ISBN 978-981-11-0008-6
Proceedings of 2016 6th International Workshop on Computer Science and Engineering (WCSE 2016)
Tokyo, Japan, 17-19 June, 2016, pp. 603-606
doi: 10.18178/wcse.2016.06.104

Improving Feedback to Novice Learners on Constructing i* Requirements Diagrams

Hajer Mejri [1 +] and Pauline N. Kawamoto [2]

[1]Interdisciplinary Graduate School of Science and Technology, Shinshu University, Nagano, Japan
[2]Department of Computer Science and Engineering, Shinshu University, Nagano, Japan

Abstract. Although a number of free computer-aided tools are available to help users construct requirements models according to the prescribed design rules of the i* modelling language, none of the tools surveyed thus far provided a full set of syntax checking features. To cover this gap in the current i* tools landscape, we are developing a web-based system called *i*Check* to complement the available tools and offer users automatic on-demand feedback on the syntactical quality of their diagram designs. In this work, we use *i*Check* to explore how different kinds of feedback can help the beginner recognize design mistakes at an early stage in learning. By asking new learners to debug a series of i* diagrams, we investigated the effectiveness of our tool in guiding them to locate and correct defects in given designs compared to some existing tools' checking features. We aimed to determine the factors that contribute to the success or failure of our solution. Based on experimental observations we noticed that i) the usage of *i*Check* seems to lead to a good overall score and ii) this tool bears potential for improvement especially making it more concise in locating where in a diagram the defects appear.

Keywords: design quality of requirements models, *i** framework, recognition and correction of design errors.

1. Introduction

Requirements Engineering is steadily gaining recognition as an integral step in the software development process and the i* (i-star) goal-modelling language and framework [1] is emerging as one of the top candidates for introducing the subject in undergraduate and graduate engineering courses. Despite its simple set of graphical concepts, mastering the techniques of expressing system requirements using the i* language is not straightforward for novice learners and typically requires some practical training time to learn the concepts [9]. There have been a number of efforts to develop computer-aided tools [2][3][4][8] that could be used to help students learn the i* language constructs and design methodologies, particularly in a group training environment where it is unlikely that an instructor or assistant will always be nearby to offer advice on diagram construction. However, unlike the development tools for many programming languages which are equipped with syntax checking and debugging functions, the freely available design tools surveyed had only limited checking features which can create confusion and misunderstanding for students studying alone.

In our previous paper, we proposed an online support tool called *i*Check* [5] for analyzing system requirements written by students in the i* language. The *i*Check* tool accepts text-based istarML data [6], an XML specification of i* diagram information, generated manually or exported from various i* tools as input through a web browser interface (Fig. 1), analyzes the contents and provides automatic feedback to the users. In this paper, we describe how we used our system to investigate the efficiency of different kinds of feedback (e.g., textual messages, GIF animations) in helping new i* users debug and improve the syntactical quality of i* requirements models on their own when a human instructor or assistant is not nearby to offer advice.

[+] Corresponding author. Tel.: + 81 262695591; fax: +81 262695591.
E-mail address: mejri85@gmail.com.

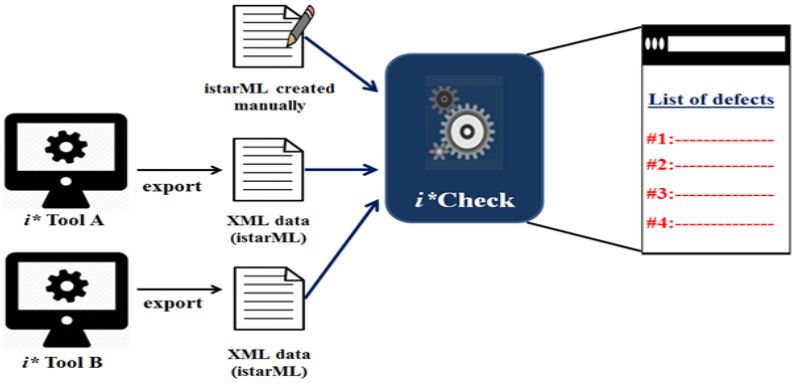

Fig. 1. Overview of *i*Check* system which detects design defects in requirements models written in istarML (XML) format manually or exported from existing tools.

2. Detection and Correction of Errors in i* Strategic Dependency (SD) Diagrams by Novice Learners

In the i* framework, two levels of diagrams are used to describe system requirements: 1) the strategic dependency (SD) model for showing how actors/agents in a system depend on one another and 2) the strategic rationale (SR) model for expressing the goals of each actor/agent and how each goal can be reasoned and achieved. In our earlier work, we found that although there are a number of software tools to help users construct these diagrams, none of the tools surveyed provided a complete syntax checker feature. Having an incomplete list of design defects or ambiguous information on the types of error to be fixed and where they occur can create confusion for beginner learners in the classroom. The resolution of such problems is time-consuming particularly if poor design habits need to be unlearned after introductory instruction has completed. In this investigation, we identify the types of problems that need to be resolved before such tools can be used in training environments or courses where large groups of students need to learn how to correct mistakes on their own.

We focused first on the SD diagram construction rules and quickly observed that although the individual rules are not complicated, when the diagrams to be reviewed or edited contain many components, it is easy for beginners to overlook errors that can cause problems in later development steps.

In developing the *i*Check* system, we initially aimed to construct a thorough syntax check of all specification concepts presented in the guidelines of the i* Wiki guide [7] for SD diagram construction. Preliminary tests of early versions of the tool indicated that although *i*Check* returned a full set of design defects in textual format, for beginners sometimes an incomplete list of errors generated by some available tools was a more effective aid in completing the task because the messages were combined with visual markers of where the errors occurred. Since the *i*Check* system is intended to work in complement with the available tools and not as a replacement for them, we expanded the development and followed up our tool to include GIF format animation (Fig. 2) for explaining how to remove specific errors in a model and this combination proved to be the most effective in reducing the confusion beginners experience in debugging exercises of defected requirements models.

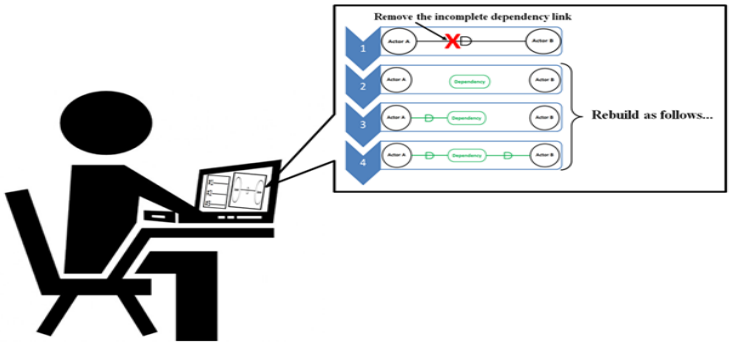

Fig. 2: Graphical hints (GIF animation) from *i*Check* system guide users step-by-step on how to correct design errors related to the core SD diagram construction rules.

In a series of tests, we measured how quickly and accurately engineering students with an introductory background on requirements engineering methodologies using the i* framework are able to identify and correct design errors in a simple SD diagram such as the example in Fig. 3 using a textual description of the system requirements and various types of syntax checking feedback to see how well each type is able to guide them through the tasks.

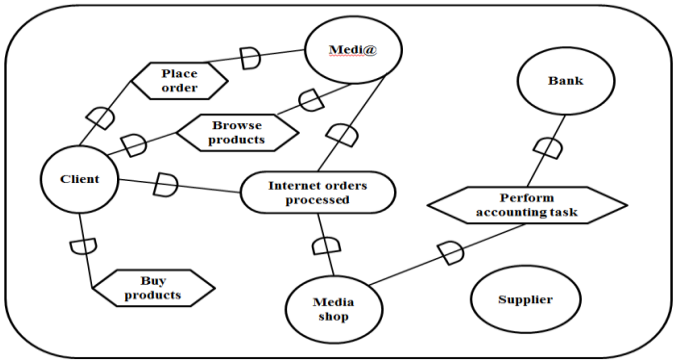

Fig. 3: Sample test item with four defects given to participants to debug using only limited feedback (two error messages) from an existing tool (average success: 50%).

Participants in the study were undergraduate computer science and engineering students at Shinshu University (Nagano, Japan) with no previous experience with the i* modelling language. Following an introductory tutorial on the *i** modelling language and framework, the participants were given a number of sample models to debug using various types of syntax checking information from currently available systems and the *i*Check* system developed in this work. The test groups included: 1) NoTool users, who received no syntax checking feedback for the debugging task, 2) WithAvailTool users, who received feedback from an available tool that offers partial syntax checking features, and 3) WithI*Check users, those using the *i*Check* support tool. In the test, the subjects of the NoTool group could rely only on their understanding and knowledge from the tutorial to do the debugging work. The WithAvailTool subjects were able to view carefully a snapshot of the syntax checking results of a selected tool as a reference in deciding how to correct the defects. The final WithI*Check group was able to use the online *i*Check* system to view the list of defects detected in the model and access the correction tips animations as needed as shown in the image in Fig. 4.

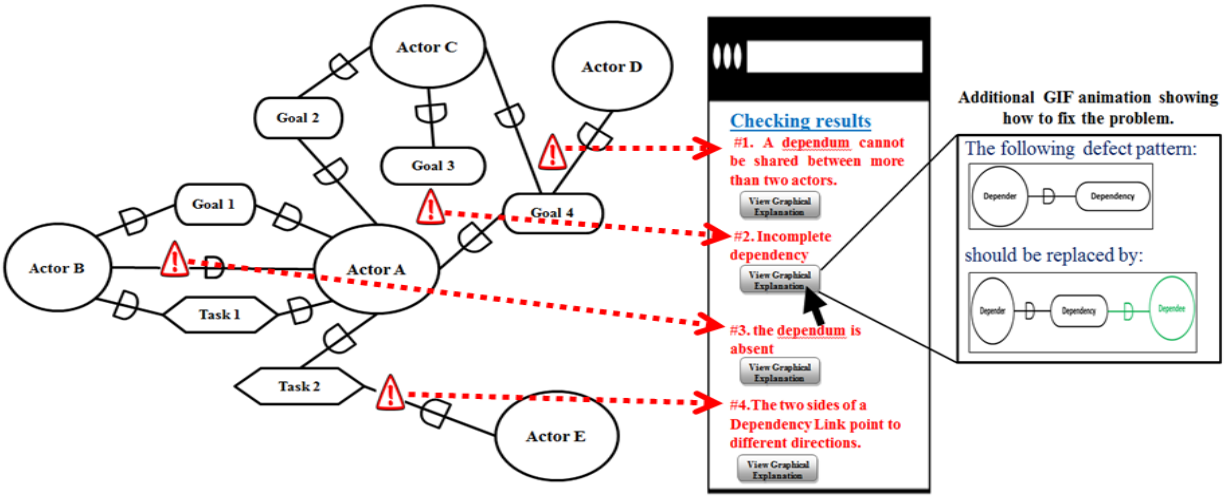

Fig. 4: *i*Check* web tool checks for design rule violations in an SD diagram and provides correction hints with text and GIF animation.

To begin, the subjects did a warmup exercise to be sure they understood the i* design concepts to be judged and each subject had no difficulty in debugging the simple example. However, each model used in

the actual test (e.g., Fig. 3) contained 4-5 errors to be corrected and the average rates of success for each group are summarized in Table I along with the average time required to complete the debugging task.

Table I. Results of i* SD diagram debugging tasks using various feedback sources

Error Feedback Type	Number of participants	Average score (%) of debugging task	Average time for task completion
NoTool	4	45%	13.5 min.
WithAvailTool	4	50%	7 min.
WithI*Check	4	75%	13.5 min.

A post-test questionnaire was used to verify that subjects were fairly confident in their answers, but as seen in the results of the table, without a complete list of defects the novice learner is not able to sufficiently find and correct more than half of the design errors even in a small-scale test model.

The average time for task completion with the *i*Check* tool was the same as the case of having no error feedback references and we observed that the subjects spent most of this time viewing the correction hints animations to achieve a high success rate in the specified task.

3. Conclusion and Future work

In this work we developed an online tool for analyzing system requirements model data written in the i* language to complement the error checking features of freely available i* design tools to provide more thorough feedback on the syntactical quality of models. Our tool also provides specific instructions in GIF animation format on how to correct the defects found in a design which proved to be a useful feature for new learners. This combination of tools will become more important as the need for introductory courses and training in requirements engineering methodologies increases in engineering curriculums.

Our future work will be to develop the analyzer function of *i*Check* to evaluate the correctness of the SR diagram elements which will overlay the SD diagram making the requirements model more complex and thus more difficult for beginners to debug. The style of providing graphical step-by-step hints on how to resolve syntax problems which served to be effective in guiding beginners in their debugging work will be continued in the next version of the tool.

References

[1] E. Yu. "Modeling Strategic Relationships for Process Reengineering". PhD thesis, Department of Computer Science, University of Toronto, Canada, 1995.

[2] HiME: Hierarchical i-star Modelling Editor, http://www.upc.edu/gessi/istar/tools/hime/index.html .

[3] Á. Malta, M. Soares, E. Santos , J. Paes , F. Alencar and J. Castro. iStarTool: Modeling requirements using the i* Framework. *CEUR Proceedings of the 5th International i* Workshop* (iStar 2011), Trento, Italy, pp. 163-165.

[4] OpenOME, an open-source requirements engineering tool, http://www.cs.toronto.edu/km/openome/ .

[5] H. Mejri and P.N. Kawamoto. i*Check: A Web-based Tool Assistant for Detecting Design Errors in i* Model Data. *Proc. of ISERD International conference*, Kyoto, Japan, 2015, pp.93-96.

[6] iStarML: The i* Mark-up Language reference guide, http://www.essi.upc.edu/~ccares/papers/istarmlRefGuide.pdf.

[7] i* Wiki Guide, http://istar.rwth-aachen.de/tiki-index.php?page=i%2A+Guide .

[8] R. Laue, A. Storch. A Flexible Approach for Validating i* Models. *CEUR Proceedings of the 5th International i* Workshop* (iStar 2011), Trento, Italy, pp. 32-36.

[9] J. Horkoff. Observational Studies of new i* Users: Challenges and Recommendations. *Proceedings of the 1st International iStar Teaching Workshop* (iStarT 2015), Stockholm, Sweden, 2015, pp. 13-18.