Journal of Theoretical and Applied Information Technology
15th July 2020. Vol.98. No 13

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

MEASURING RIPPLE EFFECT OF NATURAL LANGUAGE REQUIREMENTS CHANGE FOR ULS DYNAMIC REQUIREMENTS

AHMED SAFWAT ALY[1] , MOHAMED BADR SENOUSY[2] , ALAA M.RIAD[3]

[1]Teaching Assistant, Sadat Academy for Management Sciences, Department of Computer and Information Systems, Egypt

[2]Professor, Sadat Academy for Management Sciences, Department of Computer and Information Systems, Egypt

[3]Professor, Faculty of Computer and Information Systems, Department of Information Systems, Mansoura University, Egypt

E-mail: [1]asafwat@sadatacademy.edu.eg, [2] badr_senousy_arcoit@yahoo.com, [3] amriad2000@gmail.com

ABSTRACT

Ultra Large Scales Systems (ULS) or Ecosystems are growing dramatically alongside their interactions and dependencies among other system components, change management needs new tactics. As consider one change or more in ULS requirements may result in a lot of side effects in other running or planned requirements that could be called "Ripple Effect".
Different ULS elements are affected in this type of environments varying from RE workers, Change Requesters and Involved parties those can be called 'Crowed Sourcing" contributors in ULS environment. To challenge such problems, in this paper, we suggest a new methodology for requirements change evolution to able to measure the impact of several changes on ULS requirements which are represented by a Natural Language utilizing Similarity models. This paper reports on initial results of such an empirical study of Requirements change that led to ripple effects across an entire ULS environment, our case study around one of ecosystems for ERP with around 4480 stored requirement statements and closed to around 22 connected subsystems. We have used Natural Language Processing (NLP) and Similarity Models to support the model

Keywords: *ULS, Requirements Engineering, NLP, Similarity Models, Change Management*

1. INTRODUCTION

ULS Systems includes massive number of stakeholders and apparently that presence of all stakeholders is impossible, and the traditional methods are inapplicable. The Requirements Engineering processes therefore not only needs to increase their capabilities to include all these different stakeholders but needs also to be conducted in a carefully way in order to respond more quickly to different, conflicting, complex and changing needs. Change is inevitable in all large scales' projects. Changes are pushing the system to respond to external and internal requirements changes as long as the demands change.

In the typical ecosystems environments, a requirement change can be issued or raised by the system stakeholders, affecting one or more components, module lacking the awareness of how these changes may influence their systems.

Since, new programming languages, tools and cloud based applications are being available, new design and quick implementation needs arise. Therefore, whatever software practice is used, it is indispensable that it can adapt changes to the software being developed [1]. In ULS systems, the problems of change management are more serious than in traditional systems. The requirements have basically conflict with each other and are unknown until the time which are implemented [2].

Due to the dependency between components the change management is very difficult. How to apply these changes, so that all teams are informed about updates as quickly as

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

possible with minimum risk on the running functionalities. Dependency chain in developing system should be managed and taken into account. Because when a system is made up of assembled components, an error in system configuration may result in software miss-functioning. Requirements manipulation must be observed and controlled to guarantee that no single party or enterprises can noticeably change the system without understanding, and possibly getting approval from, the other participants [3].

Although there are a lot of studies exist to measure the impact of change at the single system level. Some work on measuring the impact of changing in place requirement on other requirement, and other measured the impact of changing Application Programming Interface (API) in framework and libraries. but limited studies have been performed on the impact of single or multiple requirements change on the entire Ultra Large-Scale Systems through decentralized development communities.

Always the requirements are expressed in Natural Language statements explained with different charts, those are the artifacts of the Requirements Engineering practices, so linking the raised changes with the huge repositories of requirements documents become a dilemma since these data are stored in unstructured formats.

The model was developed in order to be able to extract the list of textual requirements from requirements repository, then to measure the impact of different raised changes issued from different crowdsourcing parties using similarity models. Then the development teams extract the list of impacted requirements according to the similarity score ranging from 0 to 1 depending to the strength of impact

Finally, as a result, the model is able to detect the affected requirements by certain distinct number of changes affecting current requirements with different degree of impact that represented by a similarity score, the higher the similarity value, the greater the impact on ULS requirements.

1.1 Research Contributions

- We propose a model based on the Natural Language Processing (NLP) similarity models to measure the impact of group of Changes on ULS Requirements represented in Natural Language
- The model isn't biased on the way of writing either System Requirement or the change those are expressed freely by the requirements editors
- The model utilizes "Wordnet" dictionary to find the synonyms of the statements in order to measure the degree of similarity in meaning
- The model also detects and prioritizes the affected requirements that were provided by ULS Analysts and Changes requesters according to their similarity score
- The feasibility of the proposed model is validated by conjunction between Ultra Large-Scale ERP with around 4500 requirements expressed in English with around 45 raised change requests

The rest of paper is organized as follows: Section II elaborate the characteristics of ULS systems, and the limitation of current RE practices and the related effort while Section III describes the Change Impact Analysis and Requirements Traceability Metamodel. The model design and its techniques are presented in section IV. Section V unveil the results of model implementation. Section VI concludes the paper and additionally features the directions for future research.

2. BACKGROUND

The enormous availability of data, information, the continues adoption of technologies, and the continuous evolution of the systems force model and ground-breaking methodologies for building, operating, and managing software systems [4]. As a result of this continuous growing is that software systems must become more adaptable, reliable, fit, recoverable, customizable, and compliant to changing operational contexts, environments or system requirements.

ULS Systems have the following characteristics, those are far from today's systems:

- The broad variety of participants improve and uses these systems, it carries dissimilar, incompatible, complex and changing needs.
- The construction and existence are evolutionary, beside the growing need to assimilate new capabilities into ULS system while it's functioning
- A ULS will be described by diverse, variable and changing elements

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

- The scale of ULS systems affect to be certainly decentralized in a diversity of modes (Data, Development, evolution and operational management)
- Societies will not just be consumers of a ULS system, they will be parts of the system, affecting its overall growing behavior

Several examples of ULS can be specified; for instance, Health Care system incorporating systems to disease identification, cure, and management of patients records, transportation systems, and emergencies system, including several of them considered as critical embedded systems [5].

2.1 The limitations of Present Software Development Approaches

Today's software development atmospheres are deeply leaning to old-fashioned software development practices as they centralize them in a single development unit and points of control.

Meanwhile, the dev teams are first elicit the requirements and document the specifications, and then continue through complete design, programing, verification, validation and etc., while in ULS this cycle is impractical; Analysis and design methods must adapt total incompleteness, insufficiency, ambiguity, and non-determinacy in the requirements and practices that happen during the system development and evolution,

Therefor ULS require a new model of development that supports the following activities

- Combined and intersecting development, deployment and operational actions
- Distributed design process among the development, deployment and operational activities with many contributing groups
- Healthy systems structure that ensures the security and privacy of critical data and manages the constantly changing environments as the development and deployment will exceed the organization limitations that will require a
- Recurring development in an environment at run where the number of changes is huge and the period between modification time and run time is almost near time
- Decentralized development stimulates over many associations those require a modern method for verification and problems recognitions
- Extracting the systems architectures, [1]and the framework for developing and adapting ULS systems
- Dynamic and quick requirement response to preserve live ULS systems functioning abilities
- Automated validation to upkeep sustainable performing self-testing as the verification might be probabilistic, real-time and non-deterministic behavior [6]
- Automatically configure modules during the installation and protect the interconnected systems from failure when updates are installed as well as from incidents during the run time

2.2 Requirements Characteristics in ULS System

For previous generations of software projects, individuals could prospect to predict and control change by using modular design to segregate the effects of change. The change was studied and analyzed in a central manner. But the centralized management is being replaced by distributed management units that encourage invention, but the price is illogical and unanticipated. stretching older approaches arise a dilemma of either tying change and innovation by imposing interface stability or facing unacceptable levels of technical risk [7].

So, one of the ULS requirements characteristics [4] is "Ever Changing". Requirements change over time, so the changes demands often raised to a change or group of changes in requirements, changes might be issued from customers after requirements analysis or any other crowdsource [8] such as developers groups, acquirers, suppliers, testers, communities, or whoever is represented as a stakeholder, the ability to response to ever changing requirements in an distributed way cannot possibly take all different resolutions into account and manage them capably, or allow for rapid changes in response to instant needs [4]

So, the researchers in Requirement Engineering (RE) for ULS tried to find some automatic ways to measure and circulate the changes impact over all interconnected systems since the systems grow in size and complexity beyond the point that traditional ways fall behind it.

2.3 Related Work

This section displays the related work on approaches and methods for Requirement Engineering's change management, impact analysis and mining for ripple effects

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Arora et al, presented an methodology based on NLP which exploits the phrasal structure of the requirements sentences, through defining the elements of phrase to analyze the impact of changing one phrase to the others [9]

Johan et al, offered practical evaluations of the benefits of automated similarity analysis among textual requirements in identifying the relationships between the requirements [10]

In 2011 [11] Goknil et al, produced a requirement engineering model from traceability relations which the requirements worker has already stated to be used for consistency checking of relations and inferencing then in 2014 [12] the authors presented a requirements meta model using requirements relation types and their semantics. The proper semantics of associations and change types allows new projected changes to be determined

Also, in 2014 Torkamani et al, proposed and analyzed new decentralized approach for configuration management. It's able to demonstrate the dependency graph as well as management and transfer the change between different developers across the globe [13]

Table 3 presents the summary of the perspectives that are linked to our research. The "✓" sign represents that column is exist in the individual author's paper and "✗" symbol represents that the aspect is not present in this paper. The first column displays the authors who used NLP techniques for manage the changes, the second column shows the system size (Medium, Large, Ultra Large), the third column shows the implementation layer (Coding, Design, Requirements Engineering), the forth column shows the distribution and working teams (Centralized or Decentralized), and the forth column shows the source of change (Professional Technical Teams or Crowd sourcing), the fifth column shows the utilization level of NLP techniques, and the sixth column shows the level of decision making guidance.

To conclude existing literature for ULS Change Management on Requirements Engineering using NLP Similarity models has the impression that the research performed for measuring the change ripple effect on ULS requirements has not covered the decentralized nature of ULS stakeholders.

Specifically, change requests are issued by crowdsourcing stakeholder from different business domains and cultures. For the different language expression, we utilized Wordnet dictionary to find the all different synonyms among the requirements and changes. Therefore, whatever change requested is submitted, the model is extracting all possible synonyms to match against the requirements repository. Change impact analysis is performed on the whole requirements to provide all the possible affected requirements according to similarity scores, to give the development teams the capability to manage the raised changes.

2.4 Change Impact Analysis in ULS

Change Analysis is an essential phase of system evolution and it consists of sum activities before change implementation including the following steps:

(1) Dependencies extractions, which aims to pull out depended requirements and possible dependencies between various systems and modules
(2) Change impact analysis, that is used widely to specify the probable ripple effects caused by changes made to software
(3) Changeability valuation, an assessment of implementing the change
(4) Modification recommendation: that offers some productive change proposals to minimize the software maintenance exertion and cost [14]

For each requested change, an impact analysis is being done by system analysts at the initial stage, later the discussions are continued with the systems stakeholders in the efforts involved in implementing the change through different manual methods either workshops, meetings or even formal documented communications. Sine each single change in requirements always producing ripple effects [15].

Arvanitou et al, define Change impact as an analysis measures the consequences of systems changes, i.e., the propagation of changes to other parts of an ULS systems and here the propagation of requirements changes to other requirements and other requested changes, hence, it's named as (**Requirement Change Ripple Effect**).
Studying and quantifying the ripple effect can provide benefits both before and after considering the change:

- **Before considering the change:** for a requirement change requested from the user

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

- **After changes have been considered:** for other pending requirement changes, other in progress changes and the identification of relationships among all system requirements [16]

As a result, the change impact analysis is performed to determine the affected requirements to overcome the systems fall that occurs when changing requirements

This paper aims to develop a model in which we as change analysts can find the similarity between the requested change and the ULS requirements in order to address the affected set of requirements those are defined and documented in a Requirements Traceability Matrix. So, in section IV, the Requirements traceability matrix meta model is explained as it represents the main linguistic repository for the requirements

3. REQUIREMENTS TRACEABILITY CONTEXT

Requirements traceability context or matrix is captured during the requirements specifications phase to relate all the business requirements together. and used in all software development practices as an example project planning, analysis, requirements validations, change impact analysis, testing and requirements reuse

For example, change impact analysis might discover impacts on some related requirements when a change issued on a requirement, as an analyst may have to analyze a whole requirement documents to find the impact of a single change. That could lead to neglecting some other critical changes with a high proper risk on the systems beside the price of implementing this change may be several times higher than planned, additionally the high cost of redoing and fixing the defected parts of systems as a result of that. And therefore, it's not possible to plan for systems releases without considering the relations between requirements.

Gotel and Finkelstein [17] define traceability in the context of Requirements Engineering (RE) as:
Requirements traceability states to the ability to keep track the life of a requirement in a onward and backward path (i.e., from its roots, through its specification and development , to its succeeding employment and use, and through all phases of on-going amendments and iterations in any of these phases.

The concepts of forward and backward have their standards clarifications:

Forward traceability: is the capability to keep track of a requirement to ingredients of a design or implementation.

Backward traceability: is the ability to track a requirement to its origins, i.e. to a people, organization, law, agreement, etc. [18]

4. REQUIREMENTS AND CHANGES SIMILARITY MODEL SPECIFICATION

As described before the proposed model use the *Requirements Traceability information* not only to extract the similarity among the requirements themselves but also the similarity with proposed **Change Requests**.

Therefore, this paper pays attention to the similarity examination, that is performed so as to find the requirements that associated with the proposed change. Furthermore, the requirements engineer can discover it necessary to split or merge two or more requirements changes according to their complexity and similarity.

When the Requirements Expert decides whether two or changes are related or not, it's with regard to the understanding of the change context itself and the further impact on the requirements in place, surely, these decisions are made by humans, but automated analysis of information presented in natural language processing may provide the requirements engineering with some help concerning the similarity to help these judgements.

4.1 Requirement and Change Document Representation

There are numerous methods to represent a document which can be represented as a bag of words, where words are expected to appear individually, and the order is irrelevant. The bag of words or it can be called as "Words Vector" is broadly used in information retrieval and text mining

Each word represented as a feature in the subsequent data space and each document then becomes a vector containing of non-negative values on each feature. Here we use the frequency of a term as its weight illustrates terms that shown more frequently are more vital and descriptive for the document.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Let D = {d1, . . . , dn} be a set of documents
Let T = {t1, . . . ,tm} the set of individual terms occurring in D.
A document is then represented as a m-dimensional vector.
$\overrightarrow{td}$. Let tf(d, t) mean the frequency of term t ∈ T in document d ∈ D.
Then the vector representation of a document d is

$$\overrightarrow{td}. = (tf(d, t1), \ldots, tf(d, tm)) \quad (1)$$

Basically, the words that appears more frequent like "a" and "the", but neither are illustrative nor vital for the document's topic are removed.

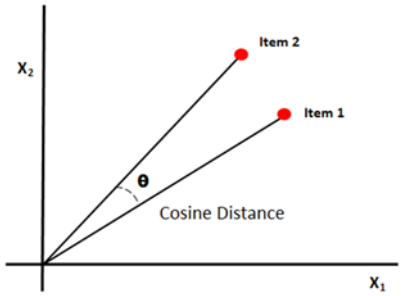

Figure 1: Angel between documents

With each document presented as vectors, we measure the degree of similarity between two documents as the association between their consistent vectors, which can be further enumerated as the cosine of the angle between the two vectors. Figure 1 shows the angle in two-dimensional plots but in practice the document plot usually has numerous of dimensions.

4.2 Similarity Measures

In order to find the degree of correlation between the requirements and changes, the researchers found extensive diversity of distance functions and similarity measures those have been utilized for clustering, such as squared Euclidean distance, cosine similarity, and relative entropy. [1]

All these methods reflect the degree of closeness and it gives a value between 0 and 1 to specify how similar a couple of sentences are, so 0 means that the sentences have no words in common while 1 means that the sentences are matching.

Cosine similarity: is a degree of similarity between two vectors of an inner product plot that calculates the cosine of the angle between them.

Dice's coefficient: is stated as twice the number of shared terms in the compared strings divided by the total number of terms in both strings.

Jaccard similarity: is calculated as the number of common terms over the number of all unique terms in both strings [2].

Natt et al., stated that Dice and Cosine Similarity measures are superior compared with Jaccard [3], while Vikas and Vivek [4] stated that Cosine similarity is clearly visible and best fit followed by Dice and Jaccard.

Strehel et al., found that extended Jaccard and cosine similarity performance is narrowly to human work outcome [5]. Agarwal et el., stated that time required when using the cosine similarity is less compared to Jaccard.

Jaccard coefficient took all the terms of single document to another to compute the similarity which is taken large amount time to finalize the process. So, cosine similarity gives more accurate result compared to Jaccard coefficient.

The researchers have found that most modest and well established measure to compute the similarity among the sentences is Cosine which is narrowly to human being and takes less time [6].

Give two documents $\overrightarrow{ta}$ and $\overrightarrow{tb}$, their cosine similarity is

$$SIM_C(\overrightarrow{ta}, \overrightarrow{tb}) = \frac{\overrightarrow{ta}.\overrightarrow{tb}}{|\overrightarrow{ta}| \; x \; |\overrightarrow{tb}|} \quad (2)$$

Where $\overrightarrow{ta}$ and $\overrightarrow{tb}$ are m-dimensional vectors over the term set T = {t1, . . . , tm}. Each dimension characterizes a term with its weight in the document, which is non-negative. Consequently, the cosine similarity is non-negative and enclosed between [0,1] [1].

4.3 Functional Model Design

Figure 2 illustrates a High-level functional model for how to calculate the similarity between each of Change Request C to each Systems Requirement R after passing through different activities in different Time T, when the total Model Running time equal the sum of $T_{1,2,3.}$

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

In order to calculate the similarity, there are some needed steps achieved through vocabulary analysis, which started by taking flow of words vector and convert them to tokens this step called Tokenize. In our model the statements will be broken to words while some characters like punctuation marks are discarded.

Frequently existing words like 'a', 'the', 'of', etc., will affect the similarity measures. Those words recognized as Stop Words, therefore they filtered out beforehand the computation using known stop word list.

Since the work could have different case, lowercase transformation is placed to change all tokens case to a lowercase such as 'SYSTEM", "System", or "system" are changed to "system".

In order to find words that repeatedly follow one another we used N-grams operator to extract sequences of words from a text. N-grams can be separated into two groups: 1) Character based and 2) Word based. The main inspiration behind this approach is that similar words will have a high part of N-grams in common, For example "business" and "strategy" are different ideas, Statistical based text processing will not obtain the context of these words, but it will tell you how many times strategy and business appear in the documents or data. So, N-grams makes a new attribute *business_strategy* and this leads to educate the model the context in which business strategy is related [7].

To figure out the words those have same synonyms for that purpose we used (Wordnet Dictionary). For example, same functions are written differently like "Sign in" and "Log in" while they different words but lead to the same function or meaning those should be considered equal.

Final step before calculating the similarity, is finding the words that are commonly written differently but leads to the same form. Therefore, they should be reduced to their ground form to be automatically matched, this technique called "Stemming" which produce a stem of word. For example both the words 'replace' and 'replacement' may outcome in the stem 'replac' and thus the words would be treated as equivalent, in this model we used "Stem (WordNet)" [3].

5. MODEL IMPLEMENTATION

We have implemented our model on RapidMiner, it's a data science application for machine learning, data science, text mining, predictive analytics and business analytics [8]. Each function in the suggested model is represented in one or more task in the tool.

In order to examine the possible benefits of the model, we have implemented the similarity measure on a real industrial requirements system for around 4000 Requirements Statements and submitted 15 change requests in same time which are hypothetical but validated with experts, this number may be more or less at certain point of time. The measures were utilized to see if automated similarity model can correctly detect if a specific change has an impact on the already defined set of requirements.

5.1 Data Collection – Industrial Case

It's based on a public Functional requirements document for ERP ecosystem [9]. Table 1 shows a summary for the number of systems requirements, minimum, maximum number of distinct tokens and the numbers of change requests

Table 1: Case Study used in the Model Evaluation

# of Requirements	Min # of distinct Tokens per Req	Max # of distinct Tokens per Req	# Change Requests
4000	6	150	15

After that we applied the following change requests in Table 2 :

Journal of Theoretical and Applied Information Technology
15th July 2020. Vol.98. No 13

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 2: Changes Requested on the System Requirements

Change Request #	Change	Number of Tokens
1	The **system** shall **support tracking** the **inventory** through **mobile application**	6
2	The **System** shall **support Weekly physical counts daily** with **update** of **balance file**.	9
3	The **system** shall **provide** the **ability** to **support** the following **item/inventory attribute Commercial cost**	9
4	The **system** shall **provide** the **ability** to **support data collection** via **bar code** and **QR Code hardware**	10
5	The **System** shall **inform** the **customer** by the **available quantity** whenever a **quotation** is **requested**	7
6	The **user** shall **able** to **simulate** a **shipment** or **consignment** before **actual receiving**	7
7	The **Landed cost** of the **received goods** shall be **calculated** in **Activity based costing**	8
8	The **system** shall **provide** the **ability** to **support kilogram** and **meters** as **units** of **measure**.	8
9	Whenever there is a **stock count** the **system** should **block** all **stock transactions**	6
10	The **System** shall **support creating Picking List Manually** and **Automatically**	7
11	The **System** shall **provide** the **ability** to **upload vendor bank details**	7
12	The **AP team** shall be able to **issue payment instructions** to the **banks**	6
13	The **Sales Admin** shall be able to **print customer invoice using QR Code**	8
14	The **Stock keeper** should not allow any **invoice** without a **QR code**	5
15	The **finance department** shall **post invoice** to **finance** with its **customer code**	7

5.2 Data Analysis and Results

The Model shows in Results Section how each change is interfered with each individual requirement represented in the calculated similarity distance. It's noticed that lower calculated measure is 0 which means that there is no impact of a change on a requirement while the maximum value is 1, conceptually this value calculated depending on the similarity between each Change's tokens vector to each original requirement's tokens vector with the following ripple effects resulted from the changes in form of count of affected Modules and Requirements in table 4

Focusing on a single change (C1) as an example as illustrated on figure 3, the impact of that change is 0.54 on R0014, that means the impact is strong, while the impact on R0087 is 0.14 which is very weak, but depends on the Requirement Engineer analysis and judgment, this change may don't have any clear impact but it could have any hidden impact that should be considered in the further human analysis

In Figure 4 shows the average similarity per each change that can assist measuring the efforts needed to apply a single change or plan for the applications versions and releases. Since the impact of C4 is much higher than the rest, while impact of C13 is less high which can be developed with the lowest risk.

In figure 2 the similarity frequencies are grouped in 10 range clusters started from (0.01 - .09) ended by (0.90 – 1.00) to observe the distribution of similarities, and it's noticed that the most of similarities frequencies fall between 0.10 till 0.50, it might be seen as the submitted changes don't have a high impact on the requirements, but on the other side it could be related to the semantics of the changes and how they are correlated with existing requirements.

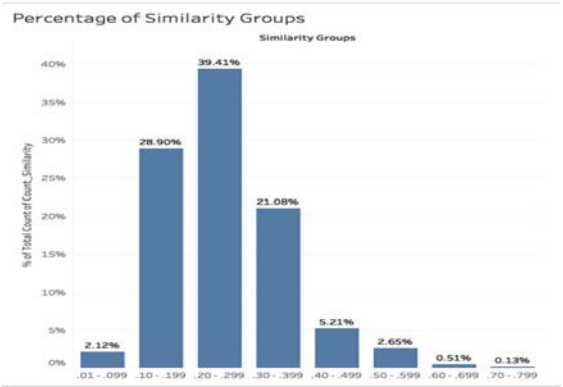

Figure 2: Similarity Frequencies Distribution

6. CONCLUSIONS

In this work, we proposed a model to measure the impact of single or many change requests or crowdsourcing ideas submitted in a free natural language against a large set of requirements. The key characteristics of the approach that is exploits the synonyms of text in which the user write the change free and the model measure the impact against the list of placed ULS requirements in the Traceability Matrix through NLP similarity models.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Automated similarity analysis used widely in many different business areas started by search engines, recommender systems and sentiment analysis and become an inspiring technique for assisting requirements engineers to manage requirements evolution since it's managed in a decentralized way by crowdsourcing stakeholders and development team in ecosystems.

In general, the model showed noticed results compared to human efforts needed to retrieve all the impacted requirements by a single change. Since there are no control over neither the structure of written textual requirement, the change requests, the formal semantics of requirements relation forms nor even the domain knowledge of the system. In addition, it can run over bulk of changes simultaneously with no barrier to the statements volume.

Less responsiveness has been directed to joining requirements with other requirements, we concentrated on how to link the change with all requirements, specifically, in case the ULS development units decided to change one or more in place requirements without raising change requests, the model won't be able to associate the change with the requirements

Finally, as it's unsupervised prediction model, consequently there is no predefined training data set to train the model so it's scaling up wherever needed.

7. FUTURE WORK

One of the main areas we'd work on is the linguistic ambiguity due to terminological expression differences that may occur among crowdsourcing stakeholders that belong to different business domains and cultures.

Handling the repeated requested changes with different format or syntax is a good field of study since we could have many different represented changes that belong on the same change topic that has to be considered

Linking the same requirements domain is one of the promising areas in order to enhance the model accuracy by utilizing the ontologies and semantics hierarchies

Finally, it's preferred to test the performance and the accuracy of this model in different business domains to have a valid assessment of the benefits and cost of such as a decision support system for requirements engineering change management

REFRENCES:

[1] H. Anna, "Similarity Measures for Text Document Clustering," 2008.

[2] W. H. Gomaa and F. Aly A, "A Survery of Text Similarity Approaches," International Journal of Computer Applications (0975-8887), vol. 68, no. 3, April 2013.

[3] N. p. Dag, J. Rengell, B. Carlshamre, P. Andresson and M. Karlsson, "A Feasibility Study of Automated Support for Similarity Analysis of Natural Language Requirements in Market-Driven Development," Requirements Engineering, vol. DOI: 10.1007/s007660200002, pp. 20-33, 2002.

[4] V. Thada and V. Jaglan, "Comparison of Jaccard,Dice, Cosine Similarity Coefficient to Find Best Fitness Value for Web Retreived Documents Using Genetic Algorithm," International Journal of Innovations in Eningeering and Technology (IJIET), August 2013.

[5] S. etal, "Impact of Similarity measures on web-page clustering," AAAI-2000: Workshop on Artificial Intelligence for Web Serach, July 2000.

[6] W. B. P. R. D. H. "TOWNE, "Measuring Similarity Similarly: LDA and Human Perception," Transactions on Intelligent Systems and Technology, Vols. ACM Trans. Intell. Syst. Technol. 7, 2, Article 25, p. 2, 2016.

[7] R. Community, "Generate n-grams Knowledge Base," RapidMiner, December 2016. [Online]. Available: https://community.rapidminer.com/discussion/35073/generate-n-grams-knowledge-base. [Accessed October 2019].

[8] R. Miner. [Online]. Available: www.rapidminer.com.

[9] "Functional Requirements for ERP Ecosystem (Cuyahoga County Government Functional Requirements)," [Online]. Available: http://it.cuyahogacountry.us. [Accessed October 2019].

[10] I. Sommerville, Software Engineering, 10th edition, Vols. ISBN-13: 978-013943030, Pearson, April 2015, p. 43.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[11] M.A.Torkamani, S.H.Ahmadi, A.Bayat, A.Bahrami, H.Bagheri and M.R.Khodabakhi, "An Approach for Configuration Management in Ultra Large Scales Systems," The International Journal of Software Computing and Software Engineering, vol. 3, no. 3, p. 463.

[12] P. Feier, R. P. Gabriel, J. Goodenough, R. Linger, T. Longstaff, R. Kazman, M. Kelien, L. Northrop, D. Schmidt, K. Sullivan and K. Wallnau, Ultra Large Scale Systems: The Software Challenge of the futhure. Technical Report, Software Engineering Institue, 2006.

[13] P. Feiler, R. Gabriel, J. Goodenough, R. Linger, T. Longstaff, R. Kazman, M. Klein, L. Northrop, D. Schmidt, K. Sullivan and K. Wallnau, Ultra-large-scale systems: The software challenge of the future. Technical report,, Software Engineering Institute, 2006.

[14] E. Nakagwa, R. Capilla, F. Diaz and F. Oquendo, "Towards the Dynamic Evolution of Context-based-Systems-of-Systems," p. 46, September 2014.

[15] S. Ian, C. Dave, C. Radu, K. Justin, K. Tim, K. Marta, M. John and P. Richard, "Large-Scale Complex IT Systems," Magzine Communications of the ACM, vol. 55, no. 7, pp. 71-77, 2012.

[16] C.Bogart, C.Kastner and J.Herbselb, "When it breaks, it breaks How Ecosystems developers reason about the stability of dependencies," in Preceedings of the ASE Workshop on Software Support for Collaborative and Global Software Engineering (SCGSE), Forthcoming, 2015.

[17] A. Elfaki, "Automated Verification of Variability Model Using First-Order Logic," in Managing Requirements Knowledge, Berlin, Springer, 013, p. 263.

[18] C. Arora, M. Sabetzadeh, A. Goknil and L. C. Briand, "Change Impact Analysis for Natural Language Requirements: An NLP Approach," in 23rd International Requirements Engineering Conference (RE), Ottawa, ON, 2015.

[19] A. Goknil, I. Kurtev, K. v. d. Berg and H. W. Veldhuis, "Semantics of trace relations in requirements models for consistency checking and inferencing," Software and Systems Modeling, p. 32, 2011.

[20] A. Goknil, K. Ivan, v. d. B. Klass and S. Wietze, "Change imapact analysis for requirements: A Metamodeling approach," IST, vol. 56, no. 8, 2014.

[21] M. A. Torkamani, H. Bagheri, A. Bahrami, A. Bayat, S. H. Ahmadi and M. R. Khodabakhshi, "An Approach for Configuration management in Ultra Large Scale systems," The International Journal of Soft Comput ing and Software Engineering [JSCSE],, vol. 3, no. 3, pp. 463-466, 2013.

[22] S. Xiaobing and L. Bixin, "Using Formal Concept Analysis to Support Change Analysis," ASE, 2011.

[23] S. Ghaisas and N. Ajmeri, "Ch 7, Knowledge-Assisted Ontology Requirements Evolution," in Managing Requirements Knowledge, Springer, 2013.

[24] A. Elvira-Maria, A. Aposolos, C. Alexander and A. Paris, "Introducing a Ripple Effect Measure: A Theoretical and Empirical Validation," in ACM/IEEE International Sympsium on Emperical Software Engineering and Measurement (ESEM), 2015.

[25] W. Spijkerman, Tool Support for Change Impact Analysis in Requirement Models, October 2010, pp. 9-10.

[26] W. Roel, "An Introduction to Requirements Tracebility," November 1995.

Journal of Theoretical and Applied Information Technology
15th July 2020. Vol.98. No 13

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 3 Summary of Directions Associated with our Research existing in Different Papers

Ref	System Size			Implementation Layer			Working Team		Source of Change		NLP				Requirement Change	
	Medium	Large	ULS	Coding	Design	RE	Centralized	Decentralized	Internal RE Workers	Distributed Teams	Tokenization	Stemming	Corpora Based	Similarity	Existing Requirement Change	New Change Request
[9]	✓	✗	✗	✗	✗	✓	✓	✗	✓	✗	✓	✓	✓	✓	✓	✗
[10]	✓	✓	✗	✗	✗	✓	✓	✗	✓	✗	✓	✓	✗	✓	✓	✗
[11]	✓	✗	✗	✗	✗	✓	✓	✗	✓	✗	✗	✗	✗	✗	✓	✗
[12]	✓	✗	✗	✗	✗	✓	✓	✗	✓	✗	✗	✗	✗	✗	✓	✗
[13]	✓	✓	✓	✓	✗	✗	✗	✓	✗	✓	✗	✗	✗	✗	✓	✗
Proposed Model	✓	✓	✓	✗	✗	✓	✗	✓	✗	✓	✓	✓	✓	✓	✓	✓

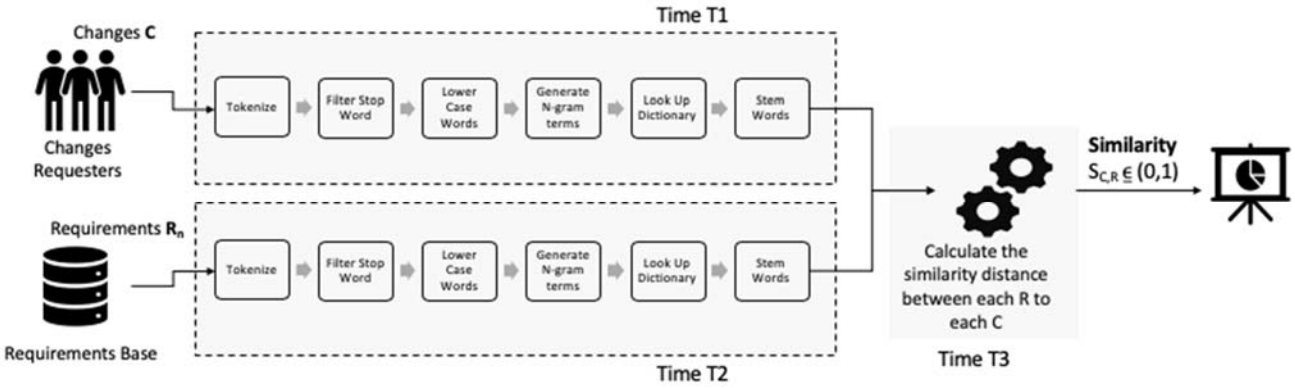

Figure 3: Similarity Model between Change (C) and Requirement (R) producing S similarity value between 0 and 1

Table 4: Impact of each single change on the whole list of modules and requirements

Change Num	Change_Text	Distinct count of Module	Distinct count of Req_Seq
C1	The system shall support tracking the inventory through mobile application	21	754
C2	The System shall support Weekly physical counts daily with update of balance file.	21	688
C3	The system shall provide the ability to support the following item/inventory attribute C..	21	1,696
C4	The system shall provide the ability to support data collection via bar code and QR Code..	21	1,644
C5	The System shall inform the customer by the available quantity whenever a quotation i..	21	518
C6	The user shall able to simulate a shipment or consignment before actual receiving	21	341
C7	The Landed cost of the received goods shall be calculated in Activity based costing	21	438
C8	The system shall provide the ability to support kilogram and meters as units of measure.	21	1,569
C9	Whenever there is a stock count the system should block all stock transactions	21	442
C10	The System shall support creating Picking List Manually and Automatically	21	769
C11	The System shall provide the ability to upload vendor bank details	21	1,546
C12	The AP team shall be able to issue payment instructions to the banks	21	201
C13	The Sales Admin shall be able to print customer invoice using QR Code	21	413
C14	The Stock keeper should not allow any invooice without a QR code	20	303
C15	The finance department shall post invoice to finance with its customer code	21	338

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Requirement Text and Change Num

Change Num	Change_Text	Requir..	Req_Text	
C1	The system shall support tracking the inventory through mobile application	R0014	System supports electronic workflow throughout all suites/modules/applications.	0.534522484
		R0051	Supports Mobile Devices	0.534522484
		R0070	The system should track system uptime and transaction response times in order to d..	0.507092553
		R0006	Provides application and system performance measurement reporting and ability to ..	0.377964473
		R0011	On-line training and demo module included with application software.	0.377964473
		R0020	Supports electronic signatures	0.377964473
		R0029	Supports single sign-on	0.377964473
		R0034	Supports group level permissions	0.377964473
		R0038	System security comprehensively includes other standard elements (e.g., Secure DM..	0.377964473
		R0041	Report manager to track multiple versions of reports	0.377964473
		R0061	The system's access requirements through firewalls and proxy must be clearly identi..	0.377964473
		R0062	Supports Fiber Channel SAN Technology.	0.377964473
		R0065	Contains toolsets to accommodate Automated tools to apply system-wide release up..	0.377964473
		R0074	The system's on-line help should be context sensitive.	0.377964473
		R0075	Provide a complete, up-to-date support site that contains a knowledge base where u..	0.377964473
		R0080	Support Windows AD Authentication and LDAP directories with the functionality to s..	0.377964473
		R0086	System error messages should appear in a consistent format across all system modul..	0.377964473
		R0097	Provides the ability to define system key words to fit current business practices and ..	0.308606700
		R0004	Security is required for each application with the ability to restrict levels of access by..	0.267261242
		R0033	System Administrator can view list of logged-in users	0.267261242
		R0066	The system should come with a comprehensive data dictionary for all system data fie..	0.267261242
		R0071	Provides automated transaction back-out/roll-back should the system fail during tra..	0.267261242
		R0099	The system should present data in a graphical manner, including bar charts, line char..	0.267261242
		R0008	Provides field level edits to ensure validity of the data being entered into the system	0.218217890
		R0012	Ad-hoc report writer packaged with application software that provides easy interfac..	0.218217890
		R0019	Ability to link to files located in a document management system (e.g. SharePoint, O..	0.218217890
		R0030	Supports different security structure for internal county users verses internet based..	0.218217890
		R0055	Ability to support Disaster Recovery and Contingency Plans defined.	0.218217890
		R0060	If hosting is proposed, the ability to support the service level agreement metrics.	0.218217890
		R0069	The system should not use any proprietary fundamental components (e.g.: vendor's ..	0.218217890
		R0073	Provide a robust on-line help available at the system, function, screen, and field level..	0.218217890
		R0078	Supports Microsoft Internet Explorer version 9 or higher lastest version of Google, F..	0.218217890
		R0079	Supports data-transfer via flat files (e.g., ASCII, variable and/or fixed length, comma-..	0.218217890
		R0081	Has the ability to interface or integrate with the County email system (Microsoft Exc..	0.218217890
		R0093	Provides simultaneous access to data by multiple users, without degradation to the ..	0.218217890
		R0095	Provides efficient application level load-balancing functionality.	0.218217890
		R0076	The system's error messages should be integrated with online help functions, allowi..	0.188982237
		R0013	Provide GUI-based end-user report viewing and query tools (within application)	0.169030851
		R0044	Ability to support multiple report generation schedules (daily, weekly, etc).	0.169030851
		R0059	Ability to operate on your proposed hardware architecture with all web applications ..	0.154303350
		R0023	Supports role-based workflow levels (different user, same security level) to execute ..	0.142857143
		R0087	Allows data entry staff to have a choice of updating the system through use of a grid/..	0.142857143

Figure 4: Sample of Impacted Requirements by a single submitted Changes represented by Similarity Distance

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

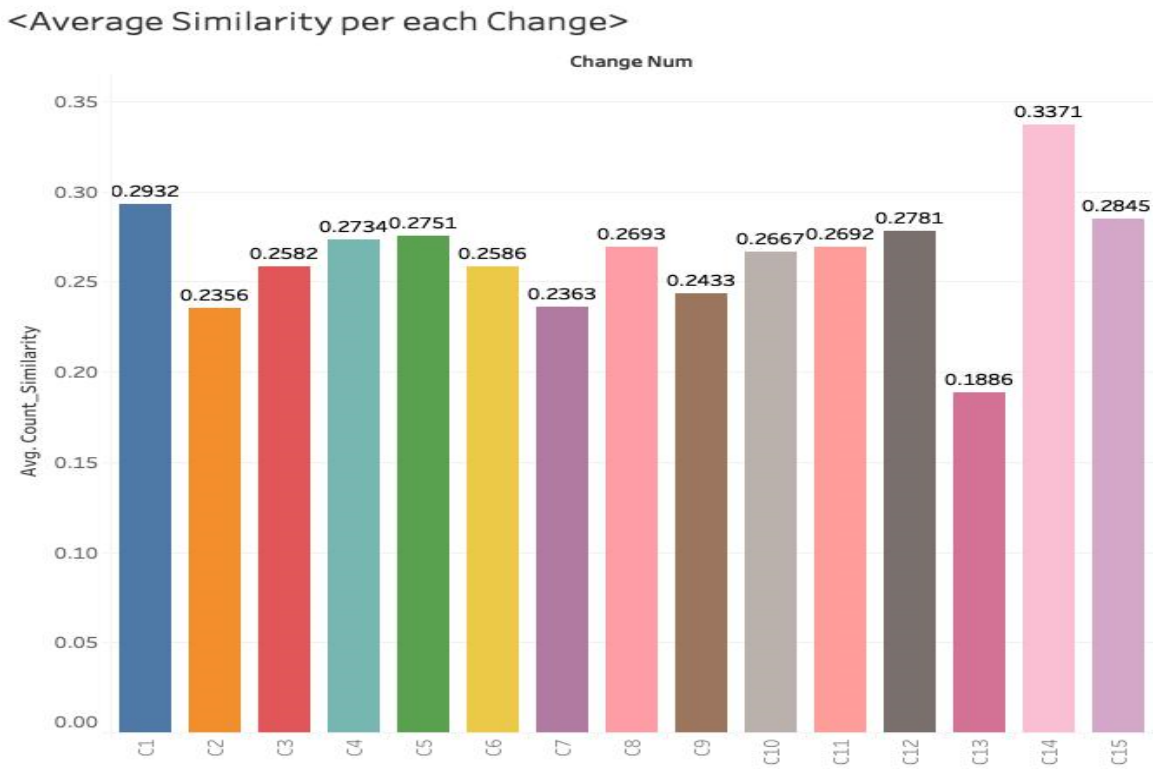

Figure 5: Average Similarity per each Change