Risk Analysis for Collaborative Systems during Requirements Engineering

Kirthy Kolluri, Robert Ahn, Lawrence Chung

Department of Computer Science
The University of Texas at Dallas
Richardson, TX, USA
{kirthy.kolluri, robert.sungsoo.ahn, chung}@utdallas.edu

Tom Hill
Fellows Consulting Group
Dallas, TX, USA
tom@fellowsconsultinggroup.com

***Abstract*- Risk, a potential occurrence of some undesirable event, can be dangerous if not adequately identified and dealt with early on during software development. However, identifying risks can be difficult, hence oftentimes resulting in a particular software system that is unable to address risks, especially critical ones adequately. This paper proposes an ontology-based framework for performing risk analysis with the Augmented Reference Model - The Reference Model augmented with risk analysis. The Reference Model emphasizes that the user requirements are met through the collaboration between the system and the events occurring in its environment - i.e., not by the system alone, hence the term "collaborative system." We also offer an activity-oriented ontology to carry out risk analysis by identifying risks from negating the events in the environment and system. Such negations of the requirements, specifications, and domain events generate a graph-like representation, called *Risk Analysis Graph (RAG)*, to help perform risk analysis. To validate our framework, we have performed two experiments using questionnaires to identify risks and use the risk analysis tool to generate RAG for performing risk analysis. We feel that at least these experiments show that RAG helps identify risks - especially the critical and uncommon ones that we would not have thought of.**

Keywords- ***Risk; Risk Analysis; Ontology; The Reference Model (WRSPM Model); Requirements Engineering***

I. INTRODUCTION

Risk, which is defined as a situation or event where something of human value (including humans themselves) has been put at stake and where the outcome is uncertain"[10], is a phenomenon faced or caused by the agent (e.g., User, Software or Hardware). If the requirements do not address critical risks as fundamental potential problems, the projected system may lead to grave consequences [1]. For instance, in building a smartphone app for helping blind people navigate indoors, it might not be too evident to requirements engineers that a blind person may not be able to walk straight in line or figure out where to turn. This is just one example of, among many such potential risks.

Risks involving the user and the system may arise due to the system malfunctioning or the user misusing the system. For example, a blind person has to walk ten steps before making a right turn. What if the smartphone application asks the blind person to turn earlier or later after walking ten steps? Or what if the user ignores the instructions and fails to turn at the right spot? Addressing these kinds of scenarios by the requirements engineers and the software developers before developing the application would help plan with risk minimization and mitigation strategies.

The Reference Model (WRSPM Model) [3] emphasizes that the user requirements are satisfied by the collaboration between the user and the events in its environment. Since it involves both the system and the user, the term collaborative system is used (e.g., a smartphone app, ***Theia***[1] for helping blind people navigate inside one of our campus buildings). Keeping Murphy's Law in mind which states, anything that can go wrong will [2], we perform risk analysis by extending the Reference Model that we adopted into the *Augmented Reference Model*.

Negating the events in the Reference model gives us the possible negative things (risks) that may arise in a particular environment. Using these possibilities, a graph-like structure called the *Risk Analysis Graph (RAG*) is generated. We use a highly activity-oriented ontology to identify the most important/critical risks obtained by the RAG in performing risk analysis. We have carried out experimentation in two parts and compared the total number of risks obtained/ignored by the students who performed both these experiments. Through this experimentation, we have observed that simple yet important risks, such as walking in a straight line, etc., can be overlooked.

DOI reference number: 10.18293/SEKE2021-192

[1] Theia is the Greek goddess of sight

Running example: An indoor navigation app (Theia) for helping blind people is used as the running example to illustrate the fundamental concepts of the risk analysis framework. For ease of understanding, we use the example of a blind person (Stevie) navigating indoors using the smartphone application (Theia). Stevie is a blind person (student) who wants to navigate in the campus building. He uses the smartphone application, Theia, to navigate from his current location

Section II describes the related work. Section III describes the proposed approach for performing risk analysis. Section IV describes the experimentation and the observations of the experimentation. Section V includes the overall observation and threats to validity. In the end, a summary of the paper is described, along with some future work in Section VI.

II. RELATED WORK

The Reference model draws attention to the vital concept of satisfying the user's requirements through the collaboration between the environment and the system through events. The environment comprises everything associated with the users (designators), the activities performed by the designators, surrounding infrastructure (e.g., buildings, things, etc.), and the environment events *(e)* are those that are associated with the environment. The system comprises the software system, the actions performed by the software system, and the programming concepts related to the software system. The system events *(s)* are those associated with the system. These events are classified as visible and hidden events – i.e., events visible and hidden to the environment and the system - (e_h, e_v) and (s_h, s_v) respectively [3, 4]. These environment events and the system events help satisfy the requirements.

In the area of Requirements Engineering, the Reference Model [3, 4] emphasizes collaboration and focuses on applying formal methods to the user requirements and reducing them to the system specification. We adopt and extend the Reference Model into the Augmented Reference Model to perform risk analysis in this work.

In the area of Risk Analysis, the work discussed in [1] proposes a Goal-Risk (GR) framework for modeling risks during the requirements engineering phase. They model goals, events, and treatments in three layers. The work discussed in [11] builds upon the framework proposed in [1] and provides multi-object optimization; hence more queries related to risk. Some similarities between our work and the work addressed in [1] are the risk analysis performed in the requirements engineering phase and an ontology provided, which analyzes risks. Our framework uses the Augmented Reference Model to perform risk analysis by negating the events (requirements, specification, and domain). The approach proposed in our paper aims to systematically obtain risks that can and cannot be obtained by logical negation.

CORAS [5] is a risk analysis framework that models, analyzes risks, and handles them. Each risk is analyzed in this framework by asking questions and prioritizing risks. Our framework provides an activity-oriented, risk-oriented ontology that addresses critical risks identified while performing risk analysis using the Reference Model and the Risk Analysis Graph. The work discussed in [6, 7, 8] explains obstacle analysis which explains decomposing the goals. They also provide a set of rules, including negation. There is some similarity in the approach, but we use only functional requirements in our work and use negation for obtaining risks.

The ontology of risk discussed in [9] is regarding its relationship with value, unlike our ontology, which is strongly tied to identifying risks that the agents face. We adopt the ontological components addressed in Requirements Modelling Language (RML) [12] and add another ontological concept, "Risk," to the existing work to tie the concept of Risk to Action and Agent.

III. A FRAMEWORK FOR PERFORMING RISK ANALYSIS

To help find and analyze risks, the risk analysis framework described in this paper uses an activity-oriented ontology. This process transforms the Reference Model into the Augmented Reference Model. The Risk Analysis Graph is generated by using negation which is explained in detail in the following steps. A tool to help generate a Risk Analysis Graph (RAG) was also developed.

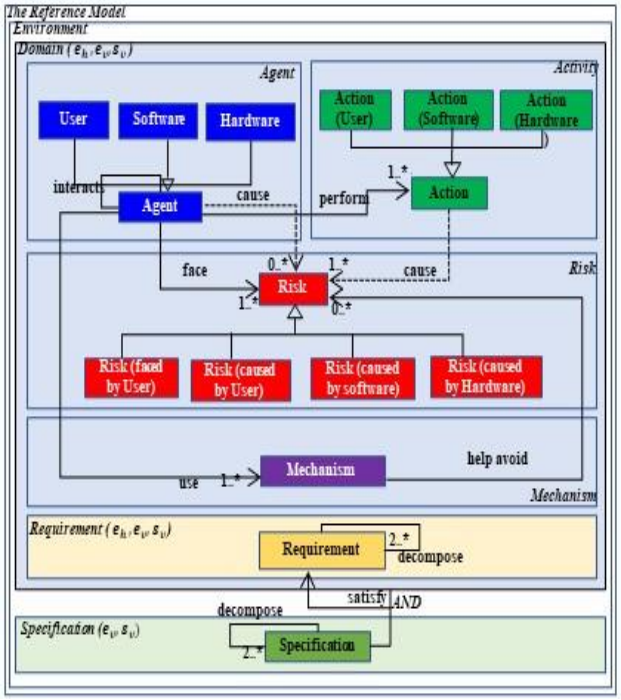

Figure 1. High-level ontology of the Risk Analysis Framework

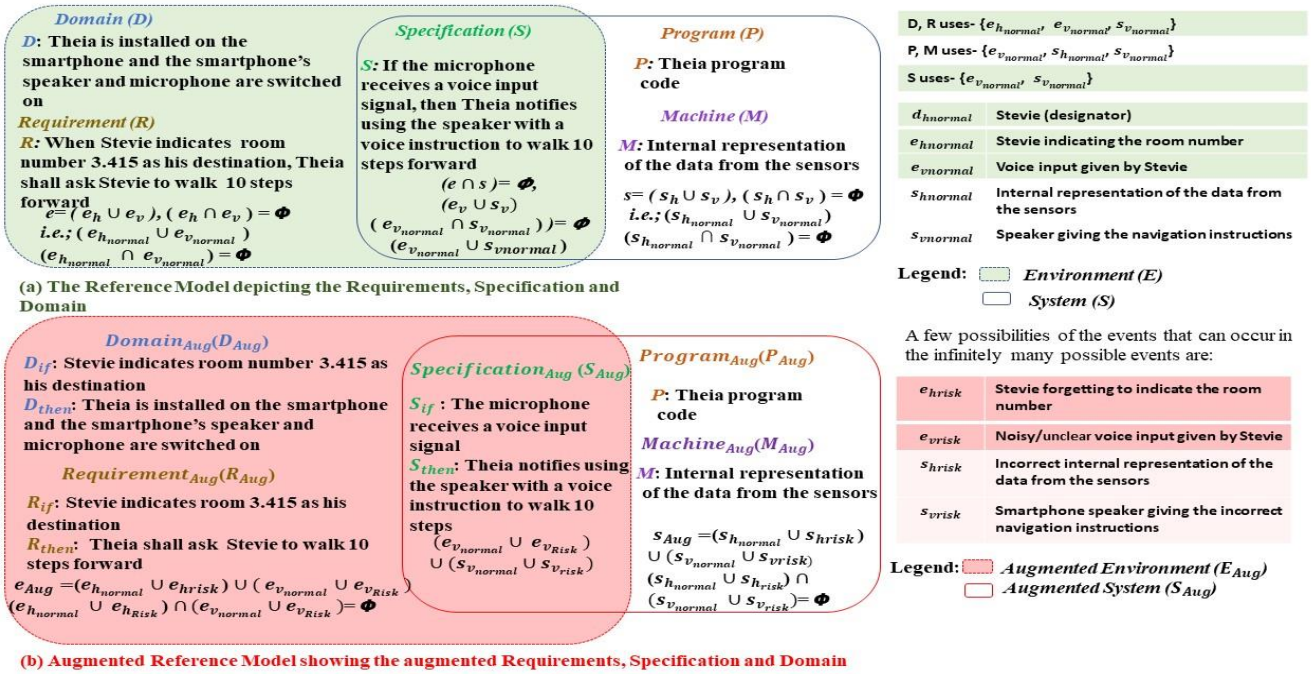

Figure 2. Transformation of the Reference Model into the Augmented Reference Model depicted using an instance-level example

A. Step 1: Obtain Overall Ontology:

It is essential to explicitly represent high-level concepts such as Agents, Risks, Actions, Requirements, Specification, Domain in a domain-independent approach to avoid omissions and commissions of risks while transforming the Reference Model into an augmented Reference Model and generating risks. Additionally, some concepts may be incorporated from a domain-dependent ontology as well. All the concepts and the relationships between them can be found in Fig. 1.

This ontology is independent of the domain and can be used for various domains which use any kind of collaborative system. In this step, we want to identify the domain-level concepts involved to help the requirements engineers/ developers to generate risks. This is an activity-oriented ontology that addresses risks associated with each activity performed by the Agent. The ontology is also used to identify the most critical risks obtained from the risk analysis outcome after step 5.

B. Step 2: Acquire and Decompose Requirements:

The proposed approach uses the functional requirements R from the Reference Model, represented in the form $i \rightarrow t$. This acquired requirement is AND-decomposed into sub-requirements: R_{if} and R_{then}. Requirements are decomposed to broaden the scope of the risk generation. Each of these sub-requirements can be further decomposed if there exists an $i \rightarrow t$ relation.

Instance-level requirements, specification, and domain were used throughout this paper for facilitating simplicity in understanding the risk analysis process.

For instance,

R: *When Stevie indicates his destination as room 3.415, Theia shall ask Stevie to walk 10 steps forward*

is AND-decomposed into

$\boldsymbol{R_{if}}$: *Stevie indicates his destination as room 3.415*

$\boldsymbol{R_{then}}$:*Theia shall ask Stevie to walk 10 steps forward*

C. Step 3: Generate Specification and Domain:

Using this proposed approach, it is possible to partially automate the specification and domain using the Ontology-based approach, which is discussed further in Step 3. As shown in the Reference Model, since every requirement has a specification and domain, all sub-requirements have sub-specifications and sub-domains, respectively. The specification and domain can be further decomposed if it is of the form $i \rightarrow t$ or if an *(AND)* or *(OR)* or *(,)* or *(.)* are present. For instance, after decomposing R into R_{if} and R_{then}, we obtain the S_{if}, S_{then}, D_{if} and D_{then} respectively. Considering the sub-requirement R_{then} (due to space limitation), we obtain

$\boldsymbol{D_{then}}$**:** *The smartphone's speaker is switched on and is in working condition*

$\boldsymbol{S_{then}}$**:** *If the microphone receives a voice input signal, Theia notifies using the speaker with a voice instruction to walk 10 steps forward*

The events associated with D_{then} and S_{then} are checked for further refinements and are decomposed based on the satisfaction of the criteria. Since S_{then} is in the form $i \rightarrow t$, S_{then} is decomposed into S_{then_if} and S_{then_then}.

$\boldsymbol{S_{then\_if}}$**:** *The microphone receives a voice input signal*

S_{then_then}: *Theia notifies using the speaker with a voice instruction to walk 10 steps forward*

D. Step 4: Perform Augmentation

When considering the Reference Model and transforming the equations from the Reference Model, the phenomenon (**Φ**) which takes place is a union of the environment events '*e* 'and the system events ' *s'* [3] . Hence,

$$\Phi = e \cup s \quad (1)$$

$$e = e_h \cup e_v, \qquad e_h \cap e_v = \phi \quad (2)$$

$$s = s_h \cup s_v, \qquad s_h \cap s_v = \phi \quad (3)$$

there are four events associated with it (normal case), i.e., e_h, e_v, s_h, s_v [3, 4]. In this piece of work, we call them normal case events, hence represented by the notation $e_{h_{normal}}$, $e_{v_{normal}}$, $s_{h_{normal}}$, $s_{v_{normal}}$. Augmenting the Reference Model is about adding risks (negating) to the normal events. The events associated with risks are e_{hrisk}, e_{vrisk}, s_{hrisk}, s_{vrisk}. Therefore, in the Augmented Reference Model we have eight events associated with it, both the normal case events and risk events namely. By substituting the normal and risk case events we get,

$$e_{h_{Aug}} = e_{h_{normal}} \cup e_{h_{risk}} \quad (4)$$

$$e_{v_{Aug}} = e_{v_{normal}} \cup e_{v_{risk}} \quad (5)$$

Similarly, the system events can be obtained as shown in equations 4 and 5. Transformation of the environment and the system events into negated events is done by substituting in equation 2.,

$$e_{Aug} = (e_{h_{normal}} \cup e_{h_{risk}}) \cup (e_{v_{normal}} \cup e_{v_{risk}}) \quad (6)$$

$$(e_{h_{normal}} \cup e_{h_{risk}}) \cap (e_{v_{normal}} \cup e_{v_{risk}}) = \boldsymbol{\phi} \quad (7)$$

Similarly, for transforming the system events, we substitute the normal and risk cases in equation 3 which is not shown here due to space limitation. The augmentation process is shown in Fig. 2. with a detailed instance-level example.

E. Step 5: Obtain Risks by generating the Risk analysis Graph (RAG)

In this paper, we propose the generation of

Risk Analysis Graph (RAG), shown in Fig. 3, by the systematic generation of risks that are hard to find. For this systematic generation of risks, we perform logical negation of AND, OR, Implication, etc., as the starting point. Due to the space limitation, we will present the process of obtaining risks by negating the logical implication *(i → t)* and another particular case *(¬ i Λ t)*. For this work, we have implications in the requirements, specification, and domain as well. We have worked on all possible combinations to obtain various risks from a set of requirements. However, we will be illustrating only the implications associated with the R in the running example due to space limitation. The requirement is of the form $i \rightarrow t$. $i \rightarrow t$ can also be written as $\neg i \vee t$. For instance, P1: $i \rightarrow t$ can be written as

P1: *¬ (Stevie indicates his destination as room 3.415) ∨ (Theia shall ask Stevie to walk 10 steps forward)*

which is equivalent to

(Stevie does not indicate his destination as room 3.415) ∨ (Theia shall ask Stevie to walk 10 steps forward)

Stevie does not indicate his destination as room 3.415 is a risk (when the destination he wants to go to

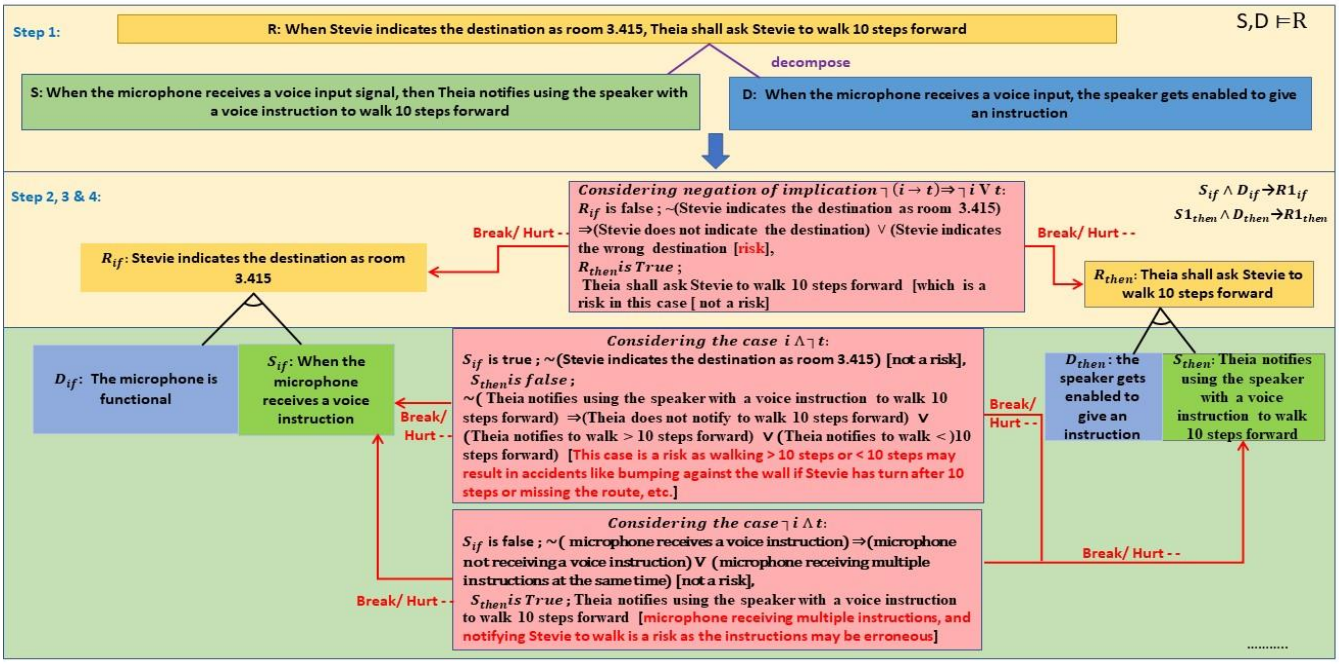

Figure 3. Risk Analysis Graph (RAG) explained with an instance-level example

3.415). This risk might have many cases, such as indicating the wrong room number as his destination, not indicating any room number after turning the app on, an unclear indication of his destination, etc. To identify different possibilities of risks, we negate ***P1*** represented as ***P2***,

P2*: ¬ [(Stevie does not indicate room 3.415 as his destination) ∨ (Theia shall ask Stevie to walk 10 steps forward)]*

Negation yields ***P3,*** which is

P3*: [(Stevie indicates his destination as room 3.415) ∧ ¬ (Theia shall ask Stevie to walk 10 steps forward)]*

which would lead to ***P4***

P4*: [(Stevie indicates his destination as room 3.415) ∧ (Theia shall not ask Stevie to walk 10 steps forward)]*

which indicates a risk. This case of risk where Theia shall not ask Stevie to walk ten steps forward can be analyzed. Multiple cases could be associated with this risk, such as Theia may ask Stevie to walk eight steps or may ask him to walk 12 steps, etc. How can we try to alleviate this risk? Risk mitigation mechanisms can be designed based on the risks obtained. For instance, to make sure that Stevie walks the correct number of steps, a screen-tapping mechanism can be introduced, where Stevie taps the screen for every step taken to keep a count on the steps taken.

Not all risks can be addressed by logic, and there are some shortcomings as well. Considering the truth values for $i \rightarrow t$, if 'i' is false, irrespective of whether 't' is true or false, the statement $i \rightarrow t$ is always true [13]. This analysis will help us find a few risks which a simple negation of $i \rightarrow t$ could not find. For instance, t: *Theia shall ask Stevie to walk 10 steps forward* makes the truth value false, but if *Stevie does not indicate his destination as room 3.415*, this entire statement is true according to logic, but in reality, it is not. Similarly, if 'i' is false and 't' is false (negated), the entire statement would still be true. Secondly, after analyzing the possibility for risks apart from the logical negation of $i \rightarrow t$, which is $(\neg i \vee t)$, it is found that $(\neg i \wedge t)$, which cannot be obtained by the logical negation of implication, pulls in risk(s).

Risk Analysis Tool: We developed a risk analysis tool to generate the RAG for performing risk analysis. The formal strategies addressed in step 5, namely $\neg (i \rightarrow t)$, $(\neg i \wedge t)$, $(i \wedge \neg t)$, etc. are used as templates for semi-automation of risks using the requirements, specifications, and domains obtained in Step 2, Step 3 and Step 4 and use the ontology captured in Step 1 to identify the most important risks obtained in the semi-automation process. The tool's images are not shown here due to space limitation, but the results have been discussed briefly.

IV. EXPERIMENTATION

We have experimented in two parts to validate our risk analysis process through 1) group projects of several undergraduate, graduate-level, PhD.-level requirements engineering courses, which one of the coauthors has been teaching for more than 12 years 2) generating the RAG to obtain risks and analyze them. Most of the students involved in the experimentation for both parts of the experiments were majoring in Computer Science. Students learned about the concepts related to the Reference Model, Ontology, etc., as a part of their coursework and applied this knowledge along with using Murphy's Law to perform both parts of the experiment, respectively.

For experiment 1, the problem domain for all the projects was to develop a smartphone app for helping blind people navigate indoors that was not the same. We have selected around 30 projects, with approximately four students on average in each project. They developed a questionnaire for identifying the risks that may arise while the blind person navigates indoors before developing the actual application.

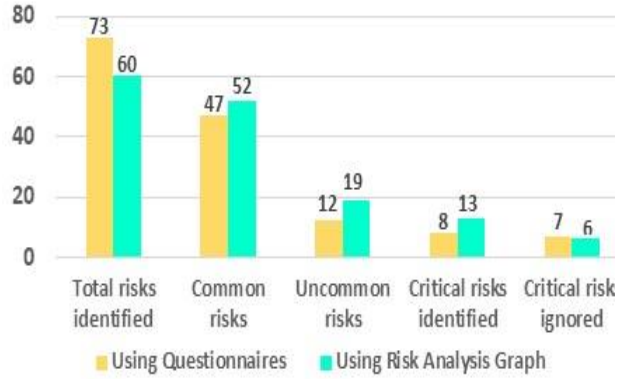

Figure 4. Graph depicting the risks found using questionnaires vs. RAG

This study has shown that students could find out risks but lacked identifying critical and uncommon risks. The Teaching Assistant (TA), one of this paper's coauthors, has carried out a detailed review and analyzed different kinds of risks obtained by the teams. The questionnaires were able to identify some risks. Since the questionnaires developed were at a graduate-level, their analysis was restricted to being very basic and shallow. The risks identified by the teams were at a brainstorming-level when compared to the risks identified by developing the RAG.

For the second part of the experiment, 30 PhD. and 30 senior-level graduate students volunteered to help us experiment. Every student was provided with the initial version of the tool required to generate the RAG. The students were given the set of functional requirements, including the running example. They had the liberty to test their own functional requirements, choose the branches of specification or domain for which risk analysis should be performed, and when to stop the risk analysis. The students followed the process described in Section III to generate the RAG.

The students provided their feedback regarding the ease of use, accuracy of the automation, usability, including a list of risks classified into critical, important,

unimportant, uncommon risks, etc. The questionnaires' results were compared to the results obtained from the Risk Analysis Graph, as shown in Fig. 4.

V. DISCUSSION

A. Overall Observation: We have observed that the students who used RAG were not only able to find common risks such as missing route, walking in the wrong direction, etc., unimportant risks such as warnings which ask them to increase the volume, increase screen brightness, etc., but also were able to identify some critical risks such as falling down, bumping into people, colliding against walls, unexpected object running into the user, etc. and uncommon risks such as oil on the floor, banana peel on the way, etc. while some students who developed and used the questionnaires to find risks have ignored a few critical risks such as low battery indication, faulty voice input due to background noise, walking in a zig-zag fashion in a straight corridor, etc. which we were able to find out by generating multiple RAG's using different sets of requirements.

B. Threats to Validity: Our evaluation is based on human knowledge, and the decision to generate the RAG using the semi-automated tool may not be accurate all the time. The results of the experimentation included are not real software projects (questionnaires). As our evaluation can be subjective and incomplete, it should be expanded with various subjects (developers, requirements engineers, etc.). Furthermore, the range of experiments and the data obtained was also limited. To try a more diverse range of domains, we do not have sufficient guiding ontology for customizing the model.

VI. CONCLUSION

This paper has presented an ontology-based framework for performing risk analysis by using a Risk Analysis Graph (RAG). The Augmented Reference Model obtained by transforming the Reference Model is illustrated by using a collaborative system as a reference application to validate the strengths and weaknesses of the Risk Analysis framework. More specifically, this paper has presented 1) an ontology, which incorporates crucial concepts such as Agents, Risks, Requirements, Specifications, etc.; 2) the Augmented Reference Model, obtained by transforming the Reference Model to perform risk analysis by negating the events in the environment; 3) A Risk Analysis Graph (RAG) to identify and analyze risks by the negation of logical implication and a couple of cases to identify risks which cannot be obtained by negation. The experimentation, we feel, shows that our approach facilitates the detection of several kinds of risks (common, uncommon, critical, etc.). Apart from these, we feel that we could find critical and unexpected risks using RAG.

There are several lines of future work that we would like to work on. We plan on adding risk prevention and risk mitigation strategies to the risks identified using the RAG. We also plan to develop a constructing algorithm for developing the RAG. There are other domains, such as the Auto-drive domain for Autonomous vehicles, etc., that we would like to extend our work to. We would also investigate more ontologies pertaining to other domains as well. The tool is in its first phase of implementation, and work is being done on adding more features for performing risk identification, risk prevention, and mitigation techniques. For engineers to develop and design their own graphically oriented Risk Analysis Graph (RAG) for identifying risks is also underway. We also plan to include safety and timeliness softgoal and extend our work using a goal-oriented approach.

REFERENCES

[1] Asnar, Y., Giorgini, P. Mylopoulos, J., "Goal-driven risk assessment in requirements engineering. Requirements", Eng 16, 101{116 (2011). https://doi.org/10.1007/s00766-010-0112-x

[2] Murphy's Law, https://en.wikipedia.org/wiki/Murphy%27s_law. Last accessed 29 September 2019

[3] Gunter, C. A., Gunter, E. L., Jackson, M. Zave, P., "A reference model for requirements and specifications," in IEEE Software, vol. 17, no. 3, pp. 37-43, May-June 2000, doi:10.1109/52.896248.

[4] Zave, P., Jackson, M.(1997)., "Four dark corners of requirements engineering', ACM Trans. Softw.Eng. Methodol. 6, 1 (Jan. 1997), 1{30.DOI:https://doi.org/10.1145/237432.237434

[5] R. Fredriksen, M. Kristiansen, B. A. Gran, K. Stølen, T. A. Opperud, and T. Dimitrakos, "The coras framework for a model-based risk management process", in Computer Safety, Reliability and Security, ser. Computer Safety, Reliability and Security. Springer Science + Business Media, 2002, pp. 94– 105

[6] Lamsweerde, A. V., "Risk-driven Engineering of Requirements for Dependable Systems", Engineering Dependable Software Systems (2013).

[7] Cailliau, A. Lamsweerde, A. V., "A probabilistic framework for goal-oriented risk analysis",(2012). 20th IEEE International Requirements Engineering Conference (RE). Chicago, IL, 2012, pp. 201-210. doi: 10.1109/RE.2012.6345805.

[8] Cailliau, A., van Lamsweerde, A, "Assessing requirements-related risks through probabilistic goals and obstacles", Requirements Eng 18, 129{146 (2013). https://doi.org/10.1007/s00766-013-0168-5

[9] Sales T.P., Bai~ao F., Guizzardi G., Almeida J.P.A., Guarino N., Mylopoulos J. (2018), " The Common Ontology of Value and Risk. In: Trujillo J. et al. (eds) Conceptual Modeling", ER 2018. Lecture Notes in Computer Science, vol 11157. Springer, Cham. https://doi.org/10.1007/978-3-030-00847-5 11

[10] Rosa, E., "Metatheoretical foundations for post-normal risk." Journal of Risk Research 1 (1998): 15-44.

[11] F. Başak Aydemir, P. Giorgini and J. Mylopoulos, "Multi-objective risk analysis with goal models," *2016 IEEE Tenth International Conference on Research Challenges in Information Science (RCIS)*, Grenoble, France, 2016, pp. 1-10, doi: 10.1109/RCIS.2016.7549302.

[12] Greenspan, S., Mylopoulos, J. Borgida, A., 1994, " On formal requirements modeling languages: RML revisited", In Proceedings of the 16th international conference on Software engineering (ICSE '94). IEEE Computer Society Press, Washington, DC, USA, 135-147.

[13] Implication, https://en.wikipedia.org/wiki/Material_conditional. Last accessed 25 February 2020