A Process for Requirement Reliability in Goal Oriented Development by Enhancing (i*) Framework

Israa Abdulrauof Othman
Faculty of Computer Science and IT
Al-zaiem Al-azhari University

Sallam Osman Fageeri
Faculty of Computer Science and IT
Al-zaiem Al-azhari University

Talab Alsamani A.
Faculty of Computer Science and IT
Neelain University

ABSTRACT

This paper proposes a process for requirement reliability in goal oriented development by enhancing the (i*) framework, which is one of the main GORE techniques (KAOS, NFR, i*), it used in the first phase of the system development life cycle, in the requirements engineering phase. The (i*) framework relies mainly on social modeling that come to replace (KAOS) and this feature is distinguished from the rest of the GORE techniques, although it lacks an important feature in KAOS. The study aims to add that feature in the model in order to increasing the efficiency, thus develop more powerful and reliable software systems. We developed an enhanced (i*) framework by adding a layer to deal with the obstacles by finding a set of alternative solutions. The developed model was applied to a set of (Google DOCS) properties as a case study. The results evaluated using DESMET methodology reveal that and enhanced (i*) framework outperform (i*) framework in awareness representation feature giving 5 marks compared to the (i*) which is giving -3.

Keywords

Requirement Engineering, Requirement elicitation, GORE, KAOS, NFR, i*

1. INTRODUCTION

1.1 Requierment Engineering

RE can simply be defined as identifying a problem context, locating the customer's requirements within that context and delivering a specification that meets customer needs within that context. There are many requirements, methodologies that purport to do this, for example, soft systems methodology [1], scenario analysis [2], and UML [3]. Sometimes they work, sometimes they do not [4]. "If a company wishes to let a contract for a large software development project, it must define its needs in a sufficiently abstract way that a solution is not pre-defined. The requirements must be written so that several contractors can bid for the contract, offering, perhaps, different ways of meeting the client organization's needs. Once a contract has been awarded, the contractor must write a system definition for the client in more detail so that the client understands and can validate what the software will do. Both of these documents may be called the requirements document for the system."[5]. There are four requirement engineering Process include:

1. Requirements elicitation
2. Requirements analysis
3. Requirements validation
4. Requirements management

1.2 Overview of goal Oriented Requirements Engineering

Goal Oriented requirement engineering (GORE) provides an incremental approach for elicitation, specification, elaboration with refinement, analysis and modeling of software system requirements. GORE techniques are based on certain principles and concepts. There are many Various Goal Oriented Requirement Engineering (GORE) methods exist for these requirements engineering processes like GBRM, KAOS, TROPS etc. [6]. This paper focuses on the KAOS, NFR, I* (GORE) methods, and before introducing the problem that motivate us for this study, some questions will be given as an example:

If you are a developer have you ever dealt with one of GORE techniques before? If you are in another field, Have you ever heard about and be motivated to try one of GORE techniques before?

The requirements defined for a system should be: correct, consistent, verifiable and traceable. Requirements engineering is the process of eliciting, understanding, specifying and validating customers' and users' requirements. It also identifies the technological restrictions under which the application should be constructed and run [7]. It is an iterative and cooperative process with the objective to analyze the problem, to document the results in a variety of formats and evaluate the precision of the results produced [8]. Whenever a software application is built, the development team has to acquire certain knowledge about the problem domain and the application's requirements. The elicitation and specification of these requirements is a complex process as it is necessary to identify the functionality that the system has to fulfill in order to satisfy the users' and customers' needs [9]. There is a lack of a standardized process supporting requirements, handling and guaranteeing the quality of the results.

For any project, or sub-section of a project, requirement engineering defines what will be built. Many problems found during design, testing, or operation of a system is the result of incorrect, incomplete, or missing requirements. Therefore, verifying that the requirements are right is an important function of the assurance engineer.

1.3 Proplem Statment

Dealing with obstacles is very important for safety-critical systems: it allows analysts to identify and address exceptional circumstances at the requirements engineering time (instead of at programming or running time) in order to produce for instance robust requirements or new requirements to avoid or reduce impacts of obstacles. The result will be a more reliable software. We found that (i*) framework and many other requirements engineering tools do not support this important feature.

The rest of this paper is organized as follows: Part two clarifies our objective, part three discusses the related work by giving overview of the similar work in the same field, Part four reflects the methodology by specifying the methods applied in the work and details explanation about how the research process was carried out, in Part five result and discussion is presented, Part six includes paper recommendations, Part seven outlines the conclusion and future work.

2. OBJECTIVES

1- To support reliability in goal oriented development by making an enhancement in (i*) framework.

2- To Deploy the enhanced (i*) framework over a case study to assess its efficiency in achieving requirement engineering goals.

3-To Validate the enhancement in (i*) framework.

3. RELATED WORK

In [10] the researcher outlines a process that can be used to extend Tropos to address requirements traceability. He intends to develop a complete and usable requirement traceability process for Tropos which is (i*) tool, aiming to ensure the quality improvement of both the methodology and the software developed with it. This proposal outlines a process to help the software engineer in order to structure the necessary information to perform traceability in a specific project using the Tropos methodology. In our opinion the enhancement will be more efficient when we do it in the Meta model, so our proposal is making enhancement in the core (i*) framework rather than (i*) tools, to address requirement reliability. We have added a layer to deal with obstacles that will help to develop more robust system

In [11] the researchers proposed a framework which aims at supporting the design of sociotechnical systems, especially the design of a network of inter-actor dependencies intended to full fill a set of initial goals. The support comes in the form of a structured process and a tool that is founded on AI (Artificial Intelligence) planner which generates and evaluates alternative assignments of actor dependencies to identify an optimal or good enough design. They explored a range of measures for evaluating optimality, inspired by AI planning, multi-agent systems, social networks and Economics. They reported on the application of the framework to the secure domain systems design, safety critical systems, as well as to the problem of instantiation of STS designs. They were also experimenting with their prototype tool to evaluate its scalability to the real design problems.

Both our study and this study adopted improvement at the meta-modal level, we add an additional obstacles layer to support requirement reliability

4. METHODOLOGY

After careful study of more than 200 references on Gore techniques, we found that a number of previous studies have compared Gore's core techniques (KAOS,NFR,i*). These studies have indicated that: (i*) is the best one, the main reason for its favor is that it supports the social modeling .The central premise of(i*)is that to arrive at system requirements, that is, to conceive what system to build, one should examine and understand the relationships among social actors. A system aims to improve the relationship that some actors have with other actors. Rather than focusing on the behavioral properties of software, as in a mechanistic system, we should raise the level of abstraction and ask how the system will advance the relationships that some actors have in relation to other actors. then we will re-visit the (i*) Mata Model, specifically the goal task element layer and we will add the new layer:" dealing with the obstacles", this layer include three phases. We call it : obs-i* framework ,then We will choose the Google docs application to implement (i*) on two case studies, one of them using the original(i*)framework and the other using obs-i* framework (after enhancement),finally we Evaluate the New Framework Using DESMENT Methodology.

4.1 The Original (obs-i*) Framework – Enhancement Phase

Before illustrating our enhancement, figure1 shows the original i* meta model, and i* revisiting sub phase also shown in figure 2,then figure 3 shows the sub phase added to the new layer.

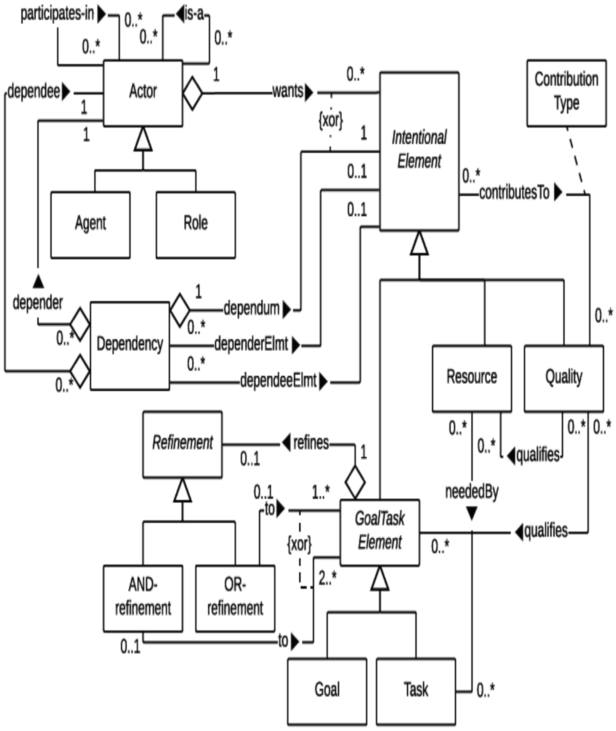


Figure 1. I* Meta model

The previous figure explains the concepts and relationships in the (i*) meta model,

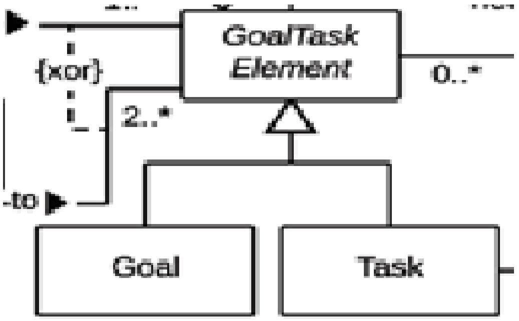


Figure 2. i* revisiting sub phase

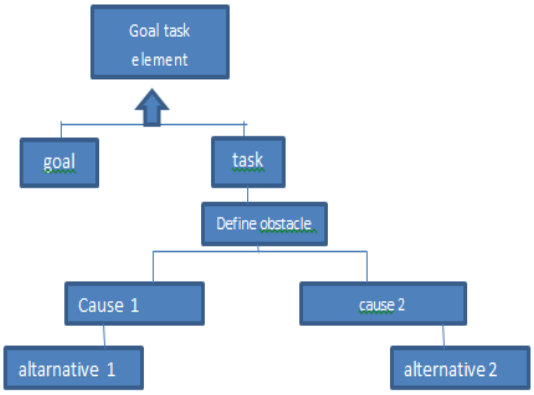


Figure 3. Dealing with obstacle layer

4.2 The obstacle layer

The main contribution of this study is to define a process to the following an enhanced (i*) phase: early requirements engineering

In this section we sketch a layer which includes three sub-layers as follows:

1. *Obstacle Gathering (OG)*: we identify the obstacle that can happen and make some failure to achieve (goal/requirement/task) to be achieved.

2. *Causes definition (CD)*: then we identify all the action that can be causes of the obstacle defined in the (OG) layer.

3. *Definition of the Alternative solutions (AS)*: Last but not least, we define the proposed alternative (task /requirement) to dealing with the obstacle causes which defined in the (CD) layer.

4.3 Case Studies

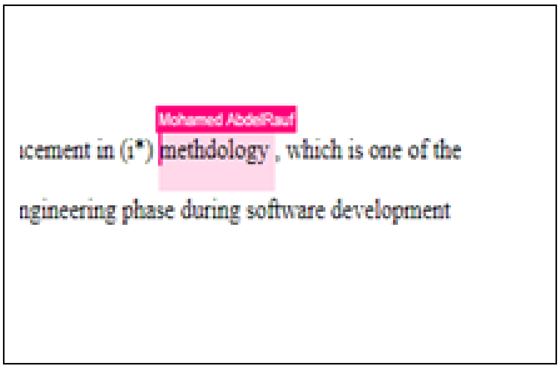
As case studies to assess how does (i*) and enhanced (i*) approaches perform for collaborative system, Google Docs [12] has been used from now in this thesis. Google Docs is a Web-based word processor, free, spreadsheet, presentation and form editor whose data storage service is provided by Google. Google Docs serves as a collaborative tool for editing documents so that they can be opened, shared, and edited by multiple users at the same time.

We selected this system for our analysis because it is widely known and it features a clear collaborative focus as its main goal.

As a starting point for our evaluation of the requirements techniques, we identified those design solutions for awareness requirements in Google Docs from the set of techniques which is proposed by Gutwin [13]. These techniques, which are commented in the following Subsections, can be also found as patterns for user collaboration in [14].

4.3.1 Remote Cursors

Based on Gutwin's telepointer [13], this technique was built, it allowed us to be aware of the other user's cursor position and if they have selected a text fragment or not. Therefore, when a remote user is writing other user can notice it in real time close to the curser‘s the users nickname appears overlapping with the text , Moreover if the users select some text it is highlighted by making it with the user's color.



Mohamed AbdelRauf

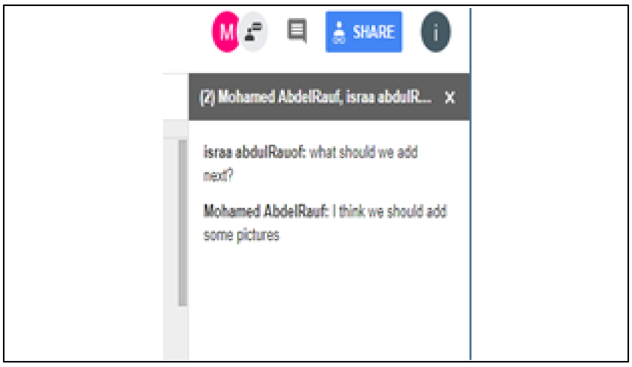
icement in (i*) methdology , which is one of the

ngineering phase during software development

Figure 5. Remote curser and remotely selected text fragment

4.3.2 Participant List and chat

Google docs show a list of participants that are editing the same document at the same time, by using this list users can communicate with each other by using a chat, which can be hidden or shown at any time, also by using this chat, users can notice the color assigned to each one of their collaborators.



(2) Mohamed AbdelRauf, israa abdulR... X

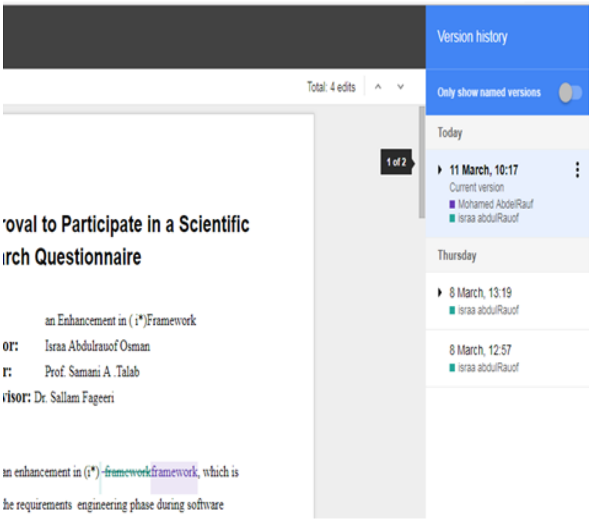
israa abdulRauof: what should we add next?

Mohamed AbdelRauf: I think we should add some pictures

Figure 5. Two uses chatting throw the participant list

4.3.3 Revision History

The techniques identified by Gutwin expressing information about authorship about the past [13] are used to make available to the user the history of changes done. They have been implemented by Google docs through using a revision history, it allows the system to keep track of all changes made by the users to the different type of editions done on the document. This revision history provides a means for users to review the changes made to the documents .in this revision history the changes made by each user were indicated by using a different color, more over the text will be also observed through style when the change made is a deletion, at any time This functionality can be activated or deactivated. This revision history has two levels of details depending on the size of showing information. Between these two levels of detail the user can switch at any time.



roval to Participate in a Scientific
rch Questionnaire

an Enhancement in (i*)Framework
or: Israa Abdulrauof Osman
r: Prof. Samani A .Talab
visor: Dr. Sallam Fageeri

an enhancement in (i*) ~~framework~~framework, which is
he requirements engineering phase during software

Figure 5. Revision history showing text elimination

4.4 Empirical Evaluations

To evaluate an enhanced (i*) approach compared to the original (i*), each one of the above mentioned awareness features is modeled in the following by using the different techniques. Firstly, we have to describe what Google Docs features can be modeled by using functional or non-functional requirements. The telepointer and avatar techniques result in NFRs because they contribute to increase some operability, such as helpfulness and ease of use. However, the third feature (Expressing information about authorship / about the past), despite contributing positively to the above mentioned quality features, it should be considered functional, the reason is the historical information storage and the rollback function.

Moreover, we have also associated the awareness functionalities both with the three characteristics of the collaborating systems (communication, collaboration, and coordination) and, with the characteristics of the ISO/IEC 25010 [15]. This standard has been used to organize properly the specification of the system following the recommendations of Moreira et al. [16]. Next, the evaluation is presented following the order in which they occurred.

4.5 Modeling The Running Case Study

Gutwin presented a conceptual framework to establish what information makes up workspace awareness. This information is can be obtained by answering the questions "what, who, and, where". That is, when we work with other users in a shared physical space, we know what they are doing, who we are working with, when various events happen, where they are working, and how those events happen.

Table 1. Elements of Workspace Awareness

Category	**Element**	**Specific questions**
Who	Presence Identity Authorship	Is anyone in the workspace? Who is participating? Who is that? Who is doing that?
What	Action Intention Artifact	What are they doing? What goal is that action part of? What object are they working on?
Where	Location Gaze View Reach	Where are they working? Where are they looking? Where can they see? Where can they reach?

After analyzing the features of Google docs, and according to Gutwin's framework for collaborative systems, we have identified clearly the systems' FRs (Table 4 illustrates a partial description of the system). Next, as can be observed in Table 5, each awareness functionality feature discovered in the system has been related to some quality factors in the SQuaRE standard, in order to identify the NFRs of Google Docs. For the clarity, and understanding of the evaluation, some requirements of Google Docs are only described.

Table 2. Relation between quality factors and awareness functionalities

Category Element Functional Requirement	Category Element Functional Requirement	Category Element Functional Requirement
Who Presence Know who is participating	Who Presence Know who is participating	Who Presence Know who is participating
What	What	What
Where	Where	Where

Table 3. Relation between quality factors and awareness functionalities

Quality Factor Awareness Functionality	**Quality Factor Awareness Functionality**
Functional Suitability Revision History, Telepointers, Participant List	Functional Suitability Revision History, Telepointers, Participant List
Reliability Revision History	Reliability Revision History
Performance Efficiency Telepointers	Performance Efficiency Telepointers
Operability Telepointers, Participant List	Operability Telepointers, Participant List
Security Revision History	Security Revision History

4.5.1 (i) Framework.*

In order to carry out the specification of Google Docs, the (i*) notation was used. Using this notation, we specified, each one of the SQuaRE quality factors previously identified in Table 4, as root soft goals of the system, These soft goals were refined into other soft goals by selecting those SQuaRE quality factors more appropriate for the system. Each one of the awareness functionalities were specified as resources provided by the system that contribute positively to satisfy some of the softest goals, that is, some quality factors. However, it can be noticed that also some of them contribute negatively because of the constraints they impose. This is the case of remote cursors, because they increase the resource utilization, also, the ease of use depends, among other factors, on the user's experience with this kind of systems. Moreover, the three FR identified in Table 3 have been specified as goals of the system that have dependency relationships with the resources, also It has been specified how the awareness techniques contribute positively to the functional aspects of collaborative systems specified as tasks in the goal model.

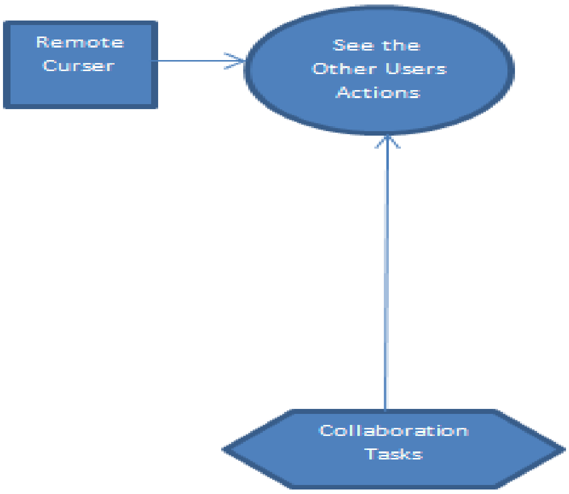


Figure 6. Strategic Rationale Model for (the Remote Curser Google Docs Feature) using i*

This is part of the requirements model for the remote feature in the Google Docs application as explained in the previous figure, the sub-task (see other user action) has been moved immediately to the collaboration task without addressing the obstacles that could happen and cause some failure in this task

4.5.2 An enhanced (i) model.*

The problem in i* is that we were not able to represent the obstacle (because this model doesn't contain a layer to dealing with obstacle), therefore the three general tasks of collaborative systems (collaboration, communication and coordination) cannot be defined. This lack of expressiveness led us to have an incomplete representation of a system's requirements, so that we have to extend this framework. In this model we have added an additional layer to dealing with obstacle with more than one alternative solution, so that it can be beneficial in order to minimize the risk.

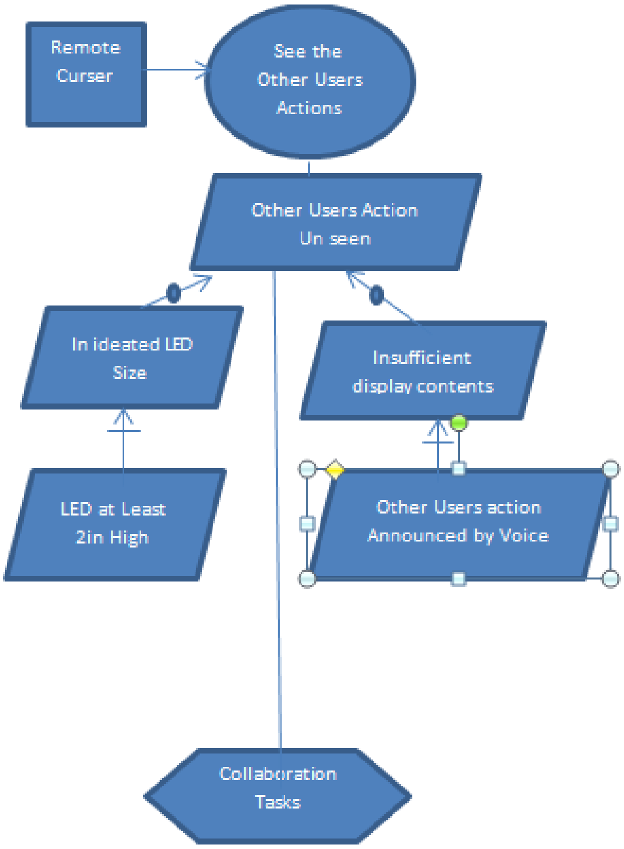

Remote Curser
See the Other Users Actions
Other Users Action Un seen
In ideated LED Size
Insufficient display contents
LED at Least 2in High
Other Users action Announced by Voice
Collaboration Tasks

Figure 7. Strategic Rationale Model for (the Remote Curser Google Docs Feature) Using an Enhanced "obs-i*"

This figure is part of a requirement model for the remote curser feature in the application of Google Docs using the enhanced (obs-i*) model As explained in the figure above, the first task is(remote curser), then a sub-task(see other user action), the next task considers that the user cannot see the other user action (unable to see other user action) , before finding solution to this obstacle we have to clarify the cases :

1-inideated LED size

2- Insufficient display contents, finally we moved to the next layer to clarify possible solutions in the sub tasks:

1-LED at least 2 in high

2- Other user's actions announced by voice

4.6 Evaluating original (i*) and an enhanced (i*) Approaches

Using as input the different specifications of the system, the evaluation of the different RE techniques was carried out by using DESMET [17]. It is a set of techniques applied to evaluate both Software Engineering tools and methods. We have used the method based on a qualitative case study that describes a feature-based evaluation. Following the guidelines of this technique, an initial list of features was prepared that a GO approach for collaborative systems should provide (see Table 6).

Table 4. List of Features for approaches evaluation.

Feature	**Description**
FR and NFR Representation	The model should be able to represent graphically FR and NFRs and differentiate them
Collaborative Systems Characteristics	The model has to represent the collaboration, communication and coordination characteristics
Awareness Representation	The model should allow one to represent the awareness characteristics of the system
Quality Factors Representation	The model must represent the SQuaRE characteristics and sub characteristics
Importance of Requirements	The model should represent the importance and preference between requirements
Hierarchical Representation	The relation between the model elements should be hierarchical
Model Complexity	The model complexity should not be too high
Quantitative Model	The model must allow one to quantify the relations between represented elements
Traceability	The represented requirements should be traceable throughout the software development process

Once Table 6 is filled in, DESMET establishes that an importance degree should be assigned to each feature. Table 6 has been filled in by using the following degrees: the degrees to apply are Mandatory (M), Desirable (D), Highly Desirable (HD), and Nice to have (N).

Table 5. Importance of the features

Feature	**Importance**
FR and NFR Representation (RR)	M
Collaborative Systems Characteristics (CSC)	M
Awareness Representation (AR)	M
Quality Factors Representation (QFR)	HD
Importance of Requirements (IR)	HD
Traceability (T)	HD
Quantitative Model (QM)	D
Hierarchical Representation (HR)	D
Model Complexity (MC)	N

Then, according to DESMET, a scale to evaluate each one of the description features should be provided. The scale proposed by DESMET (see Table 6) was applied to evaluate each feature, according to the following factors: Conformance scores obtained from candidate method (CSO) and Conformance Acceptability Threshold (CAT).

Table 6. Judgment scale to assess support for a feature

Generic scale point	**Definition of Scale point**	**Scale Point Mapping**
Makes things worse	Cause Confusion. The way the feature is represented makes difficult its modelling and/or encourage its incorrect use	-1
No support	Fails to recognize it. The approach are not able to model a certain feature	0
Little Support	The feature is supported indirectly, for example by the use of another model/approach in a non-standard combination	1
Some Support	The feature is explicitly in the feature list of the model. However, some aspects of the features used are not catered for.	2
Strong Support	The feature is explicitly in the feature list of the model. All aspects of the feature are covered but its use depends on the expertise of the user	3
Very strong Support	The feature is explicitly in the feature list of the model. All aspects of the feature are covered and the approach provides a guide to assist the user	4
Full support	The feature appears explicitly in the feature list of the model. All its aspects are covered and the approach provides a methodology to assist the user	5

5. RESULTS AND DISCUSSION

Once each feature was evaluated and discussed as shown previously the difference between CAT and CSO factors was calculated as shown in the column Difference (Dif) in Table 7. Then, we should highlight that a variation of the DESMET method was used. The importance (Imp) of each feature has been weighted on a scale from 1 to 4 (Nice to have – 1, Desirable – 2, Highly Desirable – 3, Mandatory – 4). The importance was used to calculate the final score of each feature by multiplying the Importance by the Difference. This computation is shown in the column Score (Sco) in Table 7. Finally, the last score of each technique (Total) was obtained by adding the scores of all the features. This framework has been used to evaluate all the different GO approaches studied.

Table 7. Results of approaches evaluation

	i* **framework**					**an enhanced (i*) framework**				
Feature	**Imp**	**CAT**	**CSO**	**Dif**	**Sco**	**Imp**	**CAT**	**CSO**	**Dif**	**Sco**
RR	4	5	5	0	0	4	5	5	0	0
CSC	4	4	5	1	4	4	4	5	1	4
AR	4	4	3	-1	-4	4	4	5	1	4
QFR	3	3	5	2	6	3	3	5	2	6
IR	3	3	0	-3	-9	3	3	0	-3	-9
T	3	3	3	0	0	3	3	3	0	0
QM	2	2	1	-1	-2	2	2	1	-1	-2
HR	2	2	3	1	2	2	2	3	1	2
MC	1	1	1	0	0	2	1	1	0	0
Total					**-3**					**5**

An enhanced (i*) outperforms the original (i*) with 8 marks in awareness representation (AR) which is an important feature (mandatory) as shown in table (7).

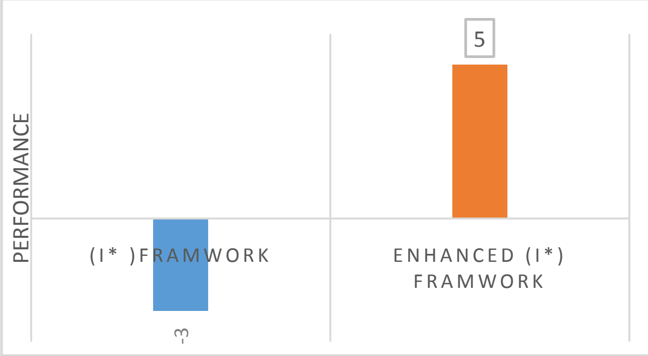


Fig. 8. Empirical analysis results

Fig. 8 shows graphically the scores obtained by each one of the approaches.

As can be observed, that an enhanced (i*) approach has the high score in (AR) than (i*) Moreover, as DESMET suggests, we have performed a comparative of the Percentage of each feature satisfied by each analyzed approach. Fig. 9 illustrates that both (i*) and an enhanced (i*) had the same score in all features except in awareness requirement (AR) and an enhanced (i*) has destination over (i*) because it had the highest score in this feature.

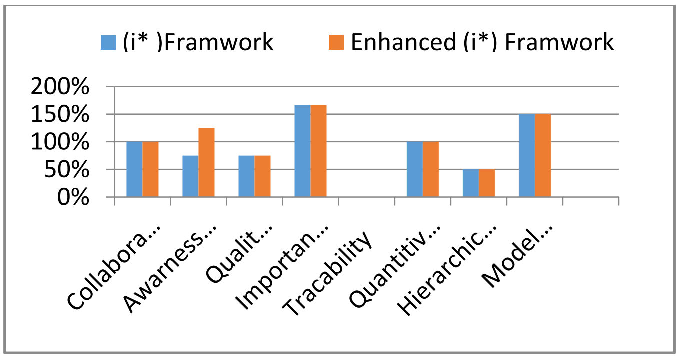


Fig. 9. Results relative to distinct features

When we apply an enhanced (i*) on Google Docs, the requirements model becomes more powerful because the model has an additional layer to dealing with obstacle as shown explicitly in the goal (see other user action) , also the alternative solutions to this obstacle was described ,that will increase the strength of the system.

As far as we know, this study has not been presented before, so we have found few related works. In the following paragraphs we will explain the differences between our study and the most two relevant related works we have selected:

1- Rosa Candida Pinto, Carla Silva and Jaelson Castro on the paper (A Process for Requirement Traceability in Agent

Oriented Development),aims to support traceability through requirements specifications, static and dynamic software design models, system architecture models, and implementation artifacts of agent-oriented software systems. But this thesis aims to support reliability through adding layer to dealing with obstacle.

In their work they have outlined a process that can be used to extend Tropos to support traceability. Our work is to extend (i*) meta model.

An e-commerce example is used to demonstrate the applicability of the proposed approach, some features of Google docs are used in our wok as a case study.

Our result is variant to their result in the degree of tractability, we think this variation accurse because they support” Tropos” tool but our enhancement support the Meta model.

2- “Volha Bryl”. In “Supporting the Design of Socio-technical Systems by Exploring and Evaluating Design Alternatives” Ph.D. Thesis, The researchers proposed a framework which aims at supporting the design of sociotechnical systems, specially the design of a network of inter-actor dependencies intended to fulfill a set of initial goals.

They validated the approach through case studies in a number of application domains and scalability experiments. But in our work we have applied the framework in Google docs, moreover we used DESMET methodology to evaluate the framework.

6. RECOMMENDATIONS

1-in our opinion, It is our responsibility as software engineering researchers to advise programmers and software developers to use our (obs-i*) framework that facilitates their work and make software systems more efficient and robust, especially in the collaborative system, because it's support social modeling

2- although we have proven in this study the quality of(obs-i*) for supporting system reliability ,but from our point of view there is An important issue to keep in mind that is if the system is simple and may not need a complicated way to extract the requirements model, it may take more time and effort than traditional methods.

So, the most important factor to choose the appropriate method to generate requirement model for a system is the size and complexity of the system.

7. CONCLUSION AND FUTURE WORK

Finally, through the case study and its application of both models before and after enhancement, it became clear that (obs-i*) outperform (i*) in the awareness representation feature, additionally we would like to stress the importance of the enhancement achieved by adding an additional layer to dealing with obstacles with three layers in the model.

From our point of view, there are many additional proposals that can help in supporting reliability in the software systems.

Our future works and way forward include the following interested points show as follows:

1-During our survey many references provided studies on (i*) framework and since it's an open source, the diversity of meta models for its-related notations and tools has arisen as a challenge and caused confusion for both researcher and tool developers . So, there must be a future work provides the standard unique method for it.

2-Making some effort to automated alternative selection in dealing with obstacle layer, using Aspects of artificial intelligent (AI).

3- Extend an enhanced (i*) framework in order to represent the requirements, importance, to giving support to determine which requirements are more important than others.

4- Apply an enhanced (i*) framework over more and more case studies that may lead to the discovery of some deficiencies and then treated accordingly.

8. REFERENCES

[1] Checkland, P., Systems Thinking, Systems Practice, Wiley Publications, 1981.

[2] Carroll, J. (ed.), Scenario-Based Design: Envisioning Work and Technology in System Development, Wiley Publications, 1995.

[3] Booch, G., Rumbaugh, J., Jacobson, The UML User Guide, Addison Wesley, 1999.

[4] A. van Lamsweerde. Goal-Oriented Requirements Engineering: A Guided Tour. Proc. 5th IEEE International Symposium on Requirements Engineering (RE'01), Toronto, Canada, August 2001

[5] Approach to Define a Non-Functional Requirements Elicitation Guide Using a Customer Language Andreia Silva, Placido Pinheiro, and Adriano Albuquerque Graduate Program in Applied Informatics University of Fortaleza (UNIFOR) Fortaleza, Brazil andrearsp@gmail.com, {placido,adrianoba}@unifor.br2016 DOI reference number: 10.18293/SEKE2016-195

[6] http://ieeexplore.ieee.org/document/4137410/ 20/9/2017 ,11:00 AM

[7] G.F. Hice, W.S. Turner, and L.F. Cashwell, System Development Methodology. North Holland, 1974.

[8] E. Munford, "Participative Systems Design: Structure and Method", Systems, Objectives, Solutions, Vol. 1, North-Holland, 1981, 5-19.

[9] Requirements Engineering for Web Applications – A Comparative Study ,M. JOSÉ ESCALONAUniversity of Seville. Spain, Journal of Web Engineering, Vol. 2, No.3 (2004) 193-212

[10] A Process for Requirement Traceability in Agent Oriented Development Rosa Candida Pinto, Carla Silva and Jaelson Castro

[11] Volha Bryl. Supporting the Design of Socio-technical Systems by Exploring and Evaluating Design Alternatives. University of Trento, 2009.

[12] Google, "Google Docs" (2001), http://docs.google.com11/9/2017 7:00 AM

[13] Gutwin, C., Greenberg, S., Roseman, M.: Workspace Awareness in Real-Time Distributed Groupware: Framework, Widgets, and Evaluation. In: HCI on People and Computers XI, pp. 281–298. Springer, Heidelberg (1996)

[14] Schümmer, T., Lukosch, S.: Patterns for Computer-Mediated Interaction. John Wiley & Sons Ltd. (2007)

[15] ISO/IEC 25010:2011, Systems and soft. engineering - Systems and soft. Quality Requirements and Evaluation (SQuaRE) - System and soft. quality models (2011)

[16] Moreira, A.M.D., Araújo, J., Rashid, A.: A Concern-Oriented Requirements Engineering Model. In: Pastor, Ó., Falcão e Cunha, J. (eds.) CAiSE 2005. LNCS, vol. 3520, pp. 293–308. Springer, Heidelberg (2005)

[17] Kitchenham, B.: A methodology for evaluating software engineering methods and tools. In: Experimental Software Engineering Issues: Critical Assessment and Future Directions, pp. 121–124. Springer, Berlin (1993)