Toward Requirements and Design Traceability using Natural Language Processing

Omer Salih Dawood and Abd-El-Kader Sahraoui

***Abstract*—The paper aimed to address the problem of incompleteness and inconsistency between requirements and design stages, and how to make efficient linking between these stages. Software requirements written in natural languages (NL), Natural Language Processing (NLP) can be used to process requirements. In our research we built a framework that can be used to generate design diagrams from requirements in semi-automatic way, and make traceability between requirements and design phases, and in contrast. Also framework shows how to manage traceability in different levels, and how to apply changes to different artifacts. Many traceability reports can be generated based on developed framework. After Appling this model we obtained good results. Based on our case study the model generate a class diagram depends on central rule engine, and traceability was built and can be managed in visualize manner. We proposed to continue this research as its very critical area by adding more Unified Modeling Language(UML) diagrams, and apply changes directly inside software requirement document.**

***Index Terms*—Requirement Engineering; Requirement Management; NLP; Requirement Traceability; UML; Software Design.**

I. Introduction

As in [1] STANDISH report found around 9,336 IT projects (69%) failed because improper requirements management, Lack of user input, and poor end-user training, and Incomplete requirements, rapidly changed, ambiguity, and poor contract managements and draft in the field of requirements there is important concept known as requirements traceability.

Software contains many artefacts; each artefact performs a specific task. In order to perform system operations these artefacts should be interacted and communicate together. There may be many artefacts that need to be linked together to accomplish system task. The process of making traceability differs from system to another, this process should guarantee a traceable artifact. In this paper a good Framework was developed, this framework assists in generate UML diagrams in semi-automatic way, and build the requirements to design traceability report. This report shows different changes that made in requirements or design and apply all these changes to UML diagram, in this study Class diagram is semi-automatically generated using (NLP) and design build rules. From Fig. 1 the traceability can be done at different levels in requirements and design phases. The main purpose of traceability is how to apply keep the impact of the change. Our research concentrates on first level (SRS) and second level (SADD), SRS is written in Natural Language while the de facto standard of design is UML. Based on our work the third level (SDDD) can be applied.

The research contribution will assist in making traceability in an easy way, and ensure a high level of completeness and consistency between requirements and design, so that we can ensure a high quality of software products from earlier stages. The rest of paper consist of following sections, section two about requirements traceability, section three about related work and previous study, section four about a Framework and Methodology, section five is conclusion, section 6 talking about future work, and section 7 is appendix.

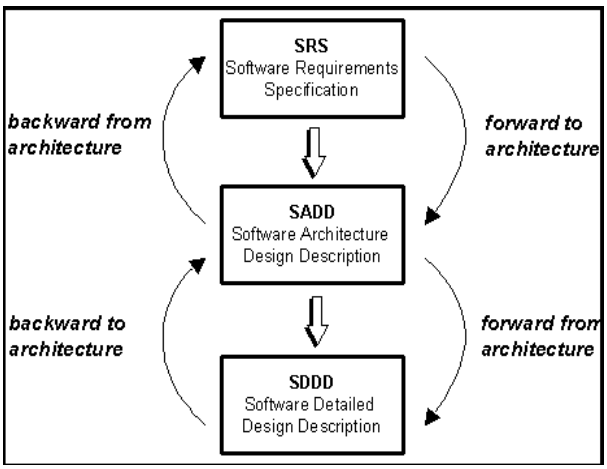

Fig. 1. Traceability between requirements and design [25]

II. Concept of Traceability

According to Francisco [2] IEEE standard 830-1984 states that the requirements specification is traceable if (i) the requirements source is clear, and if (ii) it allows for referencing of each requirement in future changes.

There are many definitions for requirements traceability [3]-[5] as follows:

1. Provides a relationship\link between the requirements, the design, and the implementation of final system.
2. Discover the history for each feature in a system so that the impacts of changes in requirements can be identified.
3. The ability to describe and follow the life of a requirement in both a forwards and backwards direction.

Requirements traceability can be considered as backbone

Published on July 31, 2018.

O. S. Dawood is researcher in Sudan University of Science and Technology, Khartum, Sudan, (e-mail: omercomail@gmail.com).

A. Sahraoui is professor in LAAS-CNRS, Université de Toulouse, CNRS, UT2J, Toulouse, France.

of any projects and helps ensure correct product delivery that meet the expectations of the clients [6].

There are six questions about artefacts the trace will answer [7]:

1. What information is recorded in the artefact? Is it an assumption, requirements, environmental constraints?
2. Who is created, or maintain the artefact and documented information? Involved stakeholders, and who is belong to user group?
3. Where the information comes from, source of information?
4. How represent the information, formally, informally, graphic?
5. When the artefact has been created, modified, or evolved?
6. Why artefact has been created, modified, or evolved? What is rational? alternatives, and why this alternative?

A. *Usages of Requirements Traceability*

Requirements Traceability is Important, and can be used in [5]:

1. Verification and Validation to assess the completeness, consistency, impact analysis, and discover requirements conflicts.
2. Maintenance to assess change requests, and tracing design rationale.
3. Document access, facilitates in finding information quickly in large documents.
4. Process visibility, ability to see and track how the software was developed, and allow for audit trail.
5. Management: managing and control changes and risks.

B. *Traceability Modes*

There are many ways to perform requirements traceability as in Fig. 2, regarding direction traceability can be defined into forward and backward direction, another way performs traceability in according to evolution which means before or after inclusion in requirements specification, lastly traceability can be performed depending the objects involved into inter or extra traceability as in Fig. 2 [2].

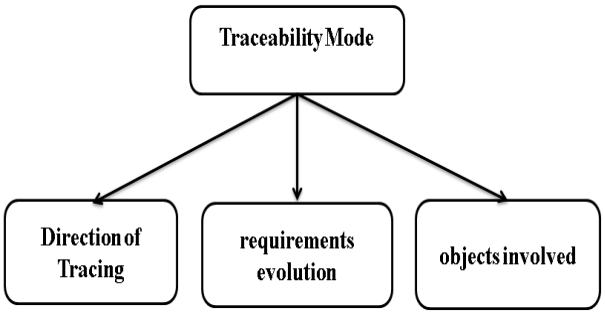

Fig. 2. Traceability Modes

C. *Forward and Backward:*

Describing trace direction between requirements and other software artefacts as shown in Fig. 3.

1. Forward Traceability:

Tracing the requirements from sources to their resulting product requirement(s) to ensure the product requirement specification is complete. This type of tracing allows to trace each unique product requirement forward into the system artefact, the design that implements that requirement, the code that implements that design and the tests that validate that requirement, and even maintenance. The objective is to ensure that all requirements are implemented and tested according as specified from source. [9],[2].

2. Backward traceability

Tracing each unique artefacts or work product (design element, object/class, code unit, and test cases) back to its associated requirement. The purpose of backward traceability is to verify that the requirements have been kept current with the design, code, and tests. In other words, tracing each requirement back to its source(s) [9],[2]. Traceability in Both directions (forward and backward) traceability known as bi-directional traceability

3. Bi-directional requirements traceability

Used to analyze the impact of a change, by ensures that changed requirements affect in work products and requirements affected by a change or defect in a work product, also assess current status of the requirements, and identification of missing requirements [9].

D. *Pre-requirements specification (pre-RS) and Post-requirements specification (post-RS)*

1. Pre-requirements specification (pre-RS) traceability:

As shown in Fig. 2, Happen before staring of requirements specification, normally concerned with aspects of a requirement's life prior to its merging in the RS (requirement Production) [11]. To get information related to the process of elicitation [2].

2. Post-requirements specification (post-RS) traceability:

In contrast with pre-RS it is concerned with aspects of a requirement's life that result from its inclusion in the RS (Requirement deployment [10]. Most problems attributed to poor requirements traceability are due to inadequate pre-RS [11]. To get information related to its use after requirement elicitation.

E. *Inter and extra-requirements traceability*

Another way to trace requirements regarding objects. The requirements can be traced to other requirements or can be traced to other object

1. Inter-requirements traceability

Is ability to link requirement to other requirements, its refer to relationship between requirements.

2. Extra-requirements traceability

Is ability to link requirement to other artefacts, its refer to relationship between requirements and other artefacts.

Inter-requirements can be used to deal with requirements evolution, and changes. It's done to know all requirements derived from a given requirement, or its chain of refinement [2].

Traceability can be divided into two distinct groups of traceability practitioners based on how they viewed the value of traceability. Firstly, low-end traceability users who viewed traceability "simply as a mandate from project

sponsors", secondly high-end traceability users viewed traceability as "an important component of a quality systems engineering process" [11].

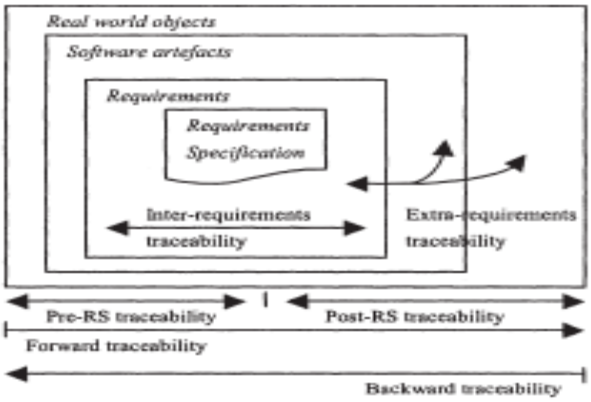

Fig. 3. Detailed Requirements Traceability Modes [2]

F. Functional and Non-Functional Tracing

Functional traces concern with well-built relationship between objects, while non-functional traces are those links related to intentions, purpose, goals, responsibilities, and other intangible concepts [2].

G. Traceability Visualization:

It's very important to show traceability in suitable way according to case situation, generally there are four ways to visualize traceability [12],[13]:

1. Traceability matrix is a table-like representation with two dimensions, consists from set of columns that represent some artefacts and rows that represent other artefact. Cell represents trace or relationship between artefacts. Traceability matrices are good for management tasks
2. Traceability graph is a multidimensional relationship between requirements and other artifacts, artefacts are represented as nodes. If any relationship exists, the edges connect between the nodes. Graphs are more suitable in development tasks.
3. List is representing traceability links in one entry. The entry may include information of source, target, and other related information. It's useful when set of operations of different artefacts should be executed.
4. Hyperlink is used to connect linked artefacts, and allow browsing target artefacts from source artefacts to show more information about traceability.

III. Related Work:

Saida et al [14] proposed a generic requirements traceability framework in case of complex system developments that based on SYSML.

Thomas et al [15] developed a literature review based reference model to support the cross-domain traceability of the requirements. This model can be used to develop tools that support the integration of products and services called Product Service System (PSS). They addressed four challenges, (1) is pre-specification traceability, inter-requirements traceability, post specification traceability, and traceability change. Andreas and Drik [16] used qualitative data analysis to improve the traceability of requirements. Their work is like theory building of social science through creation of requirements documents, glossaries, and domain model. The section of requirements document written in Natural language, then this natural language can be coded and transformed to UML Model. By their methodology can manage traceability in both directions [16]. To enable traceability between artifacts three types of relations were identified, overlap, implement, and refine. A traceability rules developed in XML that uses the relations and artifacts to identify a traceability between these artifacts [17].

Shota et al [18] address the issue of changing requirements and environment that leads to uncertainties in software development, they divide traceability into, option type, parameter type, and mixture type. In [19] the meta-model to provide explicit linkage with requirements engineering that enable traceability. The proposed meta-model divided into two parts: problem space, and solution space.

Sandhya and Dhirendra [20] proposed a framework for requirements management through requirements traceability their requirements consists from three phases: planning, execution, and management phase. They used the Requirements Traceability Matrix (RTM) to trace requirements with test case, each requirements is connected with one test case (one to one), or more test cases (one to many), then the traceability ca be used in forward or backward. Ana et al [21] evaluated traceability approaches in software products and proposed a Body Of Knowledge (BOK) for requirements traceability known as TraceBok to build the BOK Ahmed and thiam [22] provide a survey study in requirements traceability and product backlog change management in scrum. Renuka and et al [27] designed methodology to design traceability Software Requirements Specification (SRS) and Software Design Document (SDD), SRS Trace Item (SRSTI) is the template that populates data from SRS. The other one is SDD Trace Item (SDDTI) which if filled with data from SDD. Comparing SRSTI and SDDTI, if one to one mapping is not found then make tag mismatch, then generate mismatch report

Markus et al [28] address the heavy document problem of redundancy and synchronization for different stakeholder's viewpoints, they propose approach as single central requirements specification that maintaining one consistent view for all requirements of stakeholders to solve redundancy problem. According to [8] there are Difficulties making traceability from SDD to SRS, and they propose traceability guideline between software requirements and UML diagrams known FUTOR (From Uml TO Requirement) Guideline. As shown in Fig. 4, the requirements are classified according to requirements type (ReqType) in form of attribute, this ReqType is used as key for traceability guidelines that specify which UML diagram shall be used for the traceability.

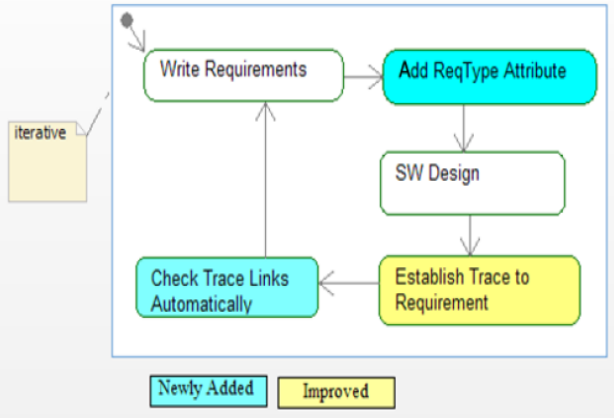

Fig. 4. FUTOR Guideline

Our research build traceability implicitly during design, when software designer selects design elements, these elements will be automatically linked with requirements elements, so that no more efforts needed to build traceability, also the research taking traceability in both ways (requirements / design). No previous research treated the traceability as our methodology. So this is a novel research in requirements and design traceability.

Many research taking about process of requirements text using NL to generate UML [23],[24], they developed some models to assist in generate the diagrams. In this research we adopt their methodology by building a central rule engine than can be used for many projects. The central rule minimizes the efforts of writing special rules for each project.

IV. Methodology:

This section describes case study that applied to developed framework, also the framework and proposed algorithm to generate a UML diagram in semi-automatic way, and make requirements traceability to design will describe in this section.

A. *Case Study*

Case study at [29] was used with some changes in sentence structure here is description of case study, Money Sender Initiate the transaction by sending the SMS containing national ID mobile PIN number, and mobile number of receiver and amount of money has to be transferred. User supposed to be use a mobile number to do this transaction. User able to send SMS message and view the reply SMS message. Money Receiver Receive an SMS message with authorization key transaction id, amount, sender mobile Number. Shows this message to the bank and collect money. Executives of bank login to the system, verify customer and issue money. Mobile operator receives the customer transaction and generate one-time password, send transaction to the bank and get reply message from the bank then send the reply with the on time generated password to the customer. Retailer Check the customer validity check transaction status of the transaction Receive the transaction details via SMS. Customer Requests purchase using the cashless purchasing system providing mobile number, amount, and NIC number. Bank group Verify Customer, Sends Confirmation message to retailer, Sends Confirmation message to customer, and Sends to Retailer. Money Sender is a User and Money Receiver is a User. Retailer is a Money Receiver, and Customer is Money Sender.

B. *Proposed Framework*

As in Fig. 5, the framework consists from the parts

1. Preprocessing of requirements text, in this step
   a) unnecessary words or stop words moved to allow better sentence processing
   b) Tokenization, to obtain more understandable words the text cab be divide up into tokens which can be used for further processing. Tokens can be words, numbers, identifiers or punctuation (depending on the use case) [26].
   c) Part of Speech Tagging, to identify phrases from noun or verb clauses [26].
   d) UML diagrams and artifacts identification rules, to allow better performance and rule redundancy we proposed a central rule engine that can be shared by multiple project.
2. UML diagram generation, based on rules described in step 1.4 all candidate classes, methods, attributes, relationships displayed, designer select needed elements and construct the class diagram.
3. Build and manage traceability, these modules depends on previous steps, selected design artifacts mapped with requirements when uml diagram generated, each specific UML artefact class, method, attribute and relationship connected with specific requirements processes text.

Requirements may consist from many classes or other artefacts. The mapping between requirements and design artefacts are stored in database. Both requirements and UML artefacts can be changed. If requirement artefacts changed the cross pending design artifacts will be changed in UML diagram and vice versa, if design artefacts changed the cross pending artefacts in requirements will be changed.to organize the process of changing artefacts a module of managing the changes was built to approve the change, and all changes were saved to allow tracing the artifacts. At the beginning the initiator or source of requirements is requirements phase a and target is design phase, when any suggested change happened in design, the change source will be designer at design stage.

For better traceability each artefact in requirements linked with design artefact a unique number was given, in many cases requirement has many design artifacts in this case if we need to track specific elements it will show all requirement elements. With traceability ID it's very easy to track a lifecycle of specific elements.

When approval of changes all linked artifacts will be changed. Finally, a detailed traceability report can be generated to show all changed happened to specific artifacts.

All parts of this algorithm are practically applied, and gave good result

Most of previous requirements traceability methodology didn't take the detailed traceability methodology that shows a full change of requirements and design. In our research we contribute by enhancing the process of building design in semi-automatic way through different phases of NLP processing and building rules. Also we built a traceability

matrix automatically during design generation.

This framework can be used to insure consistency and completeness between requirements and design, so that will enhance software quality.

The result of proposed framework is excellent and gave a good contribution in field of requirements engineering, especially in requirements to design phase.

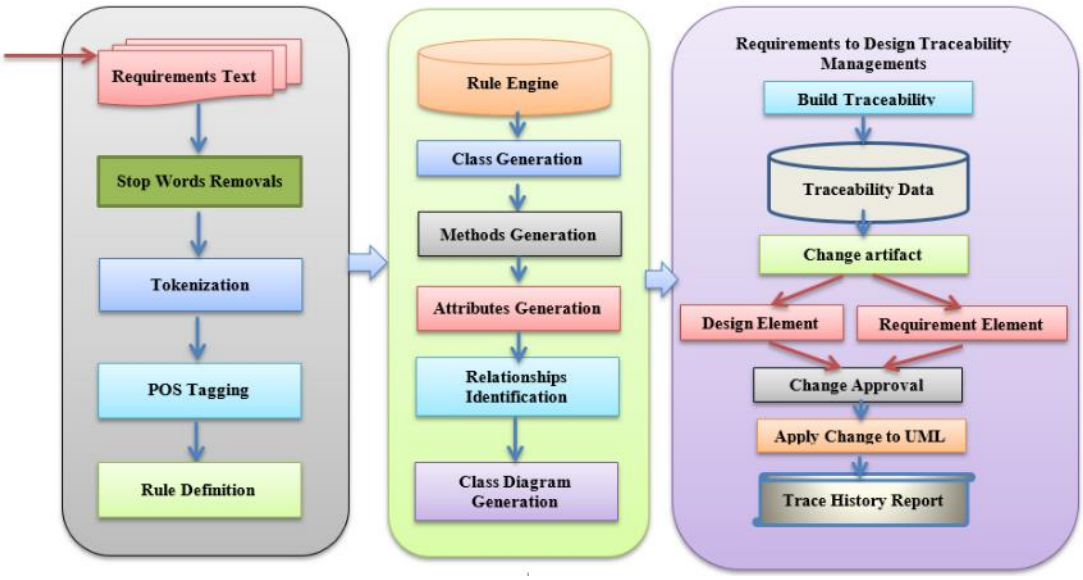

Fig. 5. Framework

C. Rules used to Generate Class diagram

According to our case study we built following rules to generate class diagram

1- Class identification rules
   a. Noun, singular(NN) + Noun, plural(NNP) + Verb(VBP)
   b. Noun, singular(NN) + Proper noun, singular(NNP)
   c. Noun, singular(NNP) + Noun, singular(NN)+Verb(VBZ)
   d. Noun, singular(NN) + Verb, base form + Noun, singular(NN)
   e. Proper noun, singular(NNP)+ Noun, plural(NNS)
   f. Proper noun, singular (NNP+ Noun, singular(NN)+ Proper noun, singular(NNP)
   g. Proper noun, singular(NNP)+ Verb, past tense

2- Method identification rules
   a. Verb, gerund or present participle + Proper noun, singular(NNP)
   b. Noun, singular(NN)+ Noun, singular(NN)+ Proper noun, singular(NNP)
   c. Proper noun, singular(NNP)+ Proper noun, singular(NNP)+ Noun, singular(NN)+ Noun, singular(NN)
   d. Verb, non-3rd person singular present(VBP)+ Noun, singular(NN)
   e. Verb, 3rd person singular present(VBZ) + Noun, singular(NN) + Noun, singular(NN)
   f. Verb, non-3rd person singular present(VBP) + Cardinal number(CC)+ Noun, singular(NN) + any words
   g. Verb, base form(VB)+ Noun, singular(NN))+ Noun, singular(NN)
   h. Noun, singular(NN)+ Noun, singular(NN)+ Noun, singular(NN)
   i. Noun, plural(NNS) + Verb, non-3rd person singular present (VBP)+ Verb, gerund or present participle(VBG) + Adjective (JJ)
   j. Noun, plural (NNS + Noun, singular(NN)+ Noun, singular(NN)
   k. Verb, base form(VB)+ Verb, base form(VB)+ Noun, singular(NN)
   l. Preposition(IN) + adjective (JJ) + Noun, singular(NN)

3- Attributes
   a. adjective (JJ)+ Noun, singular(NN)

4- Relationship rules

As in [23] some words can be used to determine the relationship type in example if two classes separated by (is a) this indicate there is generalization relationships, if words like (contain, consists,) this indicate there is composition or aggregation relationships, else the relationship is association, here are some used rules to generate relationships

   a. Verb, gerund or present participle(VBP) + Noun, singular(NN)
   b. Verb, gerund or present participle(VBP) + Proper noun, singular(NNP)
   c. Verb, gerund or present participle(VBP) + Verb, gerund or present participle(VBG)

Many rules can be used to generate classes, attributes, relationships and methods. Here the candidate elements are can be generated through rules. All these rules may be centrally stored in database to be used by many projects rather than developing rules for each project.

It is important to handle SRS change management in future research.

V. Conclusion

One of the largest issues in requirements engineering is how to keep high consistency between requirements and design, this problem happen if we want to trace requirements to design phase from design phase to requirement in many cases the requirements engineering many want to change requirements after design was made, in this case it's important to trace all made changes and apply these changes to design, and same steps happened to design. The research contribution is a model assists in process of generation design in semi-automatic way, to insure completeness and consistency of requirements and design, this process missed in current tools but was covered partially in many researches. During model generation the traceability information is stored in database, then any needed changes are approved and applied to linked artefacts. Many reports can be generated to show the status of requirements or design change.

VI. Future Work

As one of the most important issues in software engineering much work can be done, in process of UML generation more diagrams can be generated to give a full design views like use case diagram, class diagram, and sequence diagram, this means that the new rules may be needed to assists in process of building diagrams. Regarding traceability any change in requirements of design can be applied directly in SRS, but the problem is shared artifact items. Some requirements may be shared between different artefacts in example the login function can be shared between seller and buyer if we want to change login function, it will be changed to both seller and buyer.

VII. Appendix

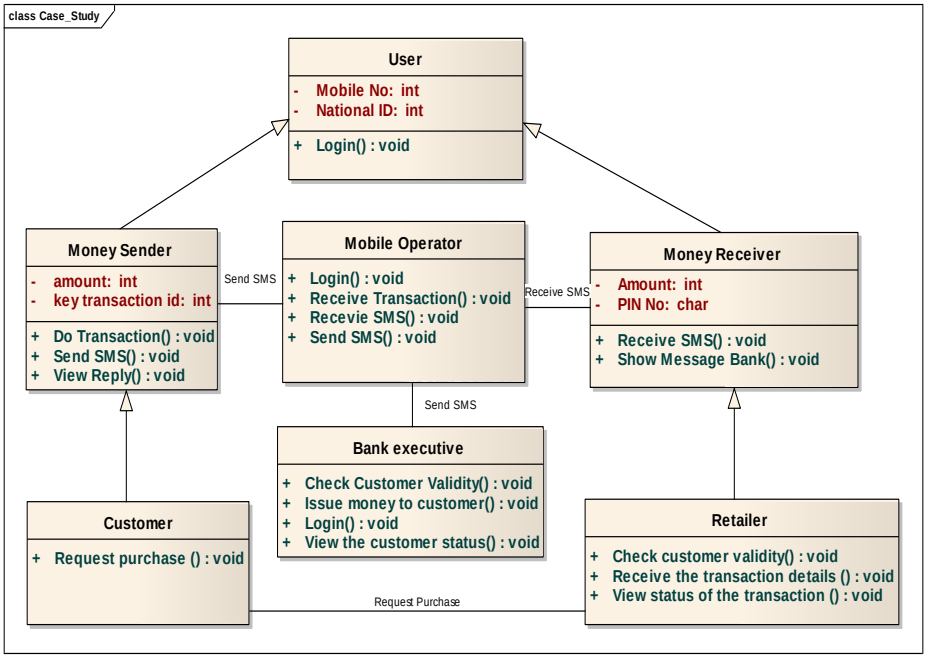

Fig. 6. Case Study Class Diagram ()

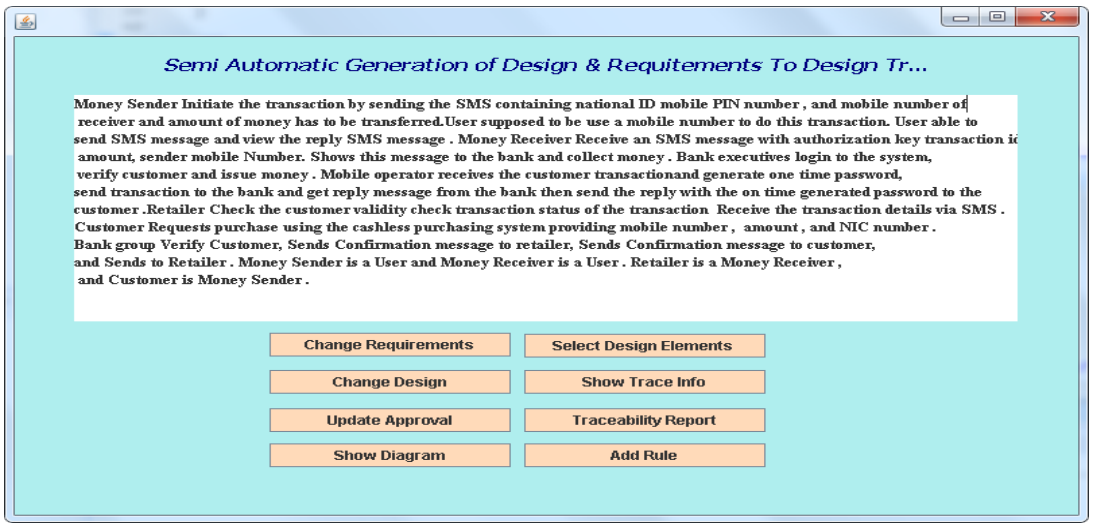

Fig. 7. Requirements text

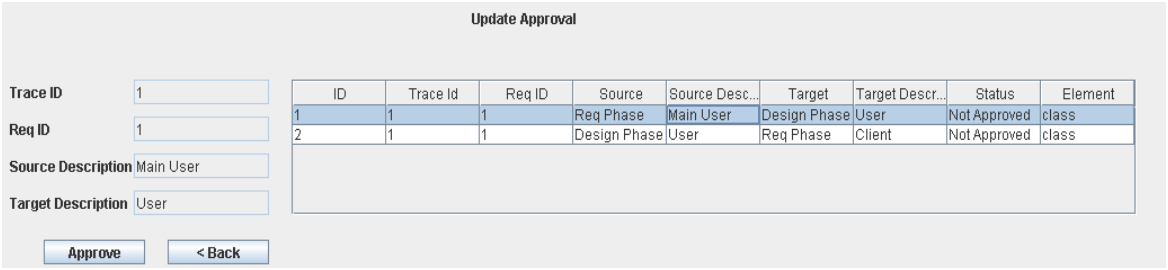

Fig. 8. Approve requested changes

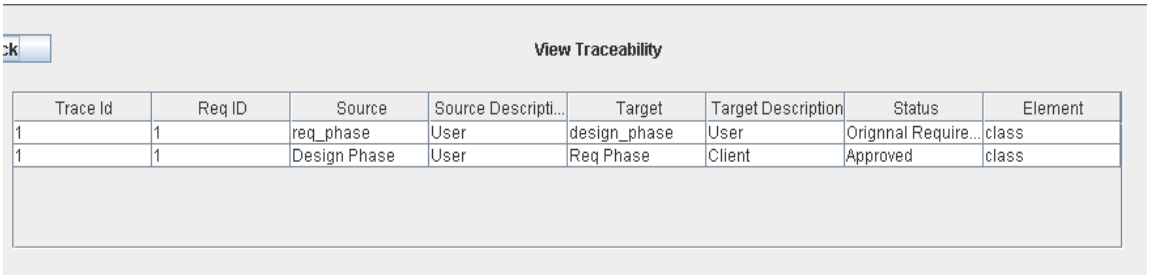

Fig. 9. Report shows that one design item has been changed and reflected to design and requirements

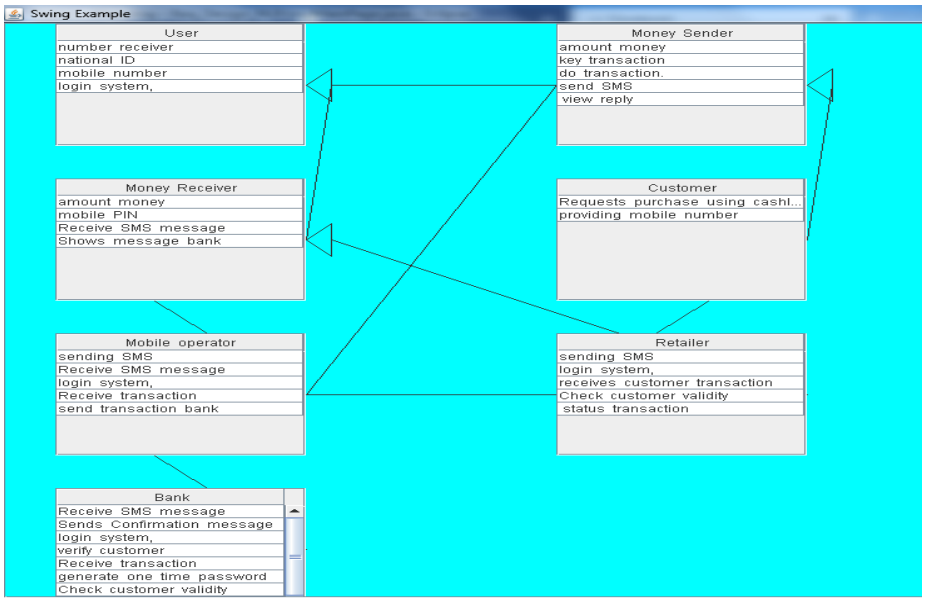

Fig. 10. Generated class diagram in semi-automatic way

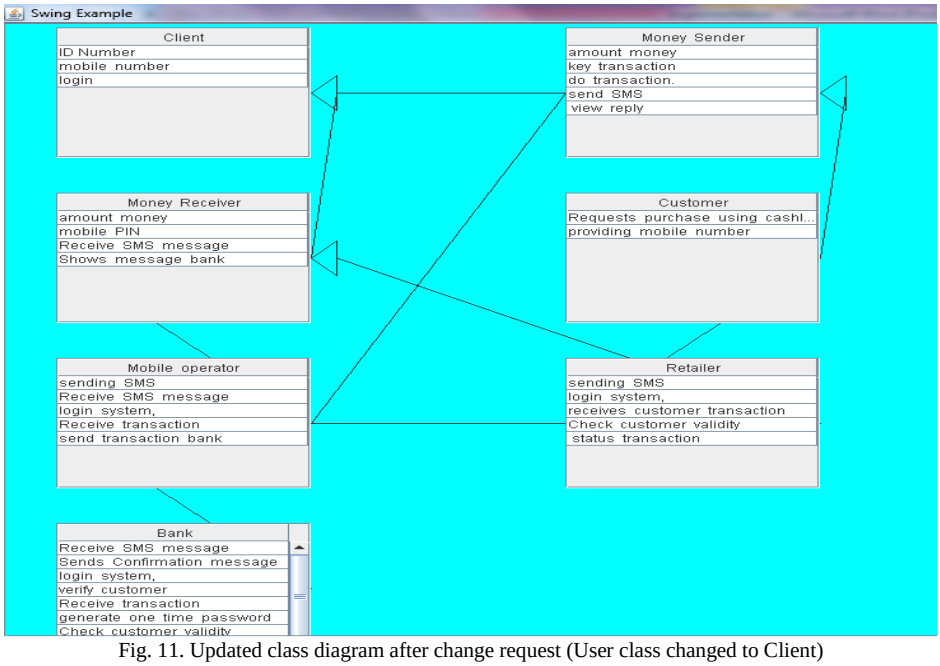

Fig. 11. Updated class diagram after change request (User class changed to Client)

REFERENCES

[1] Satyarthi, Sandhya, and Dhirendra Pandey. "Framework for Requirement Management using Requirement Traceability." *International Journal of Advanced Research in Computer Science* 8, no. 5 (2017).

[2] Pinheiro, Francisco AC. "Requirements traceability." In *Perspectives on software requirements*, pp. 91-113. Springer, Boston, MA, 2004.

[3] Ian. "Requirements Engineering Processes". "Software Engineering". "lecture notes"."Chapter 7"

[4] Kok, H. M. R. "Tracing Requirements in an Insurance Software Development Company." Master's thesis, 2016.

[5] Nicolas Sannier."Requirements Prioritization Requirements Management Requirements Traceability and Variability"."lecture notes"

[6] "CDC unified process practice guide requirements traceability".2006.[online] https://www2.cdc.gov/cdcup/library/practices_guides/CDC_UP_Requirements_Definition_Practices_Guide.pdf

[7] Winkler, Stefan, and Jens von Pilgrim. "A survey of traceability in requirements engineering and model-driven development." *Software & Systems Modeling* 9, no. 4 (2010): 529-565.

[8] Min, Hyun-Seok. "Traceability Guideline for Software Requirements and UML Design." *International Journal of Software Engineering and Knowledge Engineering* 26, no. 01 (2016): 87-113.

[9] Linda Westfall." Bidirectional Requirements Traceability".2006

[10] Gotel, Orlena CZ, and C. W. Finkelstein. "An analysis of the requirements traceability problem." In *Requirements Engineering, 1994., Proceedings of the First International Conference on*, pp. 94-101. IEEE, 1994.

[11] Williams, Jeandre Charisse. "A case study of pre-requirements specification traceability practices in a retail environment." PhD diss., University of Cape Town, 2015.

[12] Requirements traceability. https://en.wikipedia.org/wiki/Requirements_traceability#Usage_of_traceability_information .2018

[13] Li, Yang, and Walid Maalej. "Which traceability visualization is suitable in this context? a comparative study." In *International Working Conference on Requirements Engineering: Foundation for Software Quality*, pp. 194-210. Springer, Berlin, Heidelberg, 2012.

[14] Haidrar, Saida, Adil Anwar, and Ounsa Roudies. "Towards a generic framework for requirements traceability management for SysML language." In *Information Science and Technology (CiSt), 2016 4th IEEE International Colloquium on*, pp. 210-215. IEEE, 2016.

[15] Wolfenstetter, Thomas, Kathrin Füller, Markus Böhm, Helmut Krcmar, and Simon Bründl. "Towards a requirements traceability reference model for Product Service Systems." In *Industrial Engineering and Systems Management (IESM), 2015 International Conference on*, pp. 1213-1220. IEEE, 2015.

[16] Kaufmann, Andreas, and Dirk Riehle. "Improving traceability of requirements through qualitative data analysis." *Software-engineering and management 2015* (2015).

[17] Jirapanthong, Waraporn. "Requirements traceability on web applications." In *Information Technology and Electrical Engineering (ICITEE), 2015 7th International Conference on*, pp. 18-23. IEEE, 2015.

[18] Ishibashi, Shota, Kenji Hisazumi, Tsuneo Nakanishi, and Akira Fukuda. "Establishing traceability between requirements, design and operation information in lifecycle-oriented architecture." In *Advanced Applied Informatics (IIAI-AAI), 2016 5th IIAI International Congress on*, pp. 234-239. IEEE, 2016.

[19] Plataniotis, Georgios, Qin Ma, Erik Proper, and Sybren de Kinderen. "Traceability and modeling of requirements in enterprise architecture from a design rationale perspective." In *Research Challenges in Information Science (RCIS), 2015 IEEE 9th International Conference on*, pp. 518-519. IEEE, 2015.

[20] Satyarthi, Sandhya, and Dhirendra Pandey. "Framework for Requirement Management using Requirement Traceability." *International Journal of Advanced Research in Computer Science* 8, no. 5 (2017).

[21] Duarte, Ana Marcia Debiasi, Denio Duarte, and Marcello Thiry. "TraceBoK: Toward a software requirements traceability body of knowledge." In *Requirements Engineering Conference (RE), 2016 IEEE 24th International*, pp. 236-245. IEEE, 2016.

[22] Alsalemi, Ahmed Mubark, and Eng-Thiam Yeoh. "A survey on product backlog change management and requirement traceability in agile (Scrum)." In *Software Engineering Conference (MySEC), 2015 9th Malaysian*, pp. 189-194. IEEE, 2015.

[23] Shinde, Subhash K., Varunakshi Bhojane, and Pranita Mahajan. "Nlp based object oriented analysis and design from requirement specification." *International Journal of Computer Applications* 47, no. 21 (2012).

[24] Arellano, Andres, Edward Carney, and Mark A. Austin. "Natural language processing of textual requirements." In *The Tenth International Conference on Systems (ICONS 2015), Barcelona, Spain*, pp. 93-97. 2015.

[25] "Traceability Analysis". http://www.chambers.com.au/glossary/traceability_analysis.php 2018

[26] Paul Nelson . "Natural Language Processing (NLP) Techniques for Extracting Information" Available: https://www.searchtechnologies.com/blog/natural-language-processing-techniques . 2018

[27] Agarwal, Renuka, Rajiv R. Chetwani, M. Ravindra, and K. M. Bharadwaj. "Novel methodology for requirements to design traceability of onboard software." In *Advances in Electronics, Computers and Communications (ICAECC), 2014 International Conference on*, pp. 1-6. IEEE, 2014.

[28] Turban, Bernhard, Markus Kucera, Athanassios Tsakpinis, and Christian Wolff. "Bridging the requirements to design traceability gap." In *Intelligent Technical Systems*, pp. 275-288. Springer, Dordrecht, 2009.

[29] "Analysis and designing of the automated mobile money transfer syste", chapter 5. Available: http://dl.lib.mrt.ac.lk/bitstream/handle/123/1785/Chapter05.pdf?sequence=5