Ontology Driven Requirement Specification

[1]Adeel Ahmed, Fabeha Maniar

[1,2]Shaheed Zulfiqar Ali Bhutto Institute of Science and Technology, Karachi Pakistan

[1]adeel.ahmed@szabist.edu.pk

Abstract—Requirement engineering (RE) process is an important step of software development lifecycle and it includes a variety of activities starting with requirement elicitation to requirement documentation. This form of engineering is the backbone of a quality product. This research focus on the attention for requirement engineers to understand the criticality of requirement specification and how a complete, consistent requirement can be extracted using an Ontology. It covers the intrinsic knowledge of the domain into machine readable easy format and requirement represented in Ontology serves as the formal representation of natural language context in a model. It is often the case in software development that requirement creep leads to an unpredictable stage in software lifecycle which becomes hard to handle. Many techniques to improved requirement specification has been tested but this paper address how user stories are represented in Ontology model by extracting the main classes of Role, Object and Action from the user and later building a model based on these core concepts along with property link for relating the three basic root classes for completion.

***Keywords*—**Requirement Specification, Ontology, User Stories, Requirement Engineering, Business Process Model.

I. REQUIREMENT ENGINEERING

A. Requirement:

Requirement is a bridge between the real world (a world where there are problems) and the software system and the requirements can be thought of as a translation of problems into a form that can be converted into software system. Hence requirements are a reflection of problems of real world that can be used by software engineers. In software world the stakeholder's needs are regarded as requirements which are to be converted to technical solutions.

B. Importance of requirement engineering:

Requirement specification acts as a contract between developer and the stakeholders and to note if the base line is not correct, the expectation of quality, furnished product is hard to achieve. The multiple phases of RE, each plays a vital role from requirement capture to management and analysis. This is the most important aspect of any software development as the entire lifecycle depends upon it however it is highly neglected.

If requirements are not stated correctly, it may lead to following problems:

a. One of the frequent consequences is budget and delivery issues. The team has to re work and put in all efforts from scratch if requirements are not correctly understood and stated. This implies heavy loss as the cost gets on higher side than originally expected.

b. Satisfaction level of the end users and customers is compromised. If the system is not delivering the intended functionality and do not serve the purpose then it is a direct customer impact. They may abandon the system or may not use it to its full potential.

c. One of the cases can be system unreliability where system has frequent crashes and user errors.

If the system continues in bad shape, cost associated with maintaining and system evolution are exorbitant. Post delivery cost becomes difficult to handle.

C. Difficulties to capture requirements:

One of the main reasons that can be attributed to difficulty of capturing requirements is the rapidly changing business environment. One requirement that is primary in some point of time becomes secondary or it may get clubbed with another requirement. The ever changing environment poses serious threats to requirement gathering and management. Organizational level changes in policies and procedures often leads to changes in system functionality and requirement changes are proposed based on each new policy which gets difficult to capture.

Multiple stakeholders with different goals and priorities are involved. Any system always or more frequently consist of multiple stakeholders and to capture the requirements becomes difficult during the requirement engineering activity. Consequently requirements are always a compromise in the stakeholder activity. Every stakeholder jumps in with their own requirement without considering the design and technical solution. The only goal for them is solution driven change and to capture each stakeholder's needs and requirement becomes extremely difficult.

There are cases where even the stakeholder is not able to define and explain the requirement and needed functionality in clear way. Consequently the vague expression of the requirements leaves the task to system engineers to complete the unfulfilled duty.

This factor is highly evident in every function/ department because of the fact that stakeholder's are not the system technical owners neither they are aware of the design solution. The only primary focus for them is the result and there have been cases where the system owner fails to explain the requirement due to lack of business process expertise or ability to express and document.

The system engineers may get it wrong because they are not the product owners and unaware of the business and process logic. Not every person can be the master of all jobs, product owners are not developers and vice versa. The job nature and duty is specific to each role and it is important to understand while specifying requirements. The product owner has to be the master of his role to avoid gaps and similarly the developer should be technically sound to translate the requirement in software solution.

Organizational politics and factors also contribute to make the requirement engineering activity a problem. It is important to imply the best method suitable for requirement elicitation and manage the overall RE process in organized manner to avoid requirement gaps. This stage of SDLC is usually taken lightly and is not given the due importance it has, which ultimately contributes to failure. It is considered that getting requirements is easy however most users fail miserably at this point.

Requirement capture and specification is taken as an ongoing task without understanding the implied need and importance of this SDLC stage because if the notion of customer will not be reflected in the final output the whole process and efforts are wasteful, thus it is hardest part to decide precisely what needs to be built. There is growing need to understand and adopt that it is not always important to register failure to technical aspect, it can be largely due to requirement specification and RE process gap [1].

D. *Requirement specification:*

A software requirement specification serves as the parent document and starting point to software development. It is a blueprint of stakeholder requirement prior to any product development that states the customer needs and expectations in natural language which is required in the software product. It serves as the parent document to the proceeding discussions and related documents that are part of the project stating the functions /capabilities and constraints expected of the final product. It should capture the requirement (functional, non functional, quality, business logics only) and not the design and technical solution. What is expected from requirement perspective should be available in the SRS but not how to achieve and the methodology.

E. *User Stories:*

Agile methodology has led to the adoption of user stories for requirement specification. These are small snippets from user perspective for the required functionality. These texts convey the message to software engineers the product functionality and deliverables expected. It is said that the user story should contain the following three aspects:

a. Card (text of the story): It represents the text of the story.

b. Conversation (Details of the story): The conversations are matured with project progress as understanding and more clarity is received.The actual value of the user story is developed as the project gradually gets to completion.

c. Confirmation (Conversation verification): It forms the acceptance criteria for the user story.

II. KNOWLEDGE REPRESENTATION AND ONTOLOGY

A. *Knowledge representation and semantic web:*

The world is composed of all knowledge and the representation of knowledge base in a form that can be used by computers to solve complex tasks is known as knowledge representation in artificial intelligence. With the advent of semantic web by W3C, the more information on the INTERNET was able to be captured in specific terms which were difficult before. Many terms were found to be related and it became a problem to define each term and its context. It is with the Semantic Web that the knowledge base representation chaos has been lessened. Ontologies are the core concept for knowledge representation and since requirements are also knowledge in context of software development, this research has identified Ontology as the means to requirement representation in form of model rather than natural language text.

B. Ontology:

A term originated from psychology but now being used in computer science as well, meaning from popular definition of Gruber as “explicit specification of a conceptualization”. Data model representing knowledge as a set of concepts within a domain and the relationships between these concepts. Data is everywhere so it becomes difficult to capture the relationship between data in various places. Ontology captures data in a way that makes relationships visible. Ontologies work well with text. The natural language used by human for specification and description of domain/process knowledge is correlated to Ontological representation using formal language. Managing knowledge and capturing relationships. It is an explicit formal specification of a shared conceptualization (conceptualization = model/form a model of a domain and inside the domain find relevant concepts and how concepts are related with each other, Explicit = meaning of concepts must be defined, formal = machine understandable read and interpreted correctly, shared = common because only then communication is possible).It uses the concept of subject predicate and object to define domain knowledge in Ontological form consisting of classes and relationships. Knowledge can be represented using the same form [2].

C. Requirement specification and ontology:

Requirements are process and business knowledge in software world. The stakeholders document the required specifications in natural language text and provide to project team for further working. Software engineers work on the text as per their interpretation and build a product for delivery. This is where the importance of requirement representation comes in play. It is highly possible that requirement document and specification style leads to misinterpretation and the end result is not as per expectations. It is natural that a certain word or sentence can be looked upon with different representation by many people. The same text meaning can be different for one person and same can be different for other person. It is highly important to represent domain knowledge in machine readable form for uniformity and consistency among terms. This is where ontological representation holds true. Requirements needs a consistent shape and view that holds true for every software engineer, it should not be possible to amend and interpret as per yourself therefore requirement representation through Ontological model plays an important role.

With the advent of semantic web it has become easier to apply Ontologies to knowledge representation and management. Diverse requirements and point of view can be captured using a model which can be used for better product delivery. One step to make requirement specification versatile and strong through a structured representation form. Natural language vagueness can fool the developers on what to develop, they might think that this is right way however when the stakeholder reviews it, rejects the same. Ontology help formalism of terms and expressions, representing domain knowledge at semantic level i.e. classes objects and relations. With this the users can express their needs easily in more structured format. Representing natural language expressions in structural form makes it easier to interpret, understand and apply. The words of language can lead to different meaning but a model interpretation is unique and helps to decrease multi meaning expressions. It is the expression of knowledge and concepts in an explicit form through model [1].

III. LITERATURE REVIEW

Requirement specification has always been the affected by incompleteness, inconsistency and ambiguity. Various researches with different approaches have been focused to specify requirements in Ontology. The consistency of any requirement can be judged with its traceability to the actual requirement text. The class model should be capable of tracing back to the actual requirement to ensure that testable and complete requirement has been specified. This process can be achieved by extracting a domain Ontology from requirements through text analysis. In the first step the concepts are extracted from the SRS from which a glossary or model is attached and finally the class model. The technique applied to a case study and results were tabulated and it helps software engineers with to use knowledge stored in domain Ontology’s which is generic when written in form of concepts [2].

Ontology has the power to drive the requirement specification activity and address its main problems that becomes the reason for project failures. The paper illustrates the way to check requirement specifications using Ontology and provides a comparative solution between traditional checking of requirements and through Ontology used method. The numbers of errors identified in the requirements were analyzed to determine the effectiveness of using Ontology method in requirement specification checking. The requirements are specified in sentences from which verbs and nouns extracted which were mapped to Ontology. The results showed that the group using the Ontology method was able to detect more errors in the specification whereas the group working freely was not able to detect the requirement specification errors. Again the benefit of using Ontology for requirement specification was shown [3].

Requirement specifications are generally written in natural language and it becomes difficult to transform these requirements into formal specifications. Even if the requirements are somehow transformed into forma specifications, the chances of errors and correct transformation exists which results in questioning the end software product completeness and quality. The paper proposed OWL Ontology to model domain knowledge and represents the requirements which are further reasoned with an OWL reasoner [4].

Ontology supports the RE process in multiple ways. It can check requirements in many ways to make sure they are complete and validate them for benefit of product delivery. For years the requirement specification has been considered as the corner stone of project failures. The complexity to deliver exactly as stated and perceived have becomes normal which ultimately affects the product. The author highlights the obvious advantages of Ontology use when dealing with requirements. Formalization of requirements specification and requirements properties verification can be achieved through Ontology. The domain knowledge representation helps to improve the perception for engineers where requirements are huge in number and to see each requirement with equal importance.

The Ontology model is reusable to similar problems where concepts and relationships can be utilized. [5] Ambiguity is common in requirement specifications. It makes the developer of the software product confused as the stated requirements are ambiguous. The traditional practice followed is to develop what is understood from the SRS natural language text however this often leads to ruined user experience. The stakeholder at the end confirms that he is not happy with the deliverable which puts all efforts in vain. This is one of the biggest challenges facing SDLC and RE process on how to curtail this problem.

Ontology provides the solution. SRS is the start of any project and to capture these requirements in formal representation is what ontology allows and supports. The input taken is the SRS which is converted to application Domain Ontology consisting of classes, relationships, concepts and axioms. The axioms confirms to the conversion of natural language specifications to formal machine readable form. These axioms are further verified using the semantic reasoner to deduce logical inferences from the axioms [6].

During the course of literature review, the Islamic Banking Ontology was studied and analyzed as an application of the ontology in the Banking domain. The paper discusses the concepts and terms to be applicable in Islamic banking and the related products.

The review covers broadly how Ontology supports RE phases and a detailed analysis has been presented by the author. The phases, modelling styles and the types of requirements supported by Ontology driven process. The paper captures the growing interest of Ontology usage in software development requirement engineering domain and presents a quantitative study on the 7 research questions taken into account. Scope (inclusion and exclusion criteria, assumptions, time factor, and research literature) is well defined and after a series of steps, 67 papers were considered for the SLR.

Based on main research question, several questions (included in 7) were identified to be part of SLR. DoRES Ontology study which is crisis situation based awareness Ontology designed to collect, organize, analyze and share critical information between individuals and humanitarian organizations. The author has followed the NeOn methodology scenario for the DoRES ontological development [7].

Human Cardiovascular histological Ontology presented in Journal of Biomedical Semantics, 2017. This paper also follows the NeOn methodology based on competency questions and emphasis ontology development with the help of conceptual models. At each stage, the author has developed tree structure for ease of understanding and development and in sections a whole structure of the cardio vascular system is built in conceptual form followed by the OWL model [8].

Hybrid model approach based on frame Ontology and production rules is another way of specifying correct, consistent and non ambiguous, traceable requirements. The approach presents two ways to categorize the requirements of an SRS and the hybrid model shows how to classify the requirements. To understand the software requirement specification, semantic technology can be used. It helps to decode the natural language semantics to formal specifications and ensures a robust software development process. Similar to search engine, the engine returns relevant details by excluding the unnecessary details. Same is the scenario with requirement specification.

It is important to extract meaningful requirements and prioritize them. Every detail or word is not important as written by the stakeholder. To assure meaningful context is extracted semantic technology is useful. Data cleaning to be applied to SRS and formation of a cleaned graph. The graph is transformed to a matrix which becomes the base for Ontology construction. The text in software requirement specification document can be annotated to allow software engineers to work on necessary requirements. Domain Ontology helps with stakeholder's requirement. Domain knowledge representation is always useful when stating requirements.

IV. METHODOLGY

A. Steps of model creation from user stories:

- Data Gathering
- Rewriting the user stories in object oriented style.
- Extraction of main words (Role, Object, and Action) from the object oriented user story.
- Develop the class (according to Root and Specific Words) in Ontology.
- Model Creation
- Validation

B. Data gathering:

The user stories for the core banking system has been collected, analysed and implemented in business process ontological model. After the data gathering step, user stories collected were analyzed and written in object oriented style. The expression of the story followed some basic elements and representation has the following details:

- User or persona [can be users or interacting systems]
- Action [represent the activity of the user]
- Object [entity on which action is done]
- Benefit [represent the business value]

These elements have been classified into object, role and action as per the research methodology. The Root Word for each type (role, object and action) is the parent class where the Specific Word is the sub classes. Root Word is the generic words found in the business process and Specific Word is the child of the parent. This hierarchy can go to depth of N levels depending on the depth of the business process Ontology to be shown and the requirements to be taken in account. Secondly there are two other concepts as well, [Perform object] and [Perform Action] makes the user story complete. Each role performs some actions on some object which completes the object oriented user story [9].

C. Main Word Extraction from User Stories:

Role
- Customer
- Employee

Object
Account
- Saving
- Current

Relationship level
- Pakistani
- Foreigner

Action
- Account Services
- Account inquiry
- Account IVR
- Account statement inquiry
- Account Opening request
- Account deposit
- Account withdrawal
- Account transfer
- Account balance listing
- Account transaction listing

View
- Nationality
- Date of birth
- Risk marked

V. MODEL IMPLEMENTATION

The classes and schema prepared with user stories in the data gathering step has been implemented in Protégé editor. It is free, open source editor for developing intelligent systems. The classes will represent the model designed to present user stories in an Ontological model. This editor has been developed by Stanford team and has the following services available:

- Vast active community support for users.
- W3C support for latest OWL and RDF specifications.
- Open source environment along with web version availability.

The Protégé OWL editor is an extension of Protégé that supports the web Ontology language [10].

A. Class Hierarchy in Protégé:

The class hierarchy implemented on Protégé Editor as per classes identified in Role, Object and Action is presented in figure 1 below:

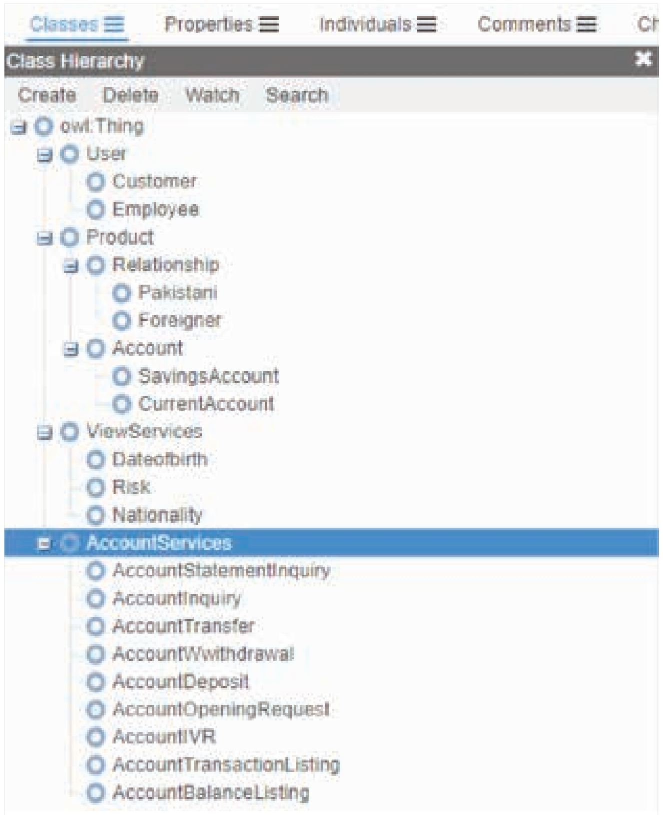
```
Classes    Properties    Individuals    Comments    Ch
Class Hierarchy
Create   Delete   Watch   Search
owl:Thing
  User
    Customer
    Employee
  Product
    Relationship
      Pakistani
      Foreigner
    Account
      SavingsAccount
      CurrentAccount
  ViewServices
    Dateofbirth
    Risk
    Nationality
  AccountServices
    AccountStatementInquiry
    AccountInquiry
    AccountTransfer
    AccountWwithdrawal
    AccountDeposit
    AccountOpeningRequest
    AccountIVR
    AccountTransactionListing
    AccountBalanceListing
```

Fig. (1). Class Hierarchy

B. Defining Property in Protégé:

As per the research methodology there are two kinds of properties identified:

- PerformObject (links Action Class to Object Class)
- PerformAction (Links Role Class to Action Class)

Customer [PerformAction] account balance listing [PerformObject] account
Customer [PerformAction] account transaction listing [PerformObject] account
Employee [PerformAction] account opening request [PerformObject] account
Customer [PerformAction] account withdrawal [PerformObject] account
Customer [PerformAction] account deposit [PerformObject] account
Customer [PerformAction] account transfer [PerformObject] account
Customer [PerformAction] account inquiry [PerformObject] account
Customer [PerformAction] account IVR [PerformObject] account
Customer [PerformAction] account statement inquiry [PerformObject] account
Employee [PerformAction] account statement inquiry [PerformObject] account
Employee [PerformAction] view nationality [PerformObject] relationship
Employee [PerformAction] view DOB [PerformObject] relationship
Employee [PerformAction] view risk [PerformObject] relationship

Each user story is now complete with class and property. The inclusion of both properties makes the user story and requirement specification complete [9].

Based on the user stories the following properties have been listed and classified accordingly in Object or Action. The same has then been implemented in Protégé editor as shown in figure 2.

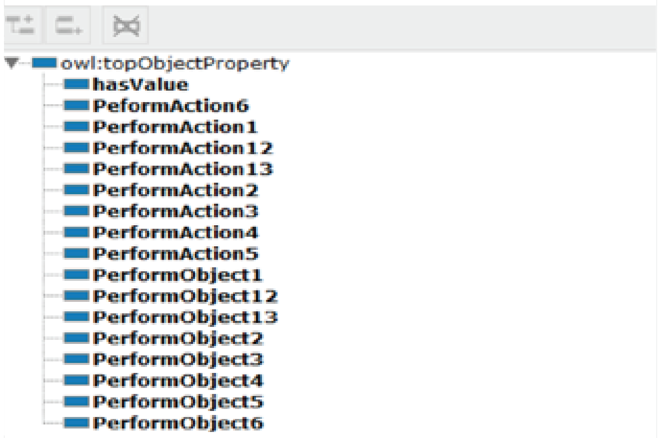
```
owl:topObjectProperty
  hasValue
  PeformAction6
  PerformAction1
  PerformAction12
  PerformAction13
  PerformAction2
  PerformAction3
  PerformAction4
  PerformAction5
  PerformObject1
  PerformObject12
  PerformObject13
  PerformObject2
  PerformObject3
  PerformObject4
  PerformObject5
  PerformObject6
```

Fig. (2). Property Hierarchy

Ontological Model that has been resulted from Class and Property is presented in figure 3.

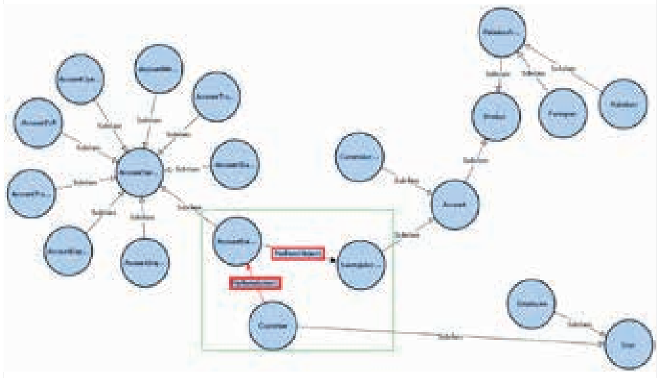

Fig. (3). Ontological Model

The second view from detailed Model created in Protégé is presented in figure 4. The class, subclass and property are shown in this figure.

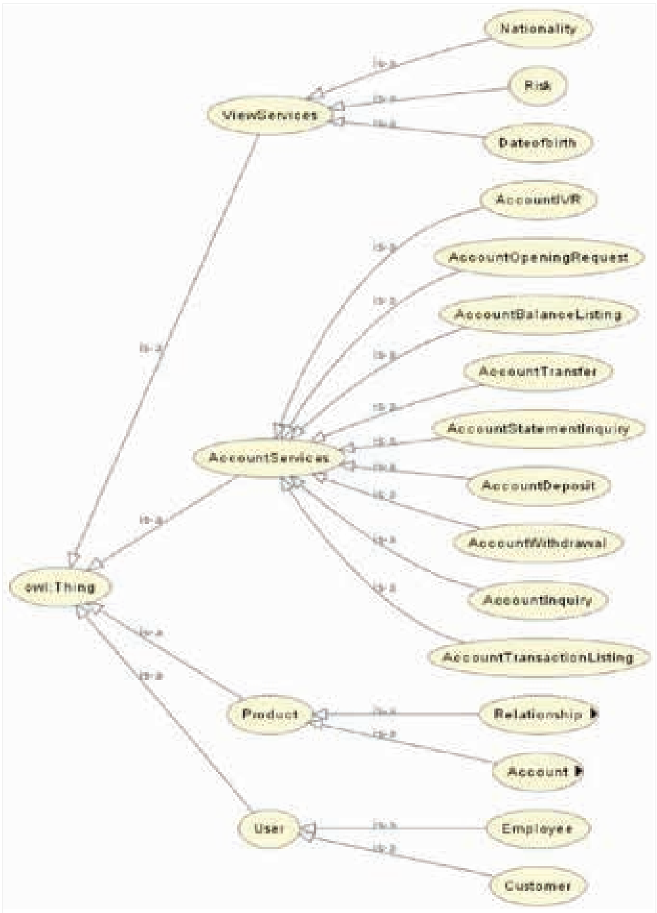

Fig. (4). Ontological Model II

VI. MODEL VALIDATION

The implemented banking ontology has been verified using the Protégé Reasoner 5.2.0. It validates the class consistency and design of the model. Since the user stories have been transformed into the Ontology model, the model validation has been done using the Protégé Reasoner.

VII. CONCLUSION AND FUTURE RESEARCH

This Study help to identify the importance and need of a better requirement specification process. The scope of study has been limited however as this can be implemented to the whole banking system keeping in mind all the products and services. The methodology can be adopted to extend this business Ontological model to all products and services and also make way to reuse for developing core banking systems in future. The research aims at giving a new way to authentication of requirement specification as user stories with ontology.

In this work, we have demonstrated an idea of how requirement specification authenticity and presentation can be improved through formal representation and demonstrated a way to better representation. This can be further extended as already stated to not only any core banking solution but also applicable to any domain as ontology helps to capture and represent domain knowledge in more expressive formal way. It is also possible to include new classes and property in the existing model and redesign accordingly. The addition of classes will not be hard to the existing design as per needs and requirements.

REFERENCES

[1] V. Castañeda, L. Ballejos, M. L. Caliusco and M. R. Galli, "The Use of Ontologies in Requirements Engineering," *Global Journal of Researches in Engineering*, vol. 10, no. 6, pp: 2-8, 2010.

[2] K. Iaroslaw, "Passing From Requirements Specification to Class Model Using Application Domain Ontology," In *Proceedings of the 2nd International Conference on Information Technology (ICIT 2010)*, 2010.

[3] D. V. Dzung and A. Ohnishi, "Evaluation of Ontology-based Checking of Software Requirements Specification," In *Proceedings of IEEE 37th Annual Computer Software and Applications Conference*, 2013.

[4] D. Sadoun, C. Dubois, Y. Ghamri-Doudane and B. Grau, "From Natural Language Requirements to Formal Specification using an Ontology," In *Proceedings of IEEE 25th International Conference on Tools with Artificial Intelligence*, 2013.

[5] T. Avdeenko and N. Pustovalova, "The Ontology-Based Approach to Support the Completeness and Consistency of the Requirements Specification," In *Proceedings of International Siberian Conference on Control and Communications (SIBCON)*, 2015.
DOI: 10.1109/SIBCON.2015.7147184

[6] A. S. Abdul-Latiff, H. Haron and M. Annamalai, "Characteristics and Development Criteria for Islamic Banking Ontology," In *Proceedings of Third International Conference on Information Retrieval and Knowledge Management (CAMP)*, 2016.
DOI: 10.1109/INFRKM.2016.7806350

[7] G. Burel, L. S. G. Piccolo, K. Meesters and H. Alani, "DoRES — A Three-tier Ontology for Modelling Crises in the Digital Age," In *Proceedings of the 14th ISCRAM Conference*, 2017.

[8] C. Mazo, L. Salazar, O. Corcho, M. Trujillo and E. Alegre, "A Histological Ontology of the Human Cardiovascular System," *Journal of Biomedical Semantics*, 2017.
DOI: 10.1186/s13326-017-0158-5

[9] C. Thamrongchote and W. Vatanawood, "Business Process Ontology for Defining User Story," In *Proceedings of IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS)*, 2016.
DOI: 10.1109/ICIS.2016.7550829

[10] M. Horridge. (2009). *A Practical Guide To Building OWL Ontologies Using Prot´eg´e 4 and CO-ODE Tools* [Online]. Available: https://tinyurl.com/ycm64lll