© 2020 *Afr. J. Comp. & ICT* – All Rights Reserved
https://afrjcict.net

A Zachman's Architecture-based Framework for Classification and Allocation of Software Requirement Elicitation Interview Questions

Hamzat Olanrewaju Aliyu[1*], Salihu Abdulkadir[2] and Tajudeen Adeleke Badmos[3]
[1,2]School of Information and Communication Technology,
Federal University of Technology, Minna,
Nigeria.
[3]Industry & Innovation Research Institute (IIRI),
Sheffield Halam University,
United Kingdom.
Email:[1]hamzat.aliyu@gmail.com, [2]salihulapai@gmail.com, [3]agbeketajudeen@gmail.com

**Corresponding author*

ABSTRACT

Elicitation of precise software requirements is a complex task involving multiple stakeholders with disparate concerns and viewpoints about the intended software. The need to build frameworks with tool supports to aid communications among these stakeholders has come to the fore in recent times. This paper presents a requirements elicitation framework that leverages the principles of the Zachman's Enterprise Architecture to classify requirements elicitation interview questions into categories that address different aspects of a software system and allocate the categories to suitable (groups of) stakeholders. The proposed framework uses a two-dimensional matrix to allocate questions to stakeholders based on the domain abstraction covered by the question and the stakeholder's perspective of the domain. This paper presents a formal specification of the framework, its supporting algorithms and a prototype implementation which is used as a proof of concept to classify and allocate the elicitation questions for a local software development project.

Keywords: *ReQueClass, Requirements Elicitation, Zachman's Enterprise Architecture, Requirements Interview Question Classification, Kipling's Interrogatives, Software Domain Abstractions, Stakeholders' Perspectives.*

African Journal of Computing & ICT Reference Format:

Hamzat Olanrewaju Aliyu, Salihu Abdulkadir and Tajudeen Adeleke Badmos (2020), A Zachman's Architecture-based Framework for Classification and Allocation of Software Requirement Elicitation Interview Questions, *Afr. J. Comp. & ICT*, Vol. 13, No. 3, pp. 70 – 89.

© Afr. J. Comp. & ICT, September 2020; P-ISSN 2006-1781

© 2020 *Afr. J. Comp. & ICT* – All Rights Reserved
https://afrjcict.net

I. INTRODUCTION

Software Requirements Elicitation (SRE) is the process of finding out the requirements for an intended software system by consulting available resources and by communicating with the appropriate stakeholders in the project to produce a clear and complete documentation of their needs [1-3]. It is a fundamental component of the requirement engineering process; doing it inaccurately portends grave consequences such as low software quality, late delivery, and high cost of development [3-5].

There are many requirements elicitation techniques in practice and in the literature - interview, questionnaire, prototyping, brainstorming, focus groups, observation, domain workshops, task analysis, domain analysis, introspection, etcetera [4, 6]. The context/domain of the problem to be solved and the Requirements Engineer (RE)'s expertise are fundamental to the choice of a (or a combination of two or more) technique(s) to elicit requirements for a software project [4, 6]. Nevertheless, interview maintains the reputation of the most commonly used, either solely or to complement other techniques to elicit software requirements [1, 7, 8].

Unfortunately, the Natural Language (NL) upon which elicitation interviews depend for two-way communications between the RE and the project stakeholders is plagued with *ambiguity and imprecision*; thereby hindering effective communications and knowledge transfers among the stakeholders in requirements elicitation [9-11]. A study by Egbokhare [12] has identified poor understanding of user's requirements as one of the major causes of failure of software projects. Therefore, it is pertinent that both parties use clear forms of communication to provide and analyze requirements that will lead to the precise definition of the problem. Separate research findings by Sutcliffe and Sawyer [11] and Distanont, et al. [13] suggest that most problems of inconsistency and incompleteness in requirements engineering are traceable to communication gaps among stakeholders, which hamper common understanding and lead to misinterpretations of intentions. Arguably, we cannot exonerate ambiguities in the expressions of questions and/or responses from the cardinal root causes of such communication gaps.

A defective requirements specification (RS) has characteristics such as inconsistency and/or incompleteness among many others [14, 15]. According to an extensive study and analysis of several defects in RSs by Langenfeld, et al. [14], incompleteness ranked among the most frequently occurring while both inconsistency and incompleteness ranked among the most costly to fix. Hence, the need to expend some research efforts on addressing some of the root causes of inconsistency and incompleteness in requirements engineering; particularly, ambiguity and imprecision in requirement questions during elicitation interviews.

In addition to the ambiguousness and imprecision of NL, other sources of inconsistency and/or incompleteness in RSs, which are widely acknowledged in the literature, include a) frequent changes in stakeholders' wants during the software development process [9, 10], and b) insufficient knowledge of the domain within which the software is to be built by the RE and at times even other stakeholders [9]. Dynamics of stakeholders' wants is widely acknowledged in the literature [11] and is, in fact, considered inevitable by some authors [4]. Currently, it is been handled with requirement management techniques such as requirements tracing and negotiation [9]. Similarly, the impacts of domain knowledge on requirement completeness have been explored extensively especially with ontology-based requirement elicitation [16]; hence, this paper does not directly address issues of (a) and (b).

The focus of this paper is to propose a systematic solution to the problems of 1) Ambiguous/imprecise expressions of certain requirement questions and 2) to whom (among the stakeholders) to direct certain requirement elicitation questions. Chepken [17] has reported the importance of giving due considerations to the diversity of stakeholders in software development lifecycle. We opine that expressing a question with clarity and precision is not sufficient to get a workable answer but also to direct such question to someone with the requisite knowledge and experience to provide correct and useful answer to it. Intuitively, a requirement question that is ambiguously expressed or directed to the wrong person(s) has a tendency of receiving no concrete answer at all or an answer that does not truly portray the intent of the stakeholders; thereby leading to missing or wrong information. This phenomenon can lead to defective

© 2020 *Afr. J. Comp. & ICT* – All Rights Reserved
https://afrjcict.net

RS with missing or wrongly stated requirements [15, 18].

Since different (groups of) stakeholders have different roles to play in the lifecycle of the intended software and, consequently, diverse depths of knowledge about the different aspects/perspectives of the project, it would be reasonable to stratify the questions into categories that are most appropriate for the different (groups of) stakeholders.

Though the problems due to ambiguity of NL in Requirements Engineering have received considerable research attentions, most of the research efforts found in the literature - at the time of writing this article - focus on the ambiguity of NL in the RS document [9, 19] while those at the level of elicitation have received little attention. We opine that elicitation is as important to requirement engineering as the requirement engineering itself is important to the entire software development life cycle; hence it is worthwhile to address requirement engineering problems at the elicitation stage to avoid them being propagated to the RS document in the later stage of the requirement phase.

We propose the Requirement Question Classifier (ReQueClass), a requirement elicitation framework that takes cue from the Zachman's Enterprise Framework (ZEF) [20] to guide a RE to:

1. Structure the requirements elicitation interview questions into categories based on the aspect of the project, which the questions are to elicit and to express each question unambiguously - using appropriate Kipling's interrogative terms [21] - for the stakeholder to comprehend
2. Determine the most suitable (group of) stakeholders to answer the question in order to get workable and less conflicting answers.

We are optimistic that an implementation of this proposal will significantly close the communication gaps between stakeholders during elicitation interviews by reducing misinterpretations of elicitation questions. As a proof of concept, we provide a web-based application to concretize the abstract framework.

In the remainder of this article, Section 2 presents a literature review. This comprises brief backgrounds to some of the concepts and techniques from which we take cues to define the proposed framework in subsequent sections and a review of related works in the literature. Section 3 gives a formal presentation of the proposed framework while Section 4 presents a demonstration of a web-based implementation to concretize the abstract concepts of the framework. Finally, Section 5 presents the conclusion and a highlight of future works.

II. LITERATURE REVIEW

This section is in two parts. The first subsection presents overviews of some theoretical foundations upon which the work in this paper is built. The last section compares existing work related works in the literature to the framework presented in later sections of this paper.

1.1. General Background Information

This subsection presents overviews of the Kipling's 5W1H interrogatives and the Zachman's Enterprise Framework (ZEF) in relation to the work in this paper. This is to enhance the reader's understanding of subsequent sections of this paper.

1.1.1. Kipling's 5W1H Questioning Technique

The Kipling's questioning technique was proposed in the early 1900s by Kipling [21]. He formulated a questioning technique, using the interrogatives *What*, *Who*, *Where*, *When*, *Why*, and *How* (5W1H), to comprehensively capture the majority of the information that people want to know about a news story from different perspectives. Intuitively, it offers the tool to extract information about an action/event, its doer, place and time of occurrence, motive and how it occurred. According to Kipling, the fundamental of communication inherent in the 5W1H primitive interrogatives is that it integrates answers to questions that enable the comprehensive description of complex ideas. Taking this into consideration, we opine that it can as well serve to integrate answers to requirements questions toward the holistic elicitation of complex software requirements.

The technique presently enjoys a multi-disciplinary applications; it is used in journalism to gather and report news [22] and in the healthcare domain by physician to completely harness information from patients for effective diagnoses and treatments [23]. According to Oranje, et al. [23], it has also been applied in academics to enhance students'

© 2020 *Afr. J. Comp. & ICT* – All Rights Reserved
https://afrjcict.net

composition writing ability such as narrative writing and to expand their ideas. Most relevant to the work presented in this paper is its use by Zachman [20] to define an enterprise architecture to explicitly and comprehensively document the information of an enterprise system (see details in next sub-section).

When applied to the SRE domain, the Kipling's techniques can help in harnessing the following information about every requirement statement [23]:

1. **What:** what is the need/requirement captured in the statement?
2. **Who:** who is/are the stakeholders concerned by the need/requirement stated in the requirement statement/question?
3. **Why:** what is (are) the motivation(s) for the need/requirement required in the statement/question?
4. **Where:** Provides information about the software process locations if necessary; where the components are relative to one another.
5. **When:** at what point in time in the software process is the need/requirement expected to be fulfilled? Or when do things happen?
6. **How:** in what way is the need/requirement being fulfilled with the existing system?

1.1.2. Zachman's Enterprise Framework (ZEF)

The ZEF [20], proposed by John Zachman in the late 1980s, is a two-dimensional logical structure to:

i. Classify and organize the total set of descriptive representations that are relevant for describing an enterprise
ii. Help govern the architectural processes to ensure the traceability needed for an enterprise to manage changes.

According to Zachman, such enterprise architecture is paramount to the survival of a dynamic enterprise with increasing complexity in the present information age. He argued that an ontological structure, rather than ad hoc or trial and error setups, is essential to ensure predictability of enterprise processes and production of repeatable results.

Table 1 presents an overview of the ZEF. It is a six-by-six matrix that methodically captures the set of descriptive representations of an enterprise and classifies them along its two dimensions based on i) the key participants in an enterprise cum the specific perspectives from which they view the enterprise system, and ii) the abstractions of the enterprise system.

1.1.2.1. ***Abstractions:*** The columns in ZEF describe abstractions of six aspects of an enterprise and associate each with a specific Kipling's interrogative, apparently to emphasize the coverage of all abstractions necessary to completely represent the different aspects of the enterprise. Table 2 presents brief descriptions of the six abstractions.

2.1.2.2 ***Participants and Perspectives***: Each row in the ZEF describes the representations of the six abstractions of an enterprise from the perspective of a specific participant. Following from the presentation in Table 1, Table 3 introduces the participants and their respective perspectives.

2.1.2.3 Matrix Cells in ZEF: Consequent upon the presentations of the columns and rows of the ZEF matrix. Each cell describes the abstraction of its column using a formalism and level of details that are suitable for its corresponding row (i.e., participant and perspective).We believe the level of details of the ZEF framework presented so far is just sufficient to follow the requirements elicitation framework to be presented in the next section. An interested reader may consult [20] for discussions of ZEF in details.

The ZEF has been widely adopted by system analysts [20, 24] to define information systems architectures. The rows in the function column describe the process of translating the mission of the enterprise into successively more detailed definitions of its operations.

2.2. Related Works

The communication gap between stakeholders in SRE is widely acknowledged in requirements engineering communality as an important problem. This is justified by the high volume of work reported in the literature employing different techniques and approaches to propose solutions to the problem from various dimensions. More relevant to the work presented in this paper in the literature are the works reporting tools and frameworks for improving communications among the stakeholders. The rest of

© 2020 *Afr. J. Comp. & ICT* – All Rights Reserved
https://afrjcict.net

this section presents some of such works in relation to the context of the work reported in this paper.

2.2.1 **ELICA - ELICitation Aid**

Abad et al. [25, 26] opined that one of the major factors responsible for the usual communication barriers between analysts and stakeholders during SRE is the limited understanding of the application domains by the analysts. As a possible solution to the problem, they recently proposed the requirements ELICitation Aid (ELICA) tool to facilitate system analysts' understanding of target application domains.

ELICA is built on Weighted Finite State Transducers (WFSTs) [27, 28], a tool used for dynamic modeling in natural language processing. ELICA uses WFSTs to automate the dynamic extraction of requirements-relevant information from existing domain requirement documents and/or transcribed responses to interviews. The information so extracted complement the information derived from other related textual documents. The kinds of information extracted by ELICA are mainly non-linguistic information such as speakers' confidence level, analytical tone and emotion. The information so extracted is processed in ELICA to help the analysts ascertain the speakers' (interviewees') intention and reduce the chances of the analysts' misconceptions of the interviewees.

In essence, ELICA seeks to facilitate the analyst's understanding of the domain of the software to be built and minimize his/her misconception of the domain expert's intention during SRE. It, however, pays little attention to the tendency of the domain expert to also misconceive the analyst's intention since SRE interviews are a form of two-way communication. The work presented in this paper seeks to fill this gap by defining a framework for expressing SRE questions on the various aspects of the application and determining the most appropriate stakeholders to answer different questions.

2.2.2 **RETTA: A Requirements Elicitation Tool for TrAffic management systems**

Noaeen, et al. [29] developed the RETTA to confront the complexity, uncertainty and imprecision associated with SRE in the Traffic Management (TM) domain. They argued that the complexity of SRE in the TM domain is connected with the diversity of the stakeholders involved in the system. RETTA employs the crowd sourcing technique [30] to elicit requirements for different components of the TM.

Technically, the framework relates each component of TM to a set of possible sources of requirement information about the component as a guide for the RE. This feature of RETTA offers an interesting point of similarity with ReQueClass, which links each group of stakeholders with a unique perspective of the system to be developed. The ideas are, however, different in a number of ways: 1) the classification of subsystems and information sources in the former is domain-specific - specifically for TM domain - while the classification of stakeholders and their corresponding system perspectives in the latter is generic and domain-independent. 2) the stakeholder identification/classification in RETTA is at a higher level than that of ReQueClass; that is, the former identifies all possible sources of information about a sub-system of the TM but does not link any of the sources to a particular aspect/perspective of the subsystem as is considered by the latter. 3) the former is silent about the format of expression of requirement questions while the latter proposes a format to foster a ground for common understanding the parties.

2.2.3 **QuASE: Quality Aware Software Engineering project**

The evolving QuASE project [31-35] has the overall goal to enhance the communication of quality-related requirements among the various participants in a software development process.

QuASE is underpinned by some ontologies and metamodels of quality-related information and their relevant contexts to facilitate the communication of required quality aspects of a software system. To achieve this ultimate goal, the platform leverages on software language engineering techniques [36] by defining appropriate domain-specific languages (DSLs) [37] with the terminologies understood by the different groups of stakeholders and execute translations between the DSLs to facilitate communications. The platform also aims at enabling the reuse of documented related experiences to predict quality-related issues.

In contrast to ReQueClass, the QuASE is not concerned about all aspects of a software system in its entirety; its focus is mainly on quality-related

© 2020 *Afr. J. Comp. & ICT* – All Rights Reserved
https://afrjcict.net

requirements. Moreover, it is not meant to guide the stakeholders on how to express elicitation questions and/or how to determine the most suitable (group of) stakeholders with whom to discuss the different topics as is done in ReQueClass.

2.2.4 ElicitO: A Quality Ontology-Guided NFR Elicitation Tool

ElicitO [38, 39] is a requirement elicitation framework and tool to facilitate the elicitation of non-functional quality requirements. It is underpinned by domain ontologies which provide the relevant knowledge repository that serves as a memory aid to structure elicitation interviews. The group of ontologies upon which the framework is built captures the quality aspects of the functionalities of the system to be built. This enables the user to specify the required quality properties of the functional elements of the domain.

In order to overcome the problem of the analyst having limited knowledge of the domain, ElicitO replaces the analyst with its underlying functional and non-functional ontologies of the domain. By interaction with the framework, a project stakeholder has the privilege to specify and describe requirements/constraints on some pre-defined quality metrics and save them to the requirement database.

The approach of ElicitO offers the interesting benefit of reducing the occurrence of human errors associated with the real interview due to the analyst's limited knowledge of the domain. However, the set of requirements that can be specified is also limited by the expressiveness of the underlying ontologies as there is no room to get related questions emanating from the human analyst's experience.

Moreover, the framework is silent about which of the various stakeholders should specify the quality constraints on which functionality. The work reported in the present paper is different from ElicitO in that it supports two-way communications between the RE and the stakeholders while providing a guide for RE to express interview questions (for common understanding) and a guide for identifying the appropriate (group of) stakeholders to answer the various questions.

2.2.5 UMD: Unified Model of Dependability

UMD [40, 41] is a dependability requirement elicitation framework developed by a group of researchers at the University of Maryland to provide a common language for a consortium of researchers working on a NASA project to specify, communicate and understand dependability requirements.

At the core of UMD is a basic Domain-Specific Language (DSL) language for use by domain experts and REs to identify and explicitly state the dependability requirements of a system. The DSL raises the level of abstraction in dependability requirements engineering by using the concepts event, issue, scope, measure, and reaction to concretize the expressions of requirements about abstract dependability attributes such as safety, performance, integrity and security.

The syntax is considerably expressive in that the concepts it captures can be easily contextualized by domain experts who associate them with relevant entities in their application domains. Nevertheless, UMD is focused only on the elicitation of dependability requirements and does not address the concerns of ReQueClass regarding the interviews for eliciting functional and non-functional requirements.

2.2.6 ReQueClass: A Framework for Classifying Requirement Elicitation Questions Based on Kipling's Technique and Zachman's Enterprise Framework

In a preliminary report of this work, Abdulkadir and Aliyu [42] proposed the ReQueClass, a framework that mirrors the ZEF (we introduced ZEF in Section 2.1.2.) for a systematic classification of requirements elicitation interview questions and subsequent allocation of question sets to different categories of stakeholders. While the classification of elicitation interview questions is to define a uniform formats for presenting questions relating to the same aspects of a software project, the allocation of question categories to stakeholders aims at ensuring that questions are directed to the people who are in the best positions to answer them.

The present paper extends the conceptual framework reported in [42] with the following:

- A formal ontological specification of the framework to provide a rigorous and unambiguous basis for its discussion and a foundation upon which an implementation can be built

© 2020 *Afr. J. Comp. & ICT* – All Rights Reserved
https://afrjcict.net

- Formal descriptions of the relationships between the elements of the ontology to elucidate the classifications of requirements questions and their subsequent allocations to stakeholders
- A formal algorithmic specification of the workflow of the framework to guide any reader who intends to replicate the implementation of the automated processes in the framework in any chosen platform
- A demonstration of a web-based implementation of the framework to concretize the abstract concepts

III. THE PROPOSED FRAMEWORK

This section introduces the proposed framework for Requirements Elicitation Questions Classification (ReQueClass). We take cues from the 5W1H and the ZEF in the definition of the ReQueClass framework. By drawing a parallel between the fundamental objectives of Software Requirements Elicitation (SRE) and that of the combination of 5W1H and ZEF, we see that:

a. SRE seeks to comprehensively extract and methodically reconcile the needs of the stakeholders (e.g., users, decision makers, etc.) in an intended software system to develop a concise requirement document.
b. The duo of 5W1H and ZEF provides complete representations of the significant abstractions of a system from the perspectives of the different actors involved.

By juxtaposition, we see that though (a) is all about the exploration and extraction of information while (b) is aimed at representing existing information; there is a commonality of documenting complete and consistent information involving multiple groups of people with somehow divergent perspectives. Hence we can take lessons from the tools and approaches used in one to improve on the processes in the other.

Looking back at the presentation of ZEF in Table 1, we opine that if questions were to be asked about different perspectives of an enterprise, the participant associated with the perspective in the framework would be in the best position to answer such questions. For example, one would intuitively expect the planner to provide a more useful answer to a question on the business scope than the other participants. Similarly, the builder would be expected to offer a more workable response to a question asked from technological perspective of the enterprise than any other participant. We can replicate this logical structure and division of labor in a framework for requirement elicitation to methodically classify the elicitation interview questions and determine the most appropriate stakeholder to provide the most useful answers to them. We assume the reader has an average knowledge of Sets and Relations to follow a bit of mathematical elements that will be presented later in this section.

3.1 Stakeholders in SRE and Perspectives

In SRE, stakeholder refers to any individual or group that can influence, or be influenced by, the success or failure of a software project [6, 43, 44]. That is, they are the people who stand to gain or lose from the project; hence, they are the source of requirements and, as such, the first point of consultation during requirement elicitation [6, 44].

Ideally, stakeholders are groups or individuals internal and external with different perspectives of the system requirement. In this paper, we describe stakeholders' perspectives as their distinct views of the system to be built. Stakeholders' perspectives are determined largely by their diverse backgrounds, expertise, interests, and personal goals with respect to the software project. Consequently, this suggests that stakeholders that view a project from a certain perspective would be in the best position to answer elicitation interview questions from the same perspective.

It is considered that the context of a project plays a role in the choice of its stakeholders; hence, it is practically impossible to make a ready-made list of all the stakeholders that may be applicable to any software project [45]. In this work, we take cue from the literature (e.g., [43, 44]) by adopting a finite set of four considerably universal and comprehensive categories of stakeholders: sponsors/owners, users, developers, and domain experts to encompass the various stakeholders that may be involved in a software project. Table 4 presents the four categories of stakeholders and the respective perspectives from which they participate in the elicitation process.

© 2020 *Afr. J. Comp. & ICT* – All Rights Reserved
https://afrjcict.net

3.2 **Students' Management System: A Case Study**

As a running example in subsequent sections of this paper, let us introduce the synopsis of a Students' Management System (SMS), a real-life software project of a Nigerian college of education.

The Management Board of the Minna College of Education (CoE) seeks the services of JDLab Nigeria Limited, a software development company to develop a web-based SMS to digitize the management of the records of all students of the institution from admission to graduation. Comprehensively, the students' information to be managed shall include bio-data, all course registrations and exam scores and fees payments.

The students shall use the system to do all kinds of registrations and make all kinds of payments to the institution. The Bursary Department of the CoE shall be able to extract from the system, details of all payments made by students for accounting and financial reporting purposes. The Academic Departments, Registry Academic Office shall use the system to obtain students' information, statistics and academic statuses of different kinds and for other administrative purposes. The Information and Communication Technology (ICT) Department of CoE shall be the custodian and maintainer of the system for the institution.

Following from the presentation of SRE stakeholders in the previous section, we categorize the stakeholders in the SMS project according to Table 5.

3.3. Ontological Model of ReQueClass

Mathematically, the ontology of the Requirement Elicitation Framework (REF) is a tuple that contains five sets and four relations specifying the relationships between the elements of the sets as follows:

$$REF = < Q, Z, K, S, P, \delta_{QZ}, \delta_{Zk}, \delta_{QP}, \delta_{PS} > \quad (1)$$

Where:

Q: Set of all unclassified SRE questions for a software project.

Z: Set of ZEF abstractions; it is defined as:

$$Z = \{data, function, network, people, time, motivation\}$$

K: Set of Kipling's interrogatives, which is defined as:

$$K = \{what, where, who, when, why, how\}.$$

S: set of stakeholders involved in the software project, which is defined as:

$$S = \{owner, domain\_expert, developers, user\};$$

P: set of stakeholders involved in the software project perspectives

$$P = \{business\_model, contextual\_model, usage, development\_life\_cycle\};$$

The remaining four elements of the tuple are functions defined as follows:

$\delta_{QZ}: Q \longrightarrow Z$; a function that matches every question $q \in Q$ to a specific abstraction $z \in Z$.

$\delta_{ZK}: Z \longrightarrow K$; a function that assigns every abstraction $z \in Z$ to a specific interrogative $k \in K$ which is used to express the questions matched to z.

$\delta_{QP}: Q \longrightarrow P$; a function that matches every question $q \in Q$ to a specific SRE perspective $p \in P$.

$\delta_{PS}: P \longrightarrow S$; a function that assigns every SRE perspective $p \in P$ a specific group of stakeholders $s \in S$ who are considered most suitable to answer questions related to perspective p. It is important to note here that functions δ_{ZK} and δ_{PS} are bijective functions; that is, each of them is both injective (one-to-one) and surjective (onto). We use this property of the functions as inroad to formalize the rules of expression/presentation of SRE questions and their allocations to appropriate stakeholders in the next subsections.

3.4. Allocations of Kipling's interrogatives to express SRE questions

Following from the previous subsection, we present δ_{ZK} as a finite set of ordered pairs as follows:

© 2020 *Afr. J. Comp. & ICT* – All Rights Reserved
https://afrjcict.net

$$\delta_{ZK} = \begin{Bmatrix} (data, what), (functionality, how), \\ (network, where), (people, who), \\ (time, when), (motivation, why) \end{Bmatrix}$$

For every ordered pair (z, k) in δ_{ZK}, z is a Zachman's abstraction and k is the specific Kipling's interrogative assigned to express/present all SRE questions related to z. In defining this relation, we take cue from ZEF, which already has a similar relation, but we are using it for a purpose, which is slightly more encompassing than that of ZEF. While ZEF used the relation to elaborate on the coverage of its abstractions as well as argue their (the abstractions) completeness to capture the different aspects of an enterprise, we, in addition to these, use the relation to suggest the interrogative to be used for expressing questions on each abstractions to enhance clarity and mitigate ambiguity.

The objective of this is to ensure that both the interviewer and the interviewee are clear about the context of the question being asked. Since all relevant contexts are considerably captured with the six abstractions, the framework we propose seeks to create an understanding between the two parties through the presentation of questions such that the leading interrogative explicitly gives the context or the aspect of the software being referred to.

Recall that function $\delta_{QZ}: Q \longrightarrow Z$ matches every question $q \in Q$ to a specific abstraction $z \in Z$. If we agree with Zachman's claim that the set Z completely covers all aspects of interest in an enterprise [46, 47], consequent upon the acclaimed reputation of Kipling's set of interrogatives (K) for covering all information required about a news story [16], then a non-surjective function δ_{QZ}maybe a signal that set Q is incomplete. That is, when the RE maps his set Q of SRE questions for a given project onto Z, if there exists a particular $z \in Z$ which is not an image of any question $q \in Q$, then abstraction z is not addressed in Q and the consequence may be that the resulting requirement specification may be incomplete. However, we will need a bit more research to ascertain whether or not a surjective δ_{QZ} is sufficient to claim completeness of set Q.

From the foregoing, a composite function that feeds the output of δ_{QZ} into δ_{ZK} will allocate the appropriate Kipling's interrogative to express a given SRE question. Let us represent the function as $\delta_{QK}: Q \longrightarrow K$; then for every SRE question, i.e., $\forall q \in Q$, $\delta_{QK}(q) = \delta_{ZK}(\delta_{QZ}(q))$. For example, $(\delta_{QZ}(q) = motivation) \Longrightarrow (\delta_{QK}(q) = \delta_{ZK}(motivation) = why)$. Consequently, functions δ_{QZ} and δ_{ZK} work in tandem to classify a given SRE question into a category based on Zachman's abstractions and allocate a Kipling's interrogative to express it.

Table 6 presents a highlight of how this mechanism may be used to classify SRE questions from the SMS project and assign them the appropriate interrogatives. From the table, we can have an insight to how a question is classified into appropriate interrogative term. If a question is not asked with the best interrogative term, it could lead to misunderstanding/ambiguity in requirement specification. For example, a question to elicit the motivation for certain functionalities can best be asked with the *why* interrogative for quick understanding by the interviewee. Similarly, the *how* interrogative is used to elicit the stakeholders' way of getting things done.

3.5. Assignments of SRE Perspectives to Stakeholders

Similarly, δ_{PS} is a finite set of ordered pairs:

$$\delta_{PS} = \{(business\_model, owner),$$

$$(contextual\_model, domain\_expert), (usage, user),$$
$$(development\_life\_cycle, developer)\}$$

For every ordered pair (p, s) in δ_{PS}, s is a (group of) stakeholder(s) that is considered most suitable to answer SRE questions from p perspective of the project. Similar to function in Section 3.3, we take lessons from Zachman's pairing of business participants with enterprise perspectives to define this function. We recall from Section 2.1 that ZEF maps a (group of) participant(s) to the business perspective from which the participant(s) must describe the elements of the enterprise for the different abstractions. Similarly, function δ_{PS} in the framework we propose formally specifies the group of stakeholders that is most suitable to answer questions from a specific SRE perspective.

Recall also that function $\delta_{QP}: Q \longrightarrow P$ matches every question $q \in Q$ to a specific SRE perspective $p \in P$. Therefore, we can also define a composite function $\delta_{QS}: Q \longrightarrow S$, which passes the outputs of δ_{QP} into δ_{PS} to assign SRE questions to suitable stakeholders. That

© 2020 *Afr. J. Comp. & ICT* – All Rights Reserved
https://afrjcict.net

is, given a SRE question $q \in Q$, $\delta_{QS}(q) = \delta_{PS}(\delta_{QP}(q))$. For example, if $\delta_{QP}(q) = usage$, then $\delta_{QS}(q) = \delta_{PS}(usage) = user$. Therefore, we can also define a composite function $\delta_{QS}: Q \longrightarrow S$, which passes the outputs of δ_{QP} into δ_{PS} to assign SRE questions to suitable stakeholders. That is, given a SRE question $q \in Q$, $\delta_{QS}(q) = \delta_{PS}(\delta_{QP}(q))$. For example, if $\delta_{QP}(q) = usage$, then $\delta_{QS}(q) = \delta_{PS}(usage) = use$.

We show, in Table 7, how we apply this mechanism to assign the SRE questions from the SMS project to stakeholders.

Arguably, a non-surjective function δ_{QS} is an indication of one or more of the following:

- One or more of the key stakeholders is/are not being taken along in the elicitation process.
- Some critical questions - which could have been assigned to the neglected stakeholder(s) - are missing from set Q.
- Some questions have been placed in the wrong perspectives thereby leading to the assignment of questions meant for the neglected stakeholders to the wrong groups.

Finally, after running all the functions in Equation 1 on a given set Q of unclassified SRE questions, the result will be a set of ordered triples $CQ = \{(q, ki, sh) | q \in Q, ki \in K, sh \in S\}$; ki denotes the Kipling's interrogative assigned to express q and sh is the stakeholder category to which q is addressed.

3.6. Architecture and Workflow of the ReQueClass Framework

Table 8 combines Tables 6 and 7 to produce a six-by-four matrix architecture - similar to ZEF - for ReQueClass. The matrix has the Zachman's abstractions and their corresponding Kipling's interrogatives as column labels while the stakeholders and their corresponding perspectives form the row labels. Hence, the cells of the matrix - represented by the grayed cells in Table 8 - accommodate SRE questions about specific abstractions (expressed with specific Kipling's interrogatives) and perspectives (with indications of the most suitable stakeholders to address them). For example, any question in cell $Q_{1,1}$ is about the data content from business model perspective and it is expressed with the "what" interrogative and addressed to "owner". Similarly, a question in cell $Q_{3,6}$ is about the motivation for the software project from the usage perspective and it is expressed with the "why" interrogative and addressed to the prospective "user".

Figure 1 presents the group of algorithms that describe the workflow of the ReQueClass framework. To begin with, we define a set of four global types Z, P, K and S to denote the finite sets of Zachman's abstractions, SRE perspectives, Kipling's interrogatives and Stakeholder categories respectively. These global types are used to specify the types of variables in the framework as used in the algorithms.
Procedure ReQueClass describes the main workflow of the activities of the framework. The set CQ in line 2 - which is empty by default - is a set of ordered triples (q, ki, sh) such that q, ki and sh denotes a SRE question, its assigned Kipling's interrogative and allocated stakeholder respectively. The main activities occur in the "for loop" in lines 7-13. It describes how the RE iterates over each question in Q and maps it to: 1) a Zachman's enterprise abstraction (respectively to compute an appropriate Kipling's interrogative to express it) and 2) a SRE perspective (respectively to identify the most suitable stakeholders to answer the question). Lines 10 and 11 invoke functions δ_{ZK} and δ_{PS} respectively to automate the computation of Kipling's interrogatives and suitable stakeholders. The comments on various steps provide further explanation of the algorithms.

IV. IMPLEMENTATION AND DEMONSTRATION

Figure 2 presents highlights of a web-based application, which we have built to concretize the abstract concepts presented in the previous section. The top of Figure 2 shows the two steps to concretize the classification and allocation of elicitation interview questions specified in lines 8-12 of procedure REQUECLASS in Figure 1.

As shown in the top-left corner of Figure 2, the user starts by selecting the system abstraction (i.e., Zachman's Category) addressed by the question. The system automatically maps the selected abstraction to its Kipling's interrogative and proposes the suitable interrogative with which to express the question as shown in the top-right corner of the figure.

© 2020 *Afr. J. Comp. & ICT* – All Rights Reserved
https://afrjcict.net

In the case of the sample captured in the screenshot, the system proposed the interrogative "How" for a "Functionality" abstraction.

After entering the question, the RE selects the domain perspective of the question (before submitting) to enable the system to allocate the question to the appropriate stakeholder. In the case of the screenshot in top-right corner of Figure 2, the Usage perspective selected will be mapped to the end users category of stakeholders.

While the top of Figure 2 are excerpts from the RE's interfaces, the centre and bottom parts of the figure are excerpts from the stakeholders' interfaces with the system. The screenshots at the centre show the interface for a stakeholder to select a requirement elicitation project to participate in (centre-left) and indicate his/her designation (centre-right) to answer his/her allocated questions.

Finally, the bottom part of the figure contains screenshots of interfaces through which the stakeholders get to view and answer their allocated questions. The bottom-left and bottom-right screenshots show the questions allocated to the Domain Expert and End users respectively.

V. SUMMARY, CONCLUSION AND FUTURE WORKS

This paper has presented the ReQueClass, a framework that adopts techniques from the Zachman's Enterprise Architecture to reduce the communication gaps between the RE and other stakeholders during requirements elicitation interviews. ReQueClass offers the mechanism to establish a ground for common understanding among the various stakeholders based on the choice of interrogatives for presenting elicitation questions on the different aspects of the system to be developed. The choice of the 5W1H interrogatives to present questions about the various aspects of the system serves to elicit a complete view of the system. Finally, we acknowledge that for an elicitation interview to be effective, all questions must be posed to the appropriate persons in order to get useful responses. Hence, framework provides the mechanism to automate the allocation of elicitation questions to stakeholders based on the system perspective chosen by the interviewer.

In future work, the current implementation of the framework will be extended to generate a standard requirements specification document from the interview questions and the corresponding answers provided by the stakeholder to whom they are allocated.

REFERENCES

[1] A. Ferrari, P. Spoletini, and S. Gnesi, "Ambiguity and tacit knowledge in requirements elicitation interviews," *Requirements Engineering*, journal article vol. 21, no. 3, pp. 333-355, 2016, doi: 10.1007/s00766-016-0249-3.

[2] M. Mohanan and P. Samuel, "Software requirement elicitation using natural language processing," in *Innovations in Bio-Inspired Computing and Applications*: Springer, 2016, pp. 197-208.

[3] M. A. Jabar, R. Ahmadi, M. Y. Shafazand, A. A. A. Ghani, and F. Sidi, "An automated method for requirement determination and structuring based on 5W1H elements," in *2013 IEEE 4th Control and System Graduate Research Colloquium*, 2013: IEEE, pp. 32-37.

[4] P. A. Laplante, *Requirements engineering for software and systems*. Auerbach Publications, 2017.

[5] L. R. Wong, D. S. Mauricio, and G. D. Rodriguez, "A systematic literature review about software requirements elicitation," *J Eng Sci Technol*, vol. 12, no. 2, pp. 296-317, 2017.

[6] D. Zowghi and C. Coulin, "Requirements Elicitation: A Survey of Techniques, Approaches, and Tools," in *Engineering and Managing Software Requirements*, A. Aurum and C. Wohlin Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 19-46.

[7] W. R. Friedrich and J. A. Van Der Poll, "Towards a methodology to elicit tacit domain knowledge from users," *Interdisciplinary journal of information, knowledge, and management*, vol. 2, no. 1, pp. 179-193, 2007.

[8] I. Hadar, P. Soffer, and K. Kenzi, "The role of domain knowledge in requirements elicitation via interviews: an exploratory study," *Requirements Engineering*, vol. 19, no. 2, pp. 143-159, 2014/06/01 2014, doi: 10.1007/s00766-012-0163-2.

© 2020 *Afr. J. Comp. & ICT* – All Rights Reserved
https://afrjcict.net

[9] Y. Lee and W. Zhao, "An ontology-based approach for domain requirements elicitation and analysis," in *First International Multi-Symposiums on Computer and Computational Sciences (IMSCCS'06)*, 2006, vol. 2: IEEE, pp. 364-371.

[10] A. Mishra, A. Awal, and J. Elijah, "Automation of Requirement Analysis in Software Engineering," *International Journal on Recent and Innovation Trends in Computing and Communication*, vol. 5, no. 5, pp. 1173-1188, 2017.

[11] A. Sutcliffe and P. Sawyer, "Requirements elicitation: Towards the unknown unknowns," in *2013 21st IEEE International Requirements Engineering Conference (RE)*, 2013: IEEE, pp. 92-104.

[12] F. Egbokhare, "Causes of software/information technology project failures in nigerian software development organizations," *African Journal of Computing and ICT*, vol. 7, no. 2, pp. 107-110, 2014.

[13] A. Distanont, H. Haapasalo, M. Vaananen, and J. Lehto, "The engagement between knowledge transfer and requirements engineering," *International Journal of Management, Knowledge and Learning*, vol. 1, no. 2, pp. 131-156, 2012.

[14] V. Langenfeld, A. Post, and A. Podelski, "Requirements Defects over a Project Lifetime: An Empirical Analysis of Defect Data from a 5-Year Automotive Project at Bosch," in *Requirements Engineering: Foundation for Software Quality*, Cham, M. Daneva and O. Pastor, Eds., 2016// 2016: Springer International Publishing, pp. 145-160.

[15] I. L. Margarido, J. P. Faria, R. M. Vidal, and M. Vieira, "Classification of defect types in requirements specifications: Literature review, proposal and assessment," in *6th Iberian Conference on Information Systems and Technologies (CISTI 2011)*, 15-18 June 2011 2011, pp. 1-6.

[16] X. Chen and Z. Jin, "Capturing requirements from expected interactions between software and its interactive environment: an ontology based approach," *International Journal of Software Engineering and Knowledge Engineering*, vol. 26, no. 01, pp. 15-39, 2016.

[17] C. Chepken, "Importance of Participant Characteristics in Software Systems Design In the Informal Sector," *African Journal of Computing and ICT*, vol. 7, no. 2, p. 2014, 2014.

[18] B. Nuseibeh and S. Easterbrook, "Requirements engineering: a roadmap," presented at the Proceedings of the Conference on The Future of Software Engineering, Limerick, Ireland, 2000.

[19] D. M. Fernández *et al.*, "Naming the pain in requirements engineering," *Empirical software engineering*, vol. 22, no. 5, pp. 2298-2338, 2017.

[20] J. Zachman, *The zachman framework for enterprise architecture*. Zachman Framework Associates Virginia, 2006.

[21] R. Kipling, *Just so stories for little children*. OUP Oxford, 1998.

[22] F. Carmagnola, "The five Ws for user model interoperability," in *Ubiquitous User Modeling*, 2008: D. Heckmann et al., pp. 30-36.

[23] A. P. Oranje, N. Al-Mutairi, and T. Shwayder, "Controversies in Diagnosis and Treatment," in *Practical Pediatruc Dermatology*, A. P. Oranje, N. Al-Mutairi, and T. Shwayder Eds. Switzerland: Springer, 2016, pp. 3-5.

[24] C. M. Pereira and P. Sousa, "A method to define an Enterprise Architecture using the Zachman Framework," presented at the Proceedings of the 2004 ACM symposium on Applied computing, Nicosia, Cyprus, 2004.

[25] Z. S. H. Abad, V. Gervasi, D. Zowghi, and K. Barker, "ELICA: An Automated Tool for Dynamic Extraction of Requirements Relevant Information," in *2018 5th International Workshop on Artificial Intelligence for Requirements Engineering (AIRE)*, 21-21 Aug. 2018 2018, pp. 8-14, doi: 10.1109/AIRE.2018.00007.

[26] Z. S. H. Abad, M. Rahman, A. Cheema, V. Gervasi, D. Zowghi, and K. Barker, "Dynamic Visual Analytics for Elicitation Meetings with ELICA," in *2018 IEEE 26th International Requirements Engineering Conference (RE)*, 20-24 Aug. 2018 2018, pp. 492-493, doi: 10.1109/RE.2018.00068.

[27] C. Allauzen, M. Riley, J. Schalkwyk, W. Skut, and M. Mohri, "OpenFst: A general and efficient weighted finite-state transducer library," in *International Conference on Implementation and Application of Automata*, 2007: Springer, pp. 11-23.

© 2020 *Afr. J. Comp. & ICT* – All Rights Reserved
https://afrjcict.net

[28] M. Mohri, F. Pereira, and M. Riley, "Speech recognition with weighted finite-state transducers," in *Springer Handbook of Speech Processing*: Springer, 2008, pp. 559-584.

[29] M. Noaeen, Z. S. H. Abad, and B. H. Far, "Let's Hear it from RETTA: A Requirements Elicitation Tool for TrAffic Management Systems," in *2017 IEEE 25th International Requirements Engineering Conference (RE)*, 4-8 Sept. 2017 2017, pp. 450-451, doi: 10.1109/RE.2017.78.

[30] M. Hosseini, K. T. Phalp, J. Taylor, and R. Ali, "Towards Crowdsourcing for Requirements Engineering," presented at the The 20th International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ 2014) - Empirical Track, Essen, Germany, 2014. [Online]. Available: http://eprints.bournemouth.ac.uk/21904/.

[31] V. A. Shekhovtsov and H. C. Mayr, "Towards managing understandability of quality-related information in software development processes," in *International Conference on Computational Science and Its Applications*, 2014: Springer, pp. 572-585.

[32] V. A. Shekhovtsov and H. C. Mayr, "View harmonization in software processes: from the idea to quase," *Informatik 2016*, 2016.

[33] V. A. Shekhovtsov, H. C. Mayr, S. Ianushkevych, M. Kucko, V. Lubenskyi, and S. Strell, "Implementing tool support for effective stakeholder communication in software development–a project report," *Ausgewählte Beiträge zur Anwenderkonferenz für Softwarequalität Test und Innovation-ASQT*, pp. 45-58, 2014.

[34] V. A. Shekhovtsov, H. C. Mayr, and C. Kop, "Chapter 3 - Harmonizing the Quality View of Stakeholders," in *Relating System Quality and Software Architecture*, I. Mistrik, R. Bahsoon, P. Eeles, R. Roshandel, and M. Stal Eds. Boston: Morgan Kaufmann, 2014, pp. 41-73.

[35] V. A. Shekhovtsov, H. C. Mayr, and V. Lubenskyi, "QuASE: A Tool Supported Approach to Facilitating Quality-Related Communication in Software Development," in *2014 9th International Conference on the Quality of Information and Communications Technology*, 23-26 Sept. 2014 2014, pp. 162-165, doi: 10.1109/QUATIC.2014.28.

[36] A. Kleppe, *Software language engineering: creating domain-specific languages using metamodels*. Pearson Education, 2008.

[37] B. Selic, "A systematic approach to domain-specific language design using UML," in *10th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC'07)*, 2007: IEEE, pp. 2-9.

[38] T. H. Al Balushi, P. R. F. Sampaio, D. Dabhi, and P. Loucopoulos, "ElicitO: A Quality Ontology-Guided NFR Elicitation Tool," in *International Working Conference on Requirements Engineering: Foundation for Software Quality*, 2007: Springer, pp. 306-319.

[39] T. H. Al Balushi, P. R. F. Sampaio, and P. Loucopoulos, "Eliciting and prioritizing quality requirements supported by ontologies: a case study using the ElicitO framework and tool," *Expert Systems*, vol. 30, no. 2, pp. 129-151, 2013, doi: 10.1111/j.1468-0394.2012.00625.x.

[40] V. Basili, P. Donzelli, and S. Asgari, "A unified model of dependability: capturing dependability in context," *IEEE Software*, vol. 21, no. 6, pp. 19-25, 2004, doi: 10.1109/MS.2004.30.

[41] P. Donzelli and V. Basili, "A practical framework for eliciting and modeling system dependability requirements: Experience from the NASA high dependability computing project," *Journal of Systems and Software*, vol. 79, no. 1, pp. 107-119, 2006/01/01/ 2006, doi: https://doi.org/10.1016/j.jss.2005.03.011.

[42] S. Abdulkadir and H. O. Aliyu, "ReQueclass: A Framework for Classifying Requirement Elicitation Questions Based on Kipling's Technique and Zachman's Enterprise Framework-A Guide for Software Requirement Engineers," *i-Manager's Journal on Software Engineering*, vol. 13, no. 2, p. 9, 2018.

[43] F. Anwar and R. Razali, "A practical guide to requirements elicitation techniques selection-an empirical study," *Middle-East Journal of Scientific Research*, vol. 11, no. 8, pp. 1059-1067, 2012.

[44] N. Mulla and S. Girase, "A new approach to requirement elicitation based on stakeholder recommendation and collaborative filtering," *International Journal of Software*

© 2020 *Afr. J. Comp. & ICT* – All Rights Reserved
https://afrjcict.net

Engineering & Applications, vol. 3, no. 3, p. 51, 2012.

[45] H. Saiedian and R. Dale, "Requirements engineering: making the connection between the software developer and customer," *Information and Software Technology,* vol. 42, no. 6, pp. 419-428, 2000/04/15/ 2000, doi: https://doi.org/10.1016/S0950-5849(99)00101-9.

[46] K. M. de Oliveira, F. Zlot, A. R. Rocha, G. H. Travassos, C. Galotta, and C. S. de Menezes, "Domain-oriented software development environment," *Journal of Systems and Software,* vol. 72, no. 2, pp. 145-161, 2004.

[47] I. Sommerville and P. Sawyer, "Viewpoints: principles, problems and a practical approach to requirements engineering," *Annals of software engineering,* vol. 3, no. 1, pp. 101-130, 1997.

© 2020 *Afr. J. Comp. & ICT* – All Rights Reserved

Table 1. An Overview of the Zachman's Framework for Enterprise Architecture

	Data (What?)	**Functionality** (How?)	**Network** (Where?)	**People** (Who?)	**Time** (When?)	**Motivation/ Business Goal** (Why?)
Planner (*Business Scope/ Contextual Perspective*)				$C_{1,4}$		
Owner (*Conceptual/Business Model Perspective*)				$C_{2,4}$		
Designer (*Logical/System Model Perspective*)				$C_{3,4}$		
Builder (*Physical/Technology Model Perspective*)				$C_{4,4}$		
Subcontractor(*Detailed representation*)				$C_{5,4}$		
Functioning Enterprise				$C_{6,4}$		

Table 2. An overview of abstractions in ZEF

ZEF Column	**Description**
Data (What?)	**What** is the data, or are the things, important to the enterprise from the perspective of each of the participants.
Function (How?)	What are the processes in the enterprise and **How** are they performed to realize the operational/functional goal(s) of the enterprise from the perspective of each of the participants.
Network (Where?)	**Where** are the locations from which the businesses of the enterprise operate and what are the linkages between them. i.e., what is the geographical distribution of the enterprise's activities as perceived by the participants
People (Who?)	**Who** are the people/organizations relevant to the enterprise, their respective roles and the relationships between them as viewed from the perspective of each participant.
Time (When?)	What are the events or cycles significant to the enterprise and **when** are they occurring with respect to one another.
Motivation (Why?)	What are the strategic business goals of the enterprise and how do they translate into specific ends and means. Apparently, this is to provide justifications for the (i.e., **why**) need for the other abstractions in the same row of the framework.

© 2020 *Afr. J. Comp. & ICT* – All Rights Reserved
https://afrjcict.net

Table 3 Enterprise participant and perspectives in ZEF

Participant	Perspective	Comments
Planner	Business scope/ context	The planner defines the scope of an enterprise and places it in an environmental context.
Owner	Business model/ concept	The owner defines the concepts and the expected deliverables of the enterprise.
Designer	Logical/ system model	The designer defines the logical/system model of the enterprise following from the owner's business concept in such a way as to support the deliverables set by the owner.
Builder	Physical/ technology model	The builder oversees the concrete/technological implementation of the system model. This may involve the production and coupling of some components of the enterprise's products.
Subcontractor	Detailed representation	The builder may sometimes need subcontractors to realize some out-of-context components of the enterprise's deliverables.

Table 4. Mapping of SRE stakeholders to perspectives

Stakeholder Category	Perspectives	Descriptions/Comments
Owner/Sponsor	Business model	The owner/sponsor refers to the customers or clients who pay for the software project. They have sound understanding of the business model.
Domain Expert	Contextual model	Domain Expert refers to the personnel who have sound understanding of the domain of application of the software project plus good (or at least average) knowledge of software development process. They have the necessary experience to put the requirements in the context of the application domain with clarity.
Developers	Development life cycle	Developers comprise the software development team that will build and maintain the system. They develop the software models and plan the development lifecycle.
User	Usage	Users are the people who interact with the system to get their work done. Users provide a better view of the functioning system as they are familiar with the operations of the system.

© 2020 *Afr. J. Comp. & ICT* – All Rights Reserved
https://afrjcict.net

Table 5 SRE Stakeholders in the SMS project

Stakeholder Categories	**Concrete Stakeholders**
Owner/Sponsor	Management Board of CoE
Domain Expert	ICT Personel of CoE
Developers	JDLab Team
User	Students, Departmental Administrative Officers, Bursary Officers & Registry Students' Affairs Officers of CoE

Table 6. Using specific Kipling's interrogatives to express SRE questions of different abstractions

Abstraction	**Kipling's Interrogative**	**Examples**
Data	What	a. What data need to be migrated from the present system? b. What technical platform do you use today to manage student information? c. What documents and/information should a student provide for registration? d. What are the categories of students that will use the system? e. What information would the various users need about a student? f. What information do you use to uniquely identify a student? g. What is/are the determinant(s) of students' academic statuses? h. What computing platform would you like to deploy the proposed system?
Functionality	How	a. How do students do registrations and payments at present? b. How do you want to receive payments of fees with the proposed system? c. How do you store and retrieve the required students' information presently? d. How do you determine the courses and/or exams to be registered by each student e. How do you determine the fees to be paid by each student? f. How do you compute/determine students' academic statuses? g. How do students get notifications of their academic statuses at present?
Network	Where	a. Where will the system be deployed for usage? b. Where the users should be located physically to use the system? c. Where do students do registrations and payments at present? d. Where do you store and retrieve the required students' information presently? e. Where do students get their academic statuses at present?
People	Who	a. Who are the intended end users of the proposed system? b. Who will manage the system support and maintenance? c. Who computes students' academic statuses? d. Who will grant access to students to make registrations and/or payments on the proposed system?

© 2020 *Afr. J. Comp. & ICT* – All Rights Reserved
https://afrjcict.net

		e. Who will create user accounts on the proposed system? f. Who determines the courses and/or exams to be registered by each student? g. Who determines the fees to be paid by each student
Time	When	a. When do students get their academic statuses? b. When do you determine the courses and/or exams to be registered by each student c. When do you determine the fees to be paid by each student d. When do you open/close the window for registrations and payments? e. When do you want the system to be launched? f. When do you require system update?
Motivation	Why	a. Why do you prefer the chosen computing platform for deployment?

Table 7. Allocation of SRE questions to stakeholders based on SRE perspectives

Stakeholders	**SRE Perspective**	**Examples**
Owner	Business model	a. Who are the intended end users of the proposed system? b. What are the categories of students that will use the system? c. When do you determine the fees to be paid by each student?
Domain expert	Contextual model	a. What data need to be migrated from the present system? b. What information do you use to uniquely identify a student? c. How do you determine the courses and/or exams to be registered by each student?
User	Usage	a. How do you perform registration and payment at present? b. How do you store and retrieve the required students' information presently? c. How do students get notifications of their academic statuses at present?
Developer	Development life cycle	a. What computing platform would you like to deploy the proposed system? b. Where will the system be deployed for usage?

Table 8. ReQueClass Framework Architecture

Abstractions/*Kipling's Interrogatives* / Stakeholders (*Perspectives*)	Data (*What*?)	Functionality (*How*?)	Network (*Where*?)	People (*Who*?)	Time (*When*?)	Motivation/ Business Goal (*Why*?)
Owner (*Business model*) **Domain expert** (*Contextual model*) **User** (*Usage*) **Developer** (*life cycle*)	$Q_{1,1}$					$Q_{3,6}$

© 2020 *Afr. J. Comp. & ICT* – All Rights Reserved
https://afrjcict.net

Global Types

$Z = \{data, functionality, network, people, time, motivation\}$
$P = \{business\_model, contextual\_model, life\ cycle, usage\}$
$K = \{what, who, where, when, why, how\}$
$S = \{owner, domain\_expert, user, developer\}$

---

procedure REQUECLASS(Q = {set of unclassified SRE questions})
    $CQ = \{\}$ ▷ set of classified and assigned SRE questions
    Let $abst : Z \leftarrow$ **null** ▷ variable assigned Zachman's abstractions
    Let $pers : P \leftarrow$ **null** ▷ variable to hold assigned SRE perspectives
    Let $ki : K \leftarrow$ **null** ▷ variable to hold assigned Kipling's interrogative
    Let $sh : S \leftarrow$ **null** ▷ variable to hold assigned stakeholder
    for $q \in Q$ **do**
        $abst \leftarrow z \bullet z \in Z$ ▷ match q with a Zachman's abstraction from Z
        $pers \leftarrow p \bullet p \in P$ ▷ match q with a SRE perspective from P
        $ki \leftarrow \delta_{ZK}(abst)$ ▷ compute Kipling's interrogative for q
        $sh \leftarrow \delta_{PS}(pers)$ ▷ compute the suitable stakeholder to answer q
        $CQ \leftarrow CQ \cup \{(q, ki, sh)\}$ ▷ add q and its parameters to set CQ
    end for
end procedure

---

function $\delta_{ZK}(z : Z) : K$ ▷ input of type Z; output of type K
    switch(z)
        case *data* :
            return *what* ▷ data ↦ what
        case *functionality* :
            return *how* ▷ functionality ↦ how
        case *network* :
            return *where* ▷ network ↦ where
        case *people* :
            return *who* ▷ people ↦ who
        case *time* :
            return *when* ▷ time ↦ when
        case *motivation* :
            return *why* ▷ motivation ↦ why
    end case
end function

---

function $\delta_{PS}(p : P) : S$ ▷ input of type P; output of type S
    switch(p)
        case *business_model* :
            return *owner* ▷ business_model ↦ owner
        case *contextual_model* :
            return *domain_expert*
        case *life cycle* :
            return *developer* ▷ network ↦ developer
        case *usage* :
            return *user* ▷ usage ↦ user
    end case
end function

Figure 1. Workflow of ReQueClass

© 2020 *Afr. J. Comp. & ICT* – All Rights Reserved
https://afrjcict.net

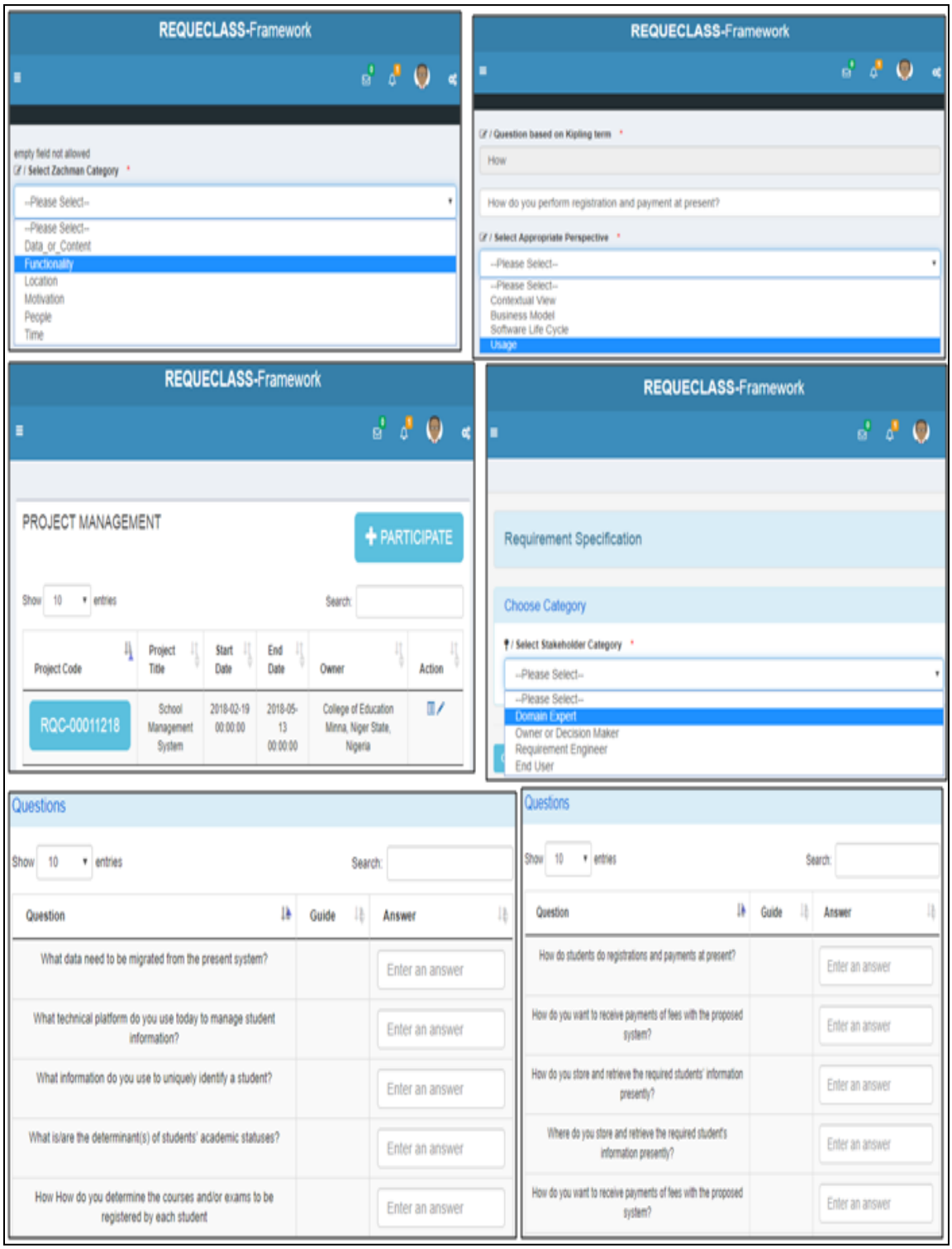

Figure 2 Excerpts from ReQueClass Implementation