2015 2nd International Conference on Trustworthy Systems and Their Applications

Requirements Characteristics Verification Method and Tool based on Rules constructed form Software Component Relationships

Nattapon Phanthanithilerd
Department of Computer Engineering Faculty of Engineering, Chulalongkorn University
Bangkok Thailand
nattapon.pha@student.chula.ac.th

Nakornthip Prompoon
Department of Computer Engineering Faculty of Engineering, Chulalongkorn University
Bangkok Thailand
nakornthip.s@chula.ac.th

***Abstract*—Natural language sentences are normally used in specifying user requirements. However, the structure of natural language sentences may result in ambiguity or many possible interpretations of meaning. Therefore, the UML language has been used for requirements modeling as a communication medium between the user and the developer and among developers to reduce this problem. However, designing software models using UML from natural language sentences is complicated, and in some projects there are many user requirements, which may result in a designed model that does not completely cover the user requirements. Software engineers should therefore give importance to verification of requirements in the form of natural language sentences and the model to have good characteristics. Therefore, this research presents a method for verifying requirements in natural language sentence form, the software model and the model description to have good characteristics according to the IEEE 830 standard: Unambiguity, Consistency, and Traceability, using rules created from the components of the requirements sentences and UML models, as well as developing a tool for verification, which shows the results as the defects of the requirements obtained from verification for subsequent correction.**

Keywords*—*Software requirement characteristics, Software models, Requirements verification

I. Introduction

In software development, the step of identifying the user requirements is an important process. The user requirements are often in the form of natural language sentences, and the system model. The form of natural language sentences are classified into 2 categories: functional requirements, used to identify the services the system provides to various types of users; for example, an online sales system must display the product list to buyers, the sellers can add products into the online sales system, etc.; and structural requirements, used to identify the data that should be stored by the system; for example, the user must record the product details, consisting of product name, price and product details etc. However, the structure of natural language sentences may result in ambiguity or many possible interpretations of meaning, which makes the developed system do not correspond to user requirements. Therefore, UML is used to create a software model to reduce the aforementioned problems.

UML [6] is a language using diagrams to specify user requirements. UML models that are widely used are use case diagrams, for identifying functions for which the system provides services to different types of users, and class diagrams, used to identify the structure and relationships between the data in the system, as well as the data that should be stored by the system. Also, in order for the users to know the details of the model better, model descriptions are created. In the use case model, they are called use case descriptions, and in the class model they are called CRC cards. This research focuses on using the use case model, the class model and the descriptions in both models for verifying the requirement characteristics.

However, although developers use UML models to reduce ambiguity from natural language sentences, UML model design is still complicated, and in some projects, there are many user requirements, which as a result, the developer may design a model that does not completely cover the user requirements. The developer must verify the requirements in natural language sentence form to reduce ambiguity, and verify consistency, as well as traceability, between the requirements in natural language sentence form, the model, and the model description. Many research works have presented concepts for verifying requirement characteristics.

Research works [1-3] present concepts for methods to verify consistency for logical or temporal contradictions using formal methods. Research work [1] verifies consistency between the use case model and the class model. Research work [2] verifies consistency between the class model and the state machine diagram. Research work [3] verifies consistency between the class model and a Prolog script. However, the formal method cannot effectively be used with natural language sentences for verification with models.

Research work [4] presents the creation of rules used in verifying requirement characteristics: Unambiguity, Consistency, and Traceability. These rules are created from relationships between the components of requirements in natural language sentence form, the UML model, and the description. The relationships are presented in structural diagrams in Figure 1 and Figure 2. There are a total of 40 rules created for verification, which can be categorized according to the verified characteristic with numbers as shown in Table I.

TABLE I. Number of rules used in verifying requirement characteristics

Verification	**Number of rules**		
	Unambiguity	***Consistency***	***Traceability***
Within natural language sentences	4		
Within the model		11	4
Between functional natural language sentences and use case models		4	4
Between structural natural language sentences and use case models		9	2
Between structural model and functional model			2
Total	4	24	12

Therefore, in this research, a method for presenting a method for verification of requirements in natural language sentence form in both structural and functional requirements, models, namely the use case model and class model, as well as the descriptions of both models, by applying the rules created from the components of the requirements, to have good characteristics according to the IEEE 830 standard [5], in the topics of good characteristics of requirements documents, namely, unambiguity, consistency, and traceability. The automated tool is developed for verification of those requirements' characteristics.

This article will be separated into three sections as follows: Section 2 will explain the related theory, Section 3 will explain the research process and examples of verification and the last section will explain the conclusion from the research and approaches for future research.

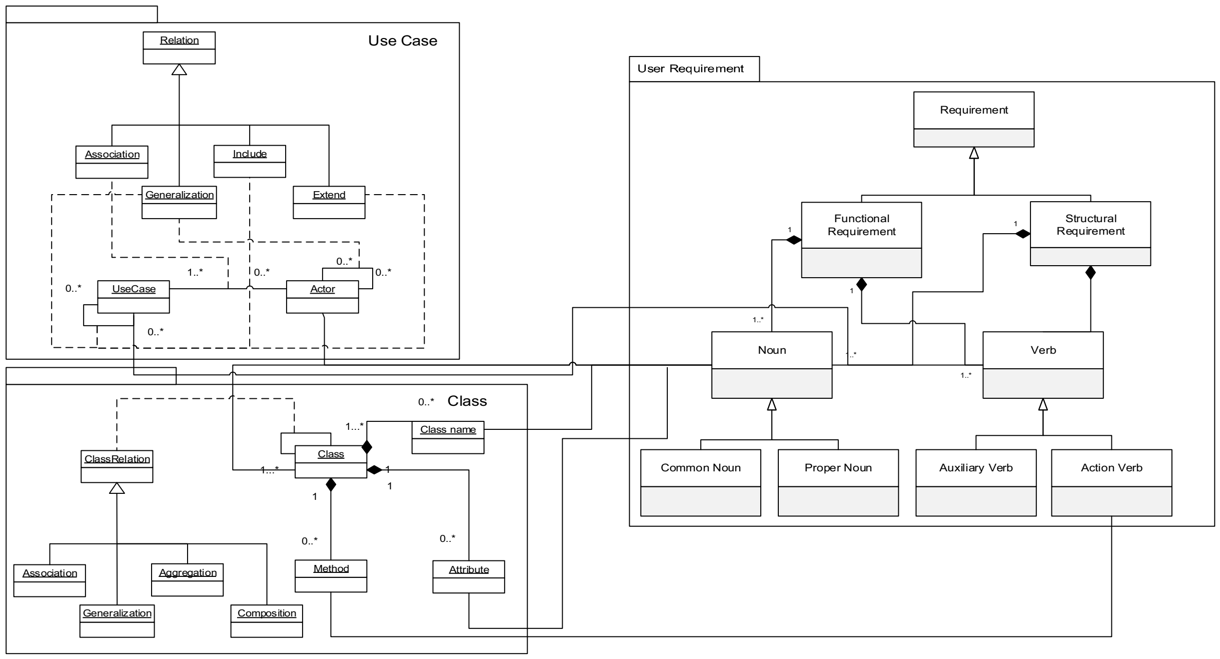


Figure 1. Structural diagram of Components relationship of natural language sentences and the model [4]

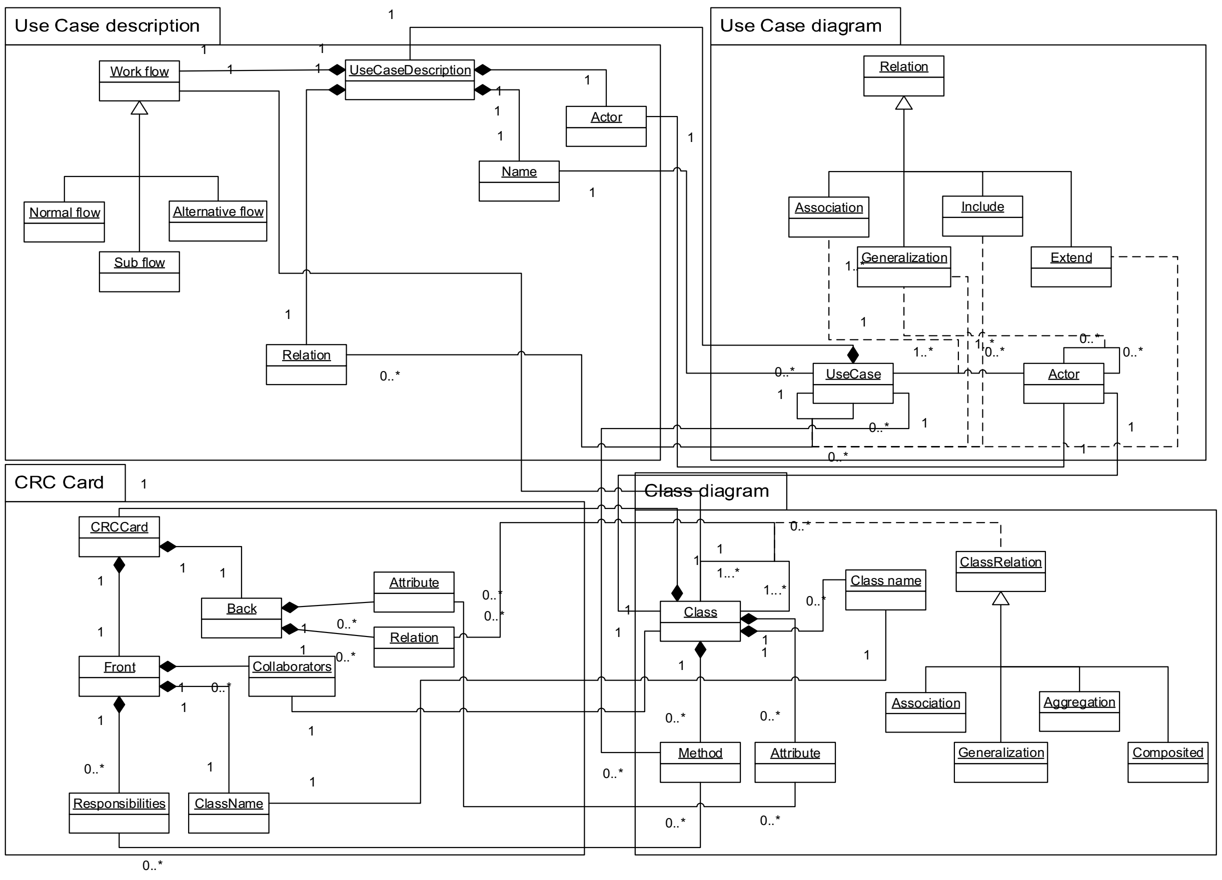


Figure 2. Structural diagram of the components relationships of the model and model description [4]

II. Related Theories

A. Software models

This research is interested in verifying the requirements characteristics between natural language sentences, software models constructed from UML, and descriptions. The software models that this research focuses on are the functional model, namely, the use case model and the use case description, and the structural model, namely, the class model and the class description, as both models are commonly used in identifying requirements in requirement documents. The most important components of the models are as follows:

1) Use case models and use case descriptions

The use case model is used to illustrate the services of the system provided to different types of users, with important components as follows: the Use Case, or the functionality, and the Actor, or the user. Also, within the model, there are relationships in various forms, namely, association, generalization and extent/include relationships. The aforementioned components are used to create relationships with the sentence structure in the functional requirements sentence.

The use case description is created to describe the details of the use case, and to make the use case more detail. The use case description should have a 1-to-1 relationship with a single use case. The important components are the Name, the Actor, the Description, the Relation, the Normal flow, the Sub flow, and the Alternative flow. These components will be used to create a relationship with a use case diagram and fuctional requirements sentence for further verification.

2) Class model and class description

The class model is used to show the information that should be stored by the system, as well as the characteristics of the data to be stored. The important components are the Class, the Attribute, and the Operation. There are also relationships between classes to be used in verification: Association, Generalization, and Aggregation/Composition. The aforementioned components are used to create relationships with the sentence structure in the structural requirements sentence.

The class description is created to describe the details of the class model, and to make the class more detail. The class description should have a 1-to-1 relationship with a single class. The important components are the Class name, the Description, the Responsibilities, the Collaborators, the Attributes, and the Relations. These components will be used to create a relationship with the class model and structrual requirements sentence for further verification.

B. *Good characteristics of requirements specifications*

Eight good characteristics of requirements specifications are specified in the IEEE 830 standard [1]: Correctness, Unambiguity, Completeness, Consistent, Ranking for importance and stability, Verifiability, Modifiability, and Traceability. This research focuses on the Unambiguity, Consistency, and Traceability characteristics, with contexts with those characteristics being defined to be suitable for the verification of requirements in natural language sentences form, models and explanations. Details are as follows:

Unambiguity: That is, natural language sentences must not include qualifiers that cannot be measured, such as *"Customer can check out many items in store."* The sentence contains the word "many", which is a qualifier cannot be measured.

Consistency: That is, Consistency occurs if none of the following 3 contradictions occur:

1) The contradictions in relationships of tokens between requirements in natural language sentence form, and components within the model. For example, in functional natural language sentences, there are the following sentences: "Wholesaler and customer should be able to search item from list item. Wholesaler should be able to create items. Wholesaler should be able to update items. Customer should be able to check out online." The use case models represent this requirement was shown in Figure 2.

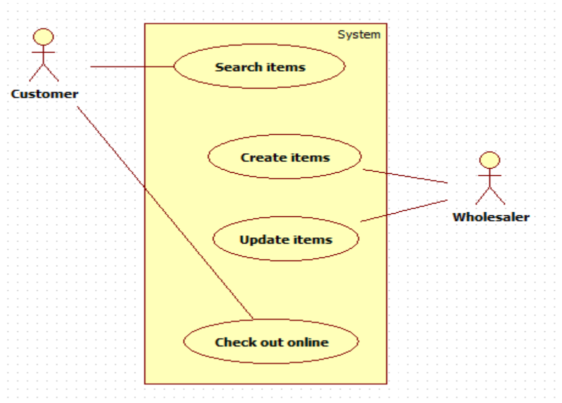


Figure 2: Example of use case model contradict from software requirement in natural language sentence

In the functional natural language sentences, there is a relationship between the noun *"*Wholesaler*"* and verb *"*Search items*"*, which is converted to an actor and use case with association relationship. However, the use case model does not include that relationship, causing the use case model to lack the consistency property.

2) The contradiction between natural language sentences with the multiplicity value in the class diagram. For example, in the natural language sentences, there is the sentence: "It is possible that each customer may buy at most 8 discount items." However, in the class model, the multiplicity between the Customer and Discount items is not 1 - 0. .8.

3) The contradiction between words and the role of the token in the natural language sentences and components of the model. For example, in the natural language sentences, there is the sentence "Item has name and price", whereas the class model for the item class does not have an attribute named price, etc.

Traceability: That is, the ability to be traced, or to determine the origin in both forwards and backwards direction. For example, in structural natural language sentences, there are the following sentences: "Customer searches item by name or category. There are exactly two types of items, normal items and discount items. It is possible that each customer may buy at least 1 item. It is possible that each item can be created by wholesaler. The class model represent this sentence was shown in Figure 3.

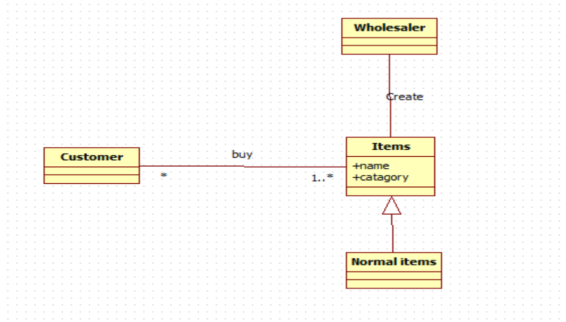


Figure 3: Example of class model contradict from software requirements in natural language sentences

In the structural natural language sentences, there is the word "Discount items", which is a noun that indicates a class in the class model, whereas the class model does not include the discount items class, causing the class model to lack the Traceability property.

III. RESEARCH METHOD

The research method is divided into 2 parts: a method for requirement characteristics verification, and a tool development to automatically assist in verification.

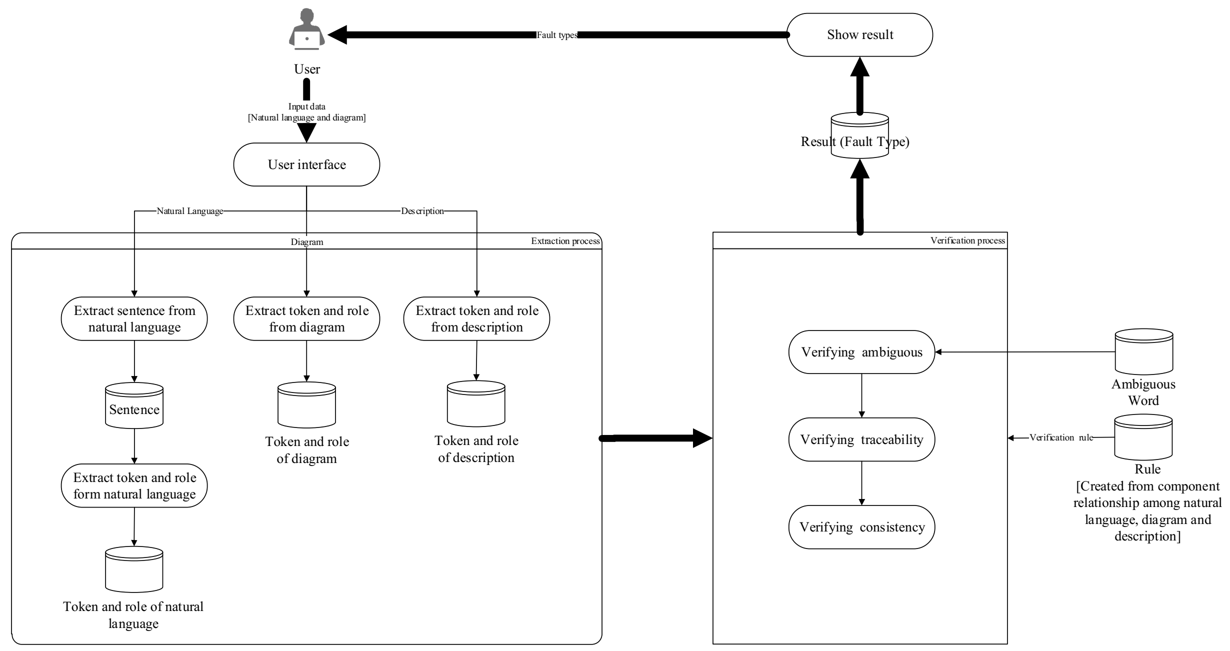

User
Fault types
Show result
Input data
[Natural language and diagram]
User interface
Result (Fault Type)
Natural Language
Description
Diagram
Extraction process
Verification process
Extract sentence from natural language
Extract token and role from diagram
Extract token and role from description
Sentence
Token and role of diagram
Token and role of description
Extract token and role form natural language
Token and role of natural language
Verifying ambiguous
Verifying traceability
Verifying consistency
Ambiguous Word
Verification rule
Rule
[Created from component relationship among natural language, diagram and description]

Figure 4. Method for verifying the presented requirements characteristics

A. *Requirement characteristics verification method*

The requirement characteristics verification method is split into 2 steps: The step of extracting words and parts of speech from the requirements in natural language sentences form including the explanations, and the step of requirementn characteristics verification. The overall picture of the requirement characteristics verification method was shown in Figure 4.

1) Extraction of words and parts of speech from natural language sentences

The step of extracting words and parts of speech from the Natural language sentences used for verification has details as follows:

a) Preparation of natural language requiremetns sentences

The user imports the natural language requirements sentences as a set of paragraph. This step will extract into a sentence form, and store in a database for further extraction of words and parts of speech in the sentence.

b) Extraction of words and parts of speech

This step extracts the words and parts of speech in natural language sentences stored in the database using a tool called the Stanford passer. For example, the structural natural language sentence is "Customers have 2 types as VIP customer and normal customer", . When analyzed this sentence with the Stanford passer [7], the obtained result is: [Customer/NN have/VBP 2/CD type/NN following/VBG VIP/NNP customer/NN and/CC normal/JJ customer/NN]. The results show the part of speech in the sentence. For example, "Customer" acts as a noun in the sentence, while "and" acts as a conjunction in the sentence etc. This information will be stored in a database for further analysis of relationships in the sentence.

c) Extraction of relationships between words in the sentence

After obtaining the parts of speech in each sentence, the next step is to extract the relationships between words in the sentence, using a tree diagram to assist in relationship extraction. The results are shown in Figure 5.

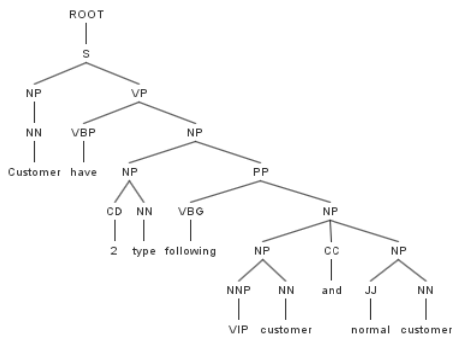


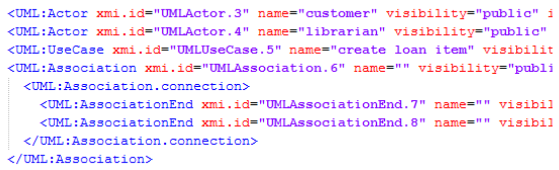
Figure 5. Words and parts of speech in a tree diagram

Figure 5 shows the extracted relationships between words in sentence, such as "Customer" acting as a noun and norminative in the sentence etc.

When the words and parts of speech obtained from the tree diagram are mapped with the relationships between the components of the sentences and model in Figure 1. For example, in this example case, it can be observed that the nouns in the sentences without qualifiers for the sentence of the subject will be created in to classes in the class model. In the example, it can be observed that Customer, VIP customer and normal customer will be turned into classes in the class model.

2) Extracting words and parts of speech from the models

The user imports model information in the form of XML files, using a tool called StarUml [8]. The results are as shown in Figure 6.



```
<UML:Actor xmi.id="UMLActor.3" name="customer" visibility="public" i
<UML:Actor xmi.id="UMLActor.4" name="librarian" visibility="public"
<UML:UseCase xmi.id="UMLUseCase.5" name="create loan item" visibilit
<UML:Association xmi.id="UMLAssociation.6" name="" visibility="publi
  <UML:Association.connection>
    <UML:AssociationEnd xmi.id="UMLAssociationEnd.7" name="" visibil
    <UML:AssociationEnd xmi.id="UMLAssociationEnd.8" name="" visibil
  </UML:Association.connection>
</UML:Association>
```

Figure 6. XML file obtained from StarUML tool [8]

From Figure 6, the words and parts of speech are extracted, where words are obtained from the attribute "name", and the part of speech can be extracted from "UML:", for example "UML:Actor" acts as an actor in the use case model etc.

3) Extracting words and parts of speech from description

The user imports the data through a program user interface where the components of the use case and class descriptions have already been specified. After the user has imported the data for each description, the system will extract the words from the data imported by the user, and the parts of speech from the specified components, and store them in a database for further use in verification.

4) Requirement characteristics verification

The step of requirement characteristics verification will verify 3 characteristics: Unambiguity, Consistency, and Traceability. Five relationships will be verified: the relationships between functional natural language sentences and the use case model, the relationships between structural natural language sentences and the class model, the relationships in the functional model, the relationships in the structural model, and the relationships between the functional and structural model. Each characteristic has a verification procedure as in Table II

TABLE II. REQUIREMENTS CHARACTERISTIC VERIFICATION STEPS

Step and example result	Type of characteristic verified		
	Ambiguity	***Consistency***	***Traceability***
Step 1. Extract source information and store tokens in to the database	Data extraction from functional and structural natural language sentences	Data extraction from relationship between the tokens of natural language sentences, models, and model descriptions	Data extraction from words and parts of speech in natural language sentences, models, and model descriptions
Step 2. Compare words and parts of speech appear in all source information	Compare with 300 ambiguous words stored in database	Compare the following relationships: Relationships of tokens of functional natural language sentences and the use case model, relationships of tokens of structural natural language sentences and the class model, relationships of the token of the use case model and the use case description, and relationships of tokens of class model and class description	Compare tokens according to the following relationships: tokens of functional natural language sentences and the use case models, tokens of structural natural language sentences and the class models, tokens of the use case model and use case description, tokens of the class model and class description, and tokens of the use case model, use case description, and class description
Step 3. Present verification result	System will identify defects under the unambiguity heading, and inform the user where ambiguous words appear.	When verification detects missing relationships between tokens, or lack of consistency, system will identify the defects, including the relationships between the aforementioned tokens, on which parts are missing or consistent.	When verification detects missing, additional, or additional tokens, the system identifies the defect, as well as the token, on where it appears or is missing.
Example result	Customer can check out many items in store. This sentence has ambiguous word which is "many".	In Functional Natural Language, there is a function provided for a wholesaler to searh items but does not appear as an association relationship in Use Case diagram	In Structural Natural Language, there is a discount items which should be a class in a class diagram but it does not appear in Class diagram.

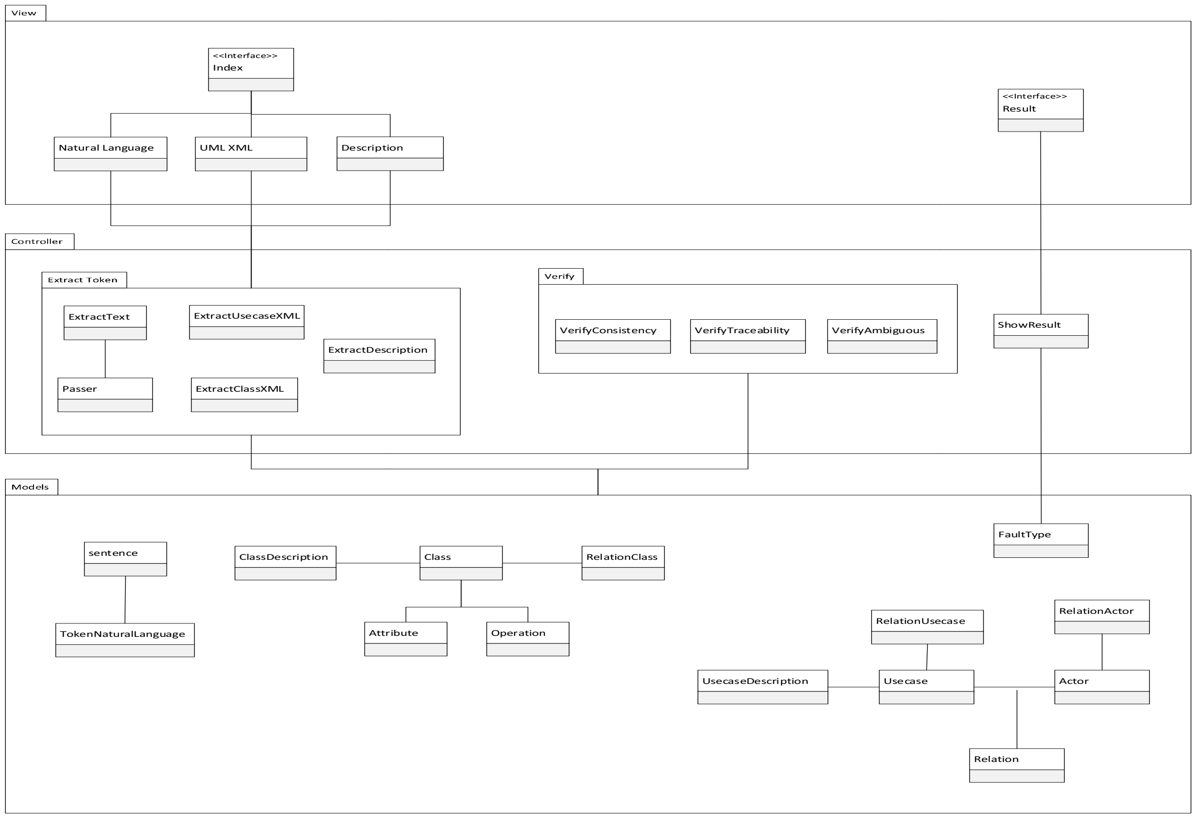


Figure 7. Software architecture of tool used for verification

B. Creating the tool and application of rules

For developing the automatic tool for verifying the requirement characteristics, namely unambiguity, consistency, and traceability, using the rules created from components of the requirements, and the aforementioned methods for extracting words and parts of speech, the software architecture of the tool is as shown in Figure 7, and the services of the tools are shown in the form of the use case diagram in Figure 8.

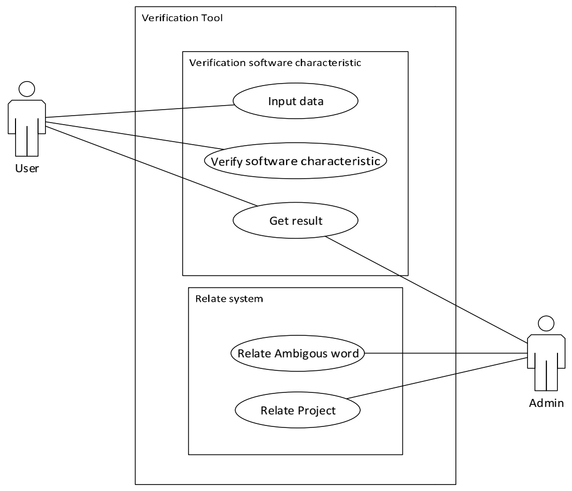


Figure 8. System use case diagram

The user imports the source data used for verification, namely, the natural language sentences, the XML files of the model, and the model description from the interface. The interface page for capturing natural language and XML files is as shown in Figure 9, and the import interface for model description is as shown in Figure 10.

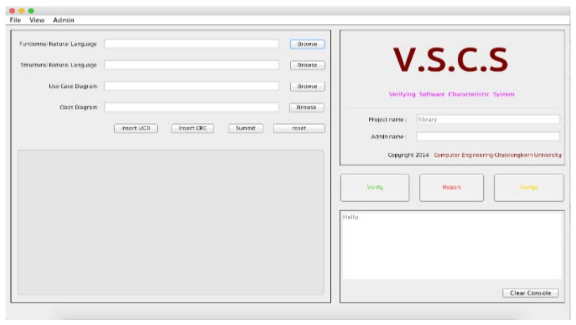
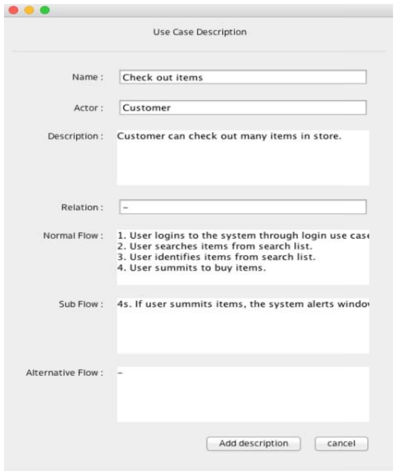


Figure 9. Interface for captuing requirements in natural language and XML file



Name : Check out items

Actor : Customer

Description : Customer can check out many items in store.

Relation : –

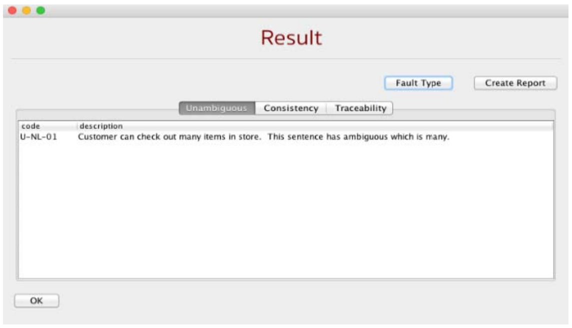
Normal Flow : 1. User logins to the system through login use cas
2. User searches items from search list.
3. User identifies items from search list.
4. User summits to buy items.

Sub Flow : 4s. If user summits items, the system alerts windo

Alternative Flow : –

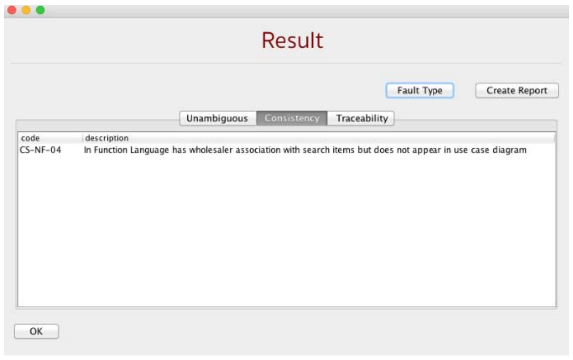
Figure 10. Interface for importing use case description

When the user begins the verification, the system verifies according to the procedure in Table 2, and the system will display the results as error codes, as well as the details of the defects to inform the user what types of defects have been found. Example results are from verifying the examples presented in section 2.2. The results are categorized according to the characteristics as follows: Figure 11 shows the results from unambiguity verification, while Figures 12 and 13 show the results from consistency and traceability verification respectively.



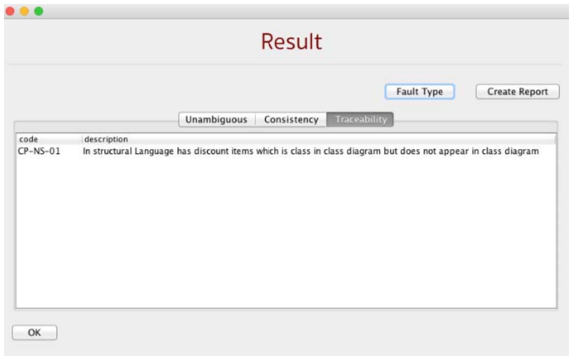
code	description
U-NL-01	Customer can check out many items in store. This sentence has ambiguous which is many.

Figure 11. Result from unambiguity verification



code	description
CS-NF-04	In Function Language has wholesaler association with search items but does not appear in use case diagram

Figure 12. Result from Consistency verification



code	description
CP-NS-01	In structural Language has discount items which is class in class diagram but does not appear in class diagram

Figure 13. Result from Traceability verification

IV. CONCLUSIONS AND FUTURE RESEARCH

This research has presented a method for verification of requirement characteristics, namely unambiguity, consistency, and traceability, by using the rules created from the relationships between various components of the requirements, consisting of functional natural language sentences, structural natural language sentences, use case model, class model, use case description, and class description. The verification was based on rules construct from elements relationship from 5 types: 1) Relationships between functional requirements in natural language sentences and use case model 2) Relationships between structural requirements in natural language sentences and class model 3) Relationships within the functional model 4) Relationships within the structural model 5) Relationships between the functional and structural models. Morover, to facilitate the proposed verification method, a tool was developed for automatically verifying those characteristics, in order to assist the software engineer in verifying the requirement characteristics, as well as identifying the defects, to allow the software engineer to correct those requirements to meet target quality and help satisfy the user requirements.

In the future, the developed tools will be used for verification of the actual software requirements from the industry. The feedback earned will be analyzed and identified the feasible tool service to further meet the expected requirements. In addition, the proposed method will be expaned for the verification between requirements in natural language and the design model.

REFERENCES

[1] L. Xiaoshan, L. Zhiming, and H. Jifeng, "Consistency checking of UML requirements," in Engineering of Complex Computer Systems, 2005. ICECCS 2005. Proceedings. 10th IEEE International Conference on, 2005, pp. 411-420.

[2] F. Mokhati, P. Gagnon, and M. Badri, "Verifying UML Diagrams with Model Checking: A Rewriting Logic Based Approach," in Quality Software, 2007. QSIC '07. Seventh International Conference on, 2007, pp. 356-362.

[3] T. Li, Y. Zongyuan, and X. Jinkui, "UCVSC: A Formal Approach to UML Class Diagram Online Verification Based on Situation Calculus," in Computer Sciences and Convergence Information Technology, 2009. ICCIT '09. Fourth International Conference on, 2009, pp. 375-380.

[4] N. Phanthanithilerd and N. Prompoon, "Verifying Software Requirements Characteristics Based on Rules Defined from Software Component Relationships," Lecture Notes on Software Engineering, vol. 4, 2016.

[5] "IEEE Recommended Practice for Software Requirements Specifications," IEEE Std 830-1998, pp. 1-40, 1998.

[6] D. P. Tegarden, A. Dennis, and B. H. Wixom, Systems Analysis and Design with UML: Wiley, 2013

[7] The Stanford NLP Group. (n.d.). 2015, from http://nlp.stanford.edu/index.shtml

[8] StarUML. (n.d.). 2015, from http://staruml.io