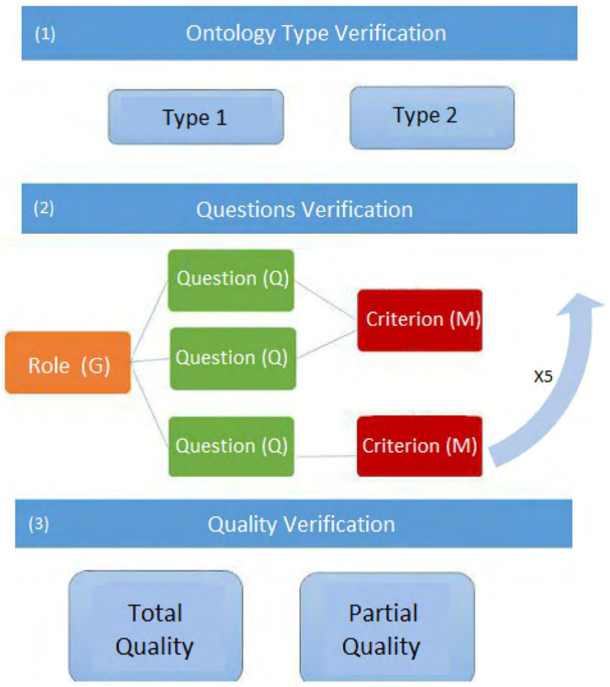


FIGURE 7. FOCA method, [30].

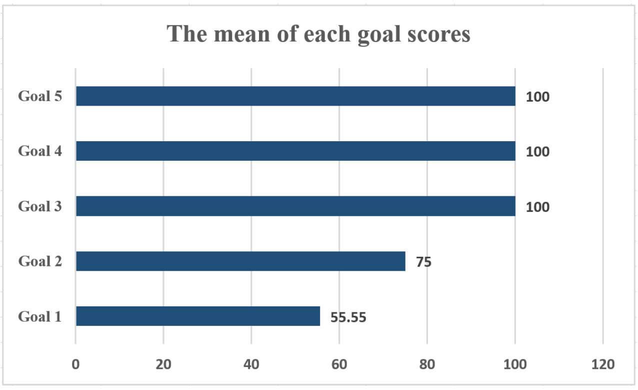


FIGURE 8. The percentage of the mean of the scores of each goal.

An RCO is verified using the following steps of the FOCA method:

1) Step 1- Ontology Type Verification: First, the ontology type should be specified. Two types are defined in FOCA: The first is dedicated for a task or domain ontology, while the second one is for application ontology. Because the ontology is a domain ontology, it should be considered a type 1. Based on FOCA, a type 1 ontology should answer Q5 instead of Q4 for goal 2, as detailed next.
2) Step 2- Questions Verification: In this step, 12 out of 13 questions (one of the questions will not be answered based on the chosen type in step 1) should be answered. These answers should then be scored by the evaluator. Each question fulfills one of the ontology quality criteria. These 12 questions serve five goals: Goal 1 is mainly concerned about competency questions and reuse. Goal 2 ensures that the ontology's terms meet the level expected. Goal 3 checks for contradictions or invalid reuse of terms in the ontology. Goal 4 is about reasoning and reasoner performance. Goal 5 is on ontology documentation and ensuring consistency between the modeled ontology and the design. The goal/question/metric (GQM) approach for the FOCA methodology is shown in Table 7. The GQM approach contains the goals and their corresponding questions and metrics. Five metrics are specified: completeness, adaptability, conciseness, computational efficiency, and clarity. Each question has a description that explains how to verify it. Using these descriptions, the evaluator decides on the score for each question, and the mean of the scores of each goal was calculated and shown in Figure 8. It was noticed that all goals obtain 100% except goals 1 and 2. Goal 1 got 55.55% because no other ontologies were reused while building the RCO. It was built from scratch. Goal 2 got 75% because RCO provides moderate abstraction for those not in full abstraction.
3) Step 3- Quality Verification: In this step, quality verification can be categorized into two kinds: total quality verification and partial quality verification. In this work, the total quality verification was chosen because most goals are considered in the evaluation. Total quality verification was calculated using beta regression models, proposed by Ferrari in [32] and shown in (1). The result for quality ranges between 0 and 1.

TABLE 7. The GQM of FOCA methodology, [30].

Goal	Question	Metric
1.Check if the ontology complies with Substitute.	Q1. Were the competency questions defined?	Completeness
	Q2. Were the competency questions answered?	Completeness
	Q3. Did the ontology reuse other ontologies?	Adaptability
2. Check if the ontology complies Ontological Commitments.	Q4. Did the ontology impose a minimal ontological commitment?	Conciseness
	Q5. Did the ontology impose a maximum ontological commitment?	Conciseness
	Q6. Are the ontology properties coherent with the domain?	Conciseness
3. Check if the ontology complies with Intelligent Reasoning.	Q7. Are there contradictory axioms?	Conciseness
	Q8. Are there redundant axioms?	Conciseness.
4. Check if the ontology complies Efficient Computation	Q9. Did the reasoner bring modelling errors?	Computational Efficiency.
	Q10. Did the reasoner perform quickly?	Computational Efficiency.
5. Check if the ontology complies with Human Expression.	Q11. Is the documentation consistent with modelling?	Clarity
	Q12. Were the concepts well written?	Clarity
	Q13. Are there annotations in the ontology that show the definitions of the concepts?	Clarity

$$\hat{\mu}_i = \frac{\exp\left\{-0.44 + 0.03\,(Cov_s \times Sb)_i + 0.02\,(Cov_c \times Co)_i + 0.01\,(Cov_R \times \mathrm{Re})_i + 0.02\,(\mathrm{Cov_{Cp}} \times Cp)_i - 0.66\,Exp_i - 25(0.1 \times Nl)_i\right\}}{1+\exp\left\{-0.44+0.03\,(Cov_s \times Sb)_i + 0.02\,(Cov_c \times Co)_i + 0.01\,(Cov_R \times \mathrm{Re})_i + 0.02\,(Cov_{Cp} \times Cp)_i - 0.66 Exp_i - 25(0.1 \times Nl)_i\right.} \tag{1}$$

To calculate the total quality:

- Cov_S is the mean of the grades from goal 1.
- Cov_C is the mean of the grades from goal 2.
- Cov_R is the mean of the grades from goal 3.
- Cov_{Cp} is the mean of the grades from goal 4.
- LExp is the variable for evaluator experience, with 1 being very experienced and 0 being not experienced at all.
- Nl is 1 if a goal is not possible to evaluate (or if any question could not be evaluated).
- sb = 1, Co = 1, Re = 1, Cp = 1 because all goals are considered, 0 if that goal is not considered.

TABLE 8. Comparison between RCO and the other two previous works.

Reference	Khatoon et.al. [22]	Khatoon et.al. [12]	RCO
# of General classes	3	8	3
#of GSD classes	14	24	17
# of RCM classes	12	27	59
Total # of classes	29	59	79
Coverage	low	medium	high
Tool used	Protégé	Protégé	Protégé

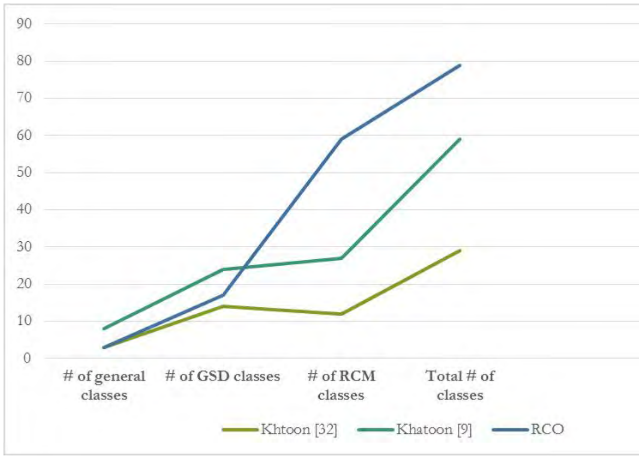


FIGURE 9. Comparison of number of classes between RCO and the other two previous works.

By substituting these values in the beta regression model, the following results emerged:

$$\mu i = \frac{exp(1.2265 + 1.5 + 1 + 2 + 0 + 0)}{1 + exp(1.2265 + 1.5 + 1 + 2 + 0 + 0)} \quad (1.a)$$

$$\mu i = \frac{exp(5.7265)}{1 + exp(5.7265)} = 0.997 \quad (1.b)$$

The result of the total quality is 0.997, which is very near to 1. This shows the high quality of the RCO.

V. DISCUSSION

As shown in section 3 after building the RCO, it was evaluated using validation and verification methods. The validation methods show that all criteria for evaluating the content on the RCO were achieved. Also, all competency questions were clearly answered and justified. The objective of building the ontology, which was specified in the design phase, was noticeably achieved. Regarding RCO verification, the RCO verified using both the ontology taxonomy evaluation and the FOCA evaluation methods. All criteria in the taxonomy evaluation were satisfied, and no violations were found in the ontology taxonomy. Furthermore, the FOCA method was followed step by step and all the steps were successfully applied. The result for total quality indicates the high quality of the RCO fulfilling almost all the most important basics of building an ontology. By achieving both validation and verification, it can be concluded that the RCO was evaluated successfully and is ready to be used in various applications.

Moreover, a comparison of the RCO with the previous two works mentioned in section 2 are shown in Table 8, which also shows the RCO improvement compared with previous works. The RCO contains 79 classes expressing RCM in the GSD domain. This is 20 classes more than the Khatoon [12] ontology and 50 classes more than the ontology in [22]. Figure 9 shows a comparison by number of classes.

VI. CONCLUSION AND FUTURE WORK

In conclusion, the aim of this paper can be summarized as building and evaluating an ontology for RCM in an GSD environment, called an RCO. A significant motivator for choosing this contribution comes from the scarcity in the ontology field regarding RCM in GSD. Therefore, building an RCO would be a valuable addition to ontology libraries. Also, the advantages of building such an ontology are useful both for software engineering and knowledge management fields. Furthermore, it can be used to mitigate miscommunication and misunderstanding issues, which are major problems facing developers in the GSD environment. This can be achieved by using the RCO, as well as other ontologies, to ensure the semantic correctness of change requests and increase their reliability. Software reliability is the most critical part since it depends on time and cost, which are two main factors affecting the quality of the soft-ware as a whole [33]. In this paper, all the steps of building this ontology were mentioned. Moreover, four methods for evaluating this ontology were used: two methods for the validation purpose and the two others for the verification purpose. Our results showed that the RCO was successfully built and evaluated.

For future work, we suggest first integrating our RCO ontology with other application domain ontologies (requirements engineering ontology, specific application domain ontology such as payroll ontology) to come up with a strong comprehensive ontology that can cover all the development aspects of a given software application. Moreover, the RCO can be translated to other languages like French and German to expand its usability.

REFERENCES

[1] S. Cakula and A.-B. M. Salem, "E-learning developing using ontological engineering," *WSEAS Trans. Inf. Sci. Appl.*, vol. 10, no. 1, pp. 14–25, 2013.
[2] R. Gayathri and V. Uma, "Ontology based knowledge representation technique, domain modeling languages and planners for robotic path planning: A survey," *ICT Express*, vol. 3, no. 2, pp. 69–74, 2018.
[3] F. Ruiz and J. Hilera, "Using ontologies in software engineering and technology," in *Ontologies for Software Engineering and Software Technology*, C. Calero, F. Ruiz, and M. Piattini, Eds. Berlin, Germany: Springer, 2006, pp. 49–102.
[4] S. Chuprina, V. Alexandrov, and N. Alexandrov, "Using ontology engineering methods to improve computer science and data science skills," in *Proc. Int. Conf. Comput. Sci.*, vol. 80, pp. 1780–1790, Jan. 2016.
[5] N. F. Noy and D. L. Mcguinness, "Ontology development 101: A guide to creating your first ontology," Stanford Knowl. Syst. Lab., Stanford, CA, USA, Tech. Rep. KSL-01-05 and Stanford Med. Inform., Stanford, CA, USA, Tech. Rep. SMI-2001-0880, 2001. [Online]. Available: https://protege.stanford.edu/publications/ontology_development/ontology101.pdf

[6] V. Castañeda, L. Ballejos, M. L. Caliusco, and M. R. Galli, "The use of ontologies in requirements engineering," *Global J. Res. Eng.*, vol. 10, no. 6, pp. 2–8, 2010. [Online]. Available: 0.5121/ijitcs.2016.6104

[7] C. Bezerra, F. Freitas, and F. Santana, "Evaluating ontologies with competency questions," in *Proc. IEEE/WIC/ACM Int. Conf. Web Intell. (WI), Intell. Agent Technol. (IAT)*, vol. 3, Nov. 2013, pp. 284–285.

[8] G. Brusa, M. L. Caliusco, and O. Chiotti, "A process for building a domain ontology: An experience in developing a government budgetary ontology," in *Proc. 2nd Australas. Workshop Adv. Ontologies*, Hobart, TAS, Australia, vol. 72, Dec. 2006, pp. 7–15.

[9] S. A. Kumar and T. A. Kumar, "Study the impact of requirements management characteristics in global software development projects: An ontology based approach," *Int. J. Softw. Eng. Appl.*, vol. 2, no. 4, pp. 107–125, 2011.

[10] R. Lai and N. Ali, "A requirements management method for global software development," *Adv. Inf. Sci.*, vol. 1, no. 1, pp. 38–58, 2013.

[11] Y. Hafeez *et al.*, "A requirement change management framework for distributed software environment," in *Proc. 7th Int. Conf. Comput. Converg. Technol. (ICCCT)*, Seoul, South Korea, Dec. 2012, pp. 944–948.

[12] A. Khatoon, Y. Hafeez, and T. Ali, "An ontological framework for requirement change management in distributed environment," *Nucleus*, vol. 46, no. 3, pp. 291–301, 2014.

[13] A. AlSanad and A. Chikh, "Software requirements change management—A comprehensive model," in *Recent Advances in Information Systems and Technologies* (Advances in Intelligent Systems and Computing), vol. 569, Á. Rocha, A. Correia, H. Adeli, L. Reis, and S. Costanzo, Eds. Cham, Switzerland: Springer, 2017, pp. 821–830.

[14] M. A. Akbar, Nasrullah, M. Shafiq, J. Ahmad, M. Mateen, and M. T. Riaz, "AZ-Model of software requirements change management in global software development," in *Proc. ICE Cube*, Nov. 2018, pp. 1–6.

[15] M. A. Akbar *et al.*, "Improving the quality of software development process by introducing a new methodology–AZ-model," *IEEE Access*, vol. 6, pp. 4811–4823, 2018.

[16] M. Shafiq *et al.*, "Effect of project management in requirements engineering and requirements change management processes for global software development," *IEEE Access*, vol. 6, pp. 25747–25763, 2018.

[17] M. A. Akbar, Nasrullah, M. Shameem, J. Ahmad, A. Maqbool, and K. Abbas, "Investigation of project administration related challenging factors of requirements change management in global software development: A systematic literature review," in *Proc. ICE Cube*, Nov. 2018, pp. 1–7.

[18] M. A. Akbar *et al.*, "Statistical analysis of the effects of heavyweight and lightweight methodologies on the six-pointed star model," *IEEE Access*, vol. 6, pp. 8066–8079, 2018.

[19] DAML. (2019). *DAML*. [Online]. Available: http://www.daml.org/ ontologies/

[20] (2019). *Protege Ontology Library*. [Online]. Available: https://protegewiki.stanford.edu/wiki/Protege_Ontology_Library

[21] Swoogle. (Jan. 2019). *Swoogle*. [Online]. Available: http://swoogle.umbc.edu/

[22] A. Khatoon, Y. H. Motla, M. Azeem, H. Naz, and S. Nazir, "Requirement change management for global software development using ontology," in *Proc. 9th Int. Conf. Emerg. Technol. (ICET)*, Dec. 2013, pp. 1–6.

[23] M. S. Rani, S. John, and N. Shah, "Proposal of an Hybrid methodology for development by extending the process models of software engineering," *Int. J. Inf. Technol. Converg. Services*, vol. 6, no. 1, pp. 37–44, 2016.

[24] A. Gómez-Pérez, M. Fernandez-Lopez, and O. Corcho, *Ontological Engineering With Examples from the Areas of Knowledge Management, e-Commerce and the Semantic Web*. London, U.K.: Springer, 2004.

[25] S. M. F. Haque and S. Vemura, "Levels of granularity of ontologies and their applications," *Int. J. Adv. Soft Comput. Technol.*, vol. 2, no. 1, 2012.

[26] Protege. (2017). *Protege*. Accessed: Apr. 24, 2017. [Online]. Available: http://protege.stanford.edu/

[27] H. Knublauch *et al.*, "The protégé OWL experience," in *Proc. Int. Semantic Web Conf. (ISWC)*, Galway, Ireland, vol. 188, Nov. 2005, pp. 1–11.

[28] D. Vrandecic, "Ontology evaluation," Ph.D. dissertation, Fac. Econ., Karlsruhe Inst. Technol., Karlsruhe, Germany, 2010.

[29] A. Gómez-Pérez, "Ontology evaluation," in *Handbook on Ontologies* (International Handbooks on Information Systems), S. Staab and R. Studer, Eds. Berlin, Germany: Springer, 2004, pp. 251–273.

[30] J. Bandeira, I. I. Bittencourt, P. Espinheira, and S. Isotani. (2016). "FOCA: A Methodology for Ontology Evaluation." [Online]. Available: https://arxiv.org/abs/1612.03353

[31] S. Lovrenčić and M. Čubrilo, "Ontology evaluation-comprising verification and validation," in *Proc. 19th Central Eur. Conf. Inf. Intell. Syst.*, 2008, pp. 657–663.

[32] S. Ferrari and F. Cribari-Neto, "Beta regression for modelling rates and proportions," *J. Appl. Statist.*, vol. 31, no. 7, pp. 799–815, 2004.

[33] A. Mateen and M. A. Akbar. (2016). "Estimating software reliability in maintenance phase through ann and statistics." [Online]. Available: https://arxiv.org/abs/1605.00774

ABEER ABDULAZIZ ALSANAD received the bachelor's degree in computer science from Prince Sultan University and the master's degree in information systems from King Saud University, where she is currently pursuing the Ph.D. degree in information systems with the College of Computer and Information Science. She is also a Lecturer with the Information Systems Department, College of Computer and Information Sciences, Imam Muhammad Ibn Saud Islamic University. She has published several conference papers. Her major interests include software engineering, requirement engineering, and change management.

AZEDDINE CHIKH received the master's degree in information systems from the National Institute of Computer Sciences, Algeria, in 1994, the degree in systems engineering from the University of Missouri Rolla, USA, in 2003, the Ph.D. degree in information systems from the National Institute of Computer Sciences and Paul Sabatier University, France, in 2004, and the master's degree in e-learning from Switzerland, in 2007. He was an Associate Professor with King Saud University, Saudi Arabia. He is currently a Professor in computer sciences and the Director of Research Laboratory LRIT at University of Tlemcen-Algeria. His research interests include learning design, communities of practice, semantic web, and software requirements engineering.

ABDULRAHMAN MIRZA received the Ph.D. degree in computer science from the Illinois Institute of Technology. He is currently a Professor of information systems with King Saud University (KSU). He is currently a consultant at the Deputyship of Planning and Development, Ministry of Education, and the Acting Director of the Center for Research on Educational Policies. Some of his previous leadership positions include the Vice Dean of Academic Affairs at the College of Computer and Information Sciences, KSU. He is the General Supervisor of the General Directorate of Teachers Affairs at the Ministry of Education. He also served as a Senior Advisor of the Minister of Education and the Minister of Higher Education. He had also held other positions, such as the Director of Quality and Accreditation at Saudi Electronic University, the Deputy Director of the Center of Excellence in Information Assurance, the Chairman of the Information Systems Department, and the CIO at the King Abdullah Foundation for Developmental Housing. His research interests include software engineering, e-commerce, and information security.