# An Automated Collaborative Requirements Engineering Tool for Better Validation of Requirements

Nor Aiza Moketar, Massila Kamalrudin, Safiah Sidek
Innovative Software System and Services Group,
Universiti Teknikal Malaysia Melaka, Melaka, Malaysia
nor.aiza09@gmail.com, {massila, safiahsidek}@utem.edu.my

Mark Robinson
Fulgent Corporation, USA
marcos@fulgentcorp.com

John Grundy
Faculty of Science Engineering and Built Environment, School of Information Technology, Melbourne Burwood Campus, Deakin University, Victoria 3125, Australia
j.grundy@deakin.edu.au

## ABSTRACT
This demo introduces an automated collaborative requirements engineering tool, called TestMEReq, which is used to promote effective communication and collaboration between client-stakeholders and requirements engineers for better requirements validation. Our tool is augmented with real time communication and collaboration support to allow multiple stakeholders to collaboratively validate the same set of requirements. We have conducted a user study focusing on validating requirements using TestMEReq with a few groups of requirements engineers and client stakeholders. The study shows that our automated tool support is able to assist requirements engineers to effectively communicate with client-stakeholders to better validate the requirements virtually in real time. (Demo video: https://www.youtube.com/watch?v=7sWLOx-N4Jo).

## CCS Concepts
**• Software and its engineering~Requirements analysis**
**• Software and its engineering~Acceptance testing • Software and its engineering~Collaboration in software development**

## Keywords
Abstract test, Essential Use Cases, Essential User Interface, requirement-based testing, requirements validation, communication and collaboration

## 1. INTRODUCTION
Communication and collaboration between requirements engineer and client-stakeholders is one of the most important activities in requirements engineering process [1]. As the initial stage of any software development, it is the key component to achieve success in a software development project [2]. Any errors found at the requirements stage will disrupt the completion of a project and cause many other problems. At this initial stage, a lot of key information and requirements about a project are reported and documented from the client-stakeholders. Therefore, it is very important to validate, verify and clarify the information so that common understanding of and agreements on the requirements can be achieved at the early stage of the development stage.

In our previous work, we have presented an automated requirements validation tool, called TestMEReq[1][3]. This tool is able to automatically generate a combination of abstract test cases and mock-up user interface (UI) prototypes from semi-formalised Essential Use Cases (EUC) and Essential User Interface (EUI) models [4], [5]. Our automated approach assists requirements engineers to validate requirements with the stakeholders; hence, it helps to reduce the cost of generating and designing test cases and user interface prototypes.

We have extended our tool to support more effective communication and collaboration for better requirements validation process. Our tool has a new feature that allows multiple users to collaborate synchronously and asynchronously regardless of their location whether in remote or co-locate project. To our knowledge, this approach is unique in the sense that it supports effective communication and collaboration for early testing during the requirements validation phase. Further, we also enhance the tool with template-based tests authoring to assist requirement engineers in writing quality test requirements and test cases.

## 2. OUR APPROACH
We have developed an approach and automated support tool called TestMEReq that enables requirements engineers to effectively communicate and collaborate with client-stakeholders to discuss and validate the quality of the captured requirements. Figure 1 shows an overview of our approach for collaborative requirements validation. It is an extension to our previous work [3], [6], [7]. Our new work is labelled as (B) and (C). The process starts with the requirements engineer and the client-stakeholders use the TestMEReq to validate the requirements (A). Here, the users need to key in their requirements in the form of user story or use case scenario (1) to generate the related EUC and EUI models (2) from the textual requirements. The requirements are analysed with the EUC and EUI pattern libraries to generate the related EUC and EUI models. Then, a set of abstract tests consisting of test requirements and test cases are generated (3)(4). This automated process is supported with the test requirements and test case pattern libraries [3]. The users may then execute the test cases on the generated mock-up user interface to review and validate the expected behaviour of the requirements. Then, they need to indicate their testing status. The testing status will be saved in the database for future reference.



*ASE'16*, September 3–7, 2016, Singapore, Singapore
© 2016 ACM. 978-1-4503-3845-5/16/09...
http://dx.doi.org/10.1145/2970276.2970295

[1]Tool demo is available at https://youtu.be/oCZYaU9Mbxg

The requirements engineer can review the testing results from the client-stakeholder. If there are any conflicting results from all the client-stakeholders, the requirements engineer can initiate for collaboration with the client-stakeholders for further discussion (B). Here, the requirements engineer needs to share the access to the client-stakeholders by sending an invitation email to all relevant stakeholders. Upon receiving the email, the client-stakeholders can click on the provided link to join the discussion. They can comment any part of the results that they disagree during the discussion. They also can simultaneously edit the document and communicate to each other using the chatting facilities.

Upon agreement from all stakeholders, the requirements engineer may add and update the pattern library of TestMEReq through the tests authoring-template (C). This template helps the requirements engineer to write quality test requirements and test cases. We have embedded a natural language parser for English language to ensure the correctness and accuracy of the test requirements and the test cases provided by the requirements engineer. The parser helps to ensure that the user follows the correct sentence structure from our test requirements pattern library. We have defined the syntax rules for the sentence structure as the following:

<Action verbs> [Actor]<Auxiliary verbs> [Action] [Condition]

The parser helps to ensure the requirements engineer uses the right terms for the test requirements in order to ensure correct sentences is written and reflected to the objectives/goals of the requirements. For this, the user must use an infinitive/action verbs and words such as “Validate that...”, “Verify that...” and “Test that...” in the test requirements statements. Some examples of the sentences that follow our test requirements sentence structure are:

1. Validate (VB) that (Art) user (NN) can (MD) login (VB) with (Prep) valid (Adj) user name and password (NN).
2. Validate (VB) that (Art) user (NN) can (MD) withdraw (VB) the correct (Adj) amount (NN).

Our tool also provides prompt notification and feedback as well as highlighting of errors to alert requirements engineer to any defects found in the test requirements and test cases. The tool is flexible as it allows users to also ignore the notification if they disagree with it. This may be helpful if the user wants an addition to be made to the test requirements and test cases, which later can be reviewed by the requirements engineer. Further, the test-authoring template also helps to enhance the scalability of our test requirements pattern library by allowing requirements engineers to insert new test requirements and test cases for other domains of applications.

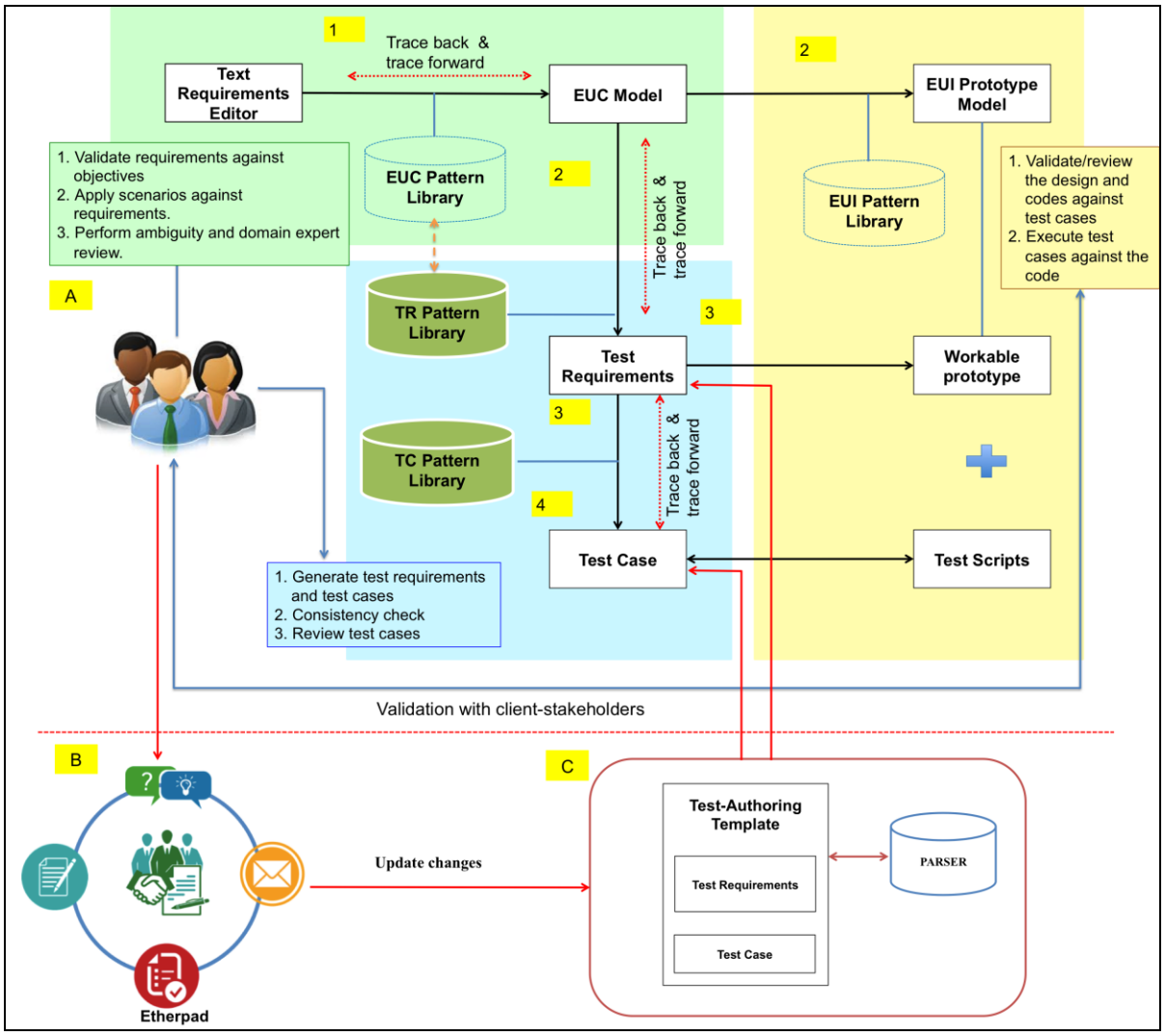


**Figure 1. The overview of our proposed approach.**

## 3. TOOL SUPPORT: TESTMEREQ

We have extended our TestMEReq tool with communication and collaborative support to allow multiple stakeholders to validate the same set of requirements at anytime and anywhere. For this, we have integrated the TestMEReq with an open source tool, Etherpad [8] an online editor for collaborative editing in real-time. Further, we enhanced the TestMEReq with a template-based test authoring to assist the requirements engineer to write quality test requirements and test cases, which are in compliance with our pattern libraries. Several screen dumps of the tool in use are shown in Figure 2. From a set of natural language requirements, a semi-formal EUC models are extracted (1) and then mapped to a low-fidelity EUI model (2). Then, a set of test requirements (3) and test cases (4) is generated. Users can validate the requirements by executing the test cases on the mock-up user interface prototype (5) and see the expected results. Then, users can indicate and save their testing status for future reference. From here, the requirements engineer can initiate collaboration with the client-stakeholders by clicking the "Review" button. Then, he/she will be navigated to the Etherpad screen that is integrated with the TestMEReq, as shown in Figure 3. Here, he/she can share the access to the screen with the client-stakeholders by sending an invitation email (A). The requirements engineer can indicate the access level for the invited users, whether he/she can edit, view or comment on the document. The requirements engineer and the invited client-stakeholders can also simultaneously edit the documents (B), add comments (C) and communicate with each other via the chatting facility (D). They can also view the online users in the collaborative session (D). Once confirmed and agreed by the client-stakeholders, the requirements engineer can add/update the EUC model, test requirements and test cases through the test-authoring template of TestMEReq, as shown in Figure 4. At the pattern editor page, the requirements engineer needs to search for the relevant EUC model, test requirements or test cases that need to be added, updated or deleted (A). Figure 6 (B) shows the test requirements template form that allows the requirements engineer to add or update the test requirements. An error message will appear to warn the requirements engineer if the test requirements do not follow our test requirements sentence structure.

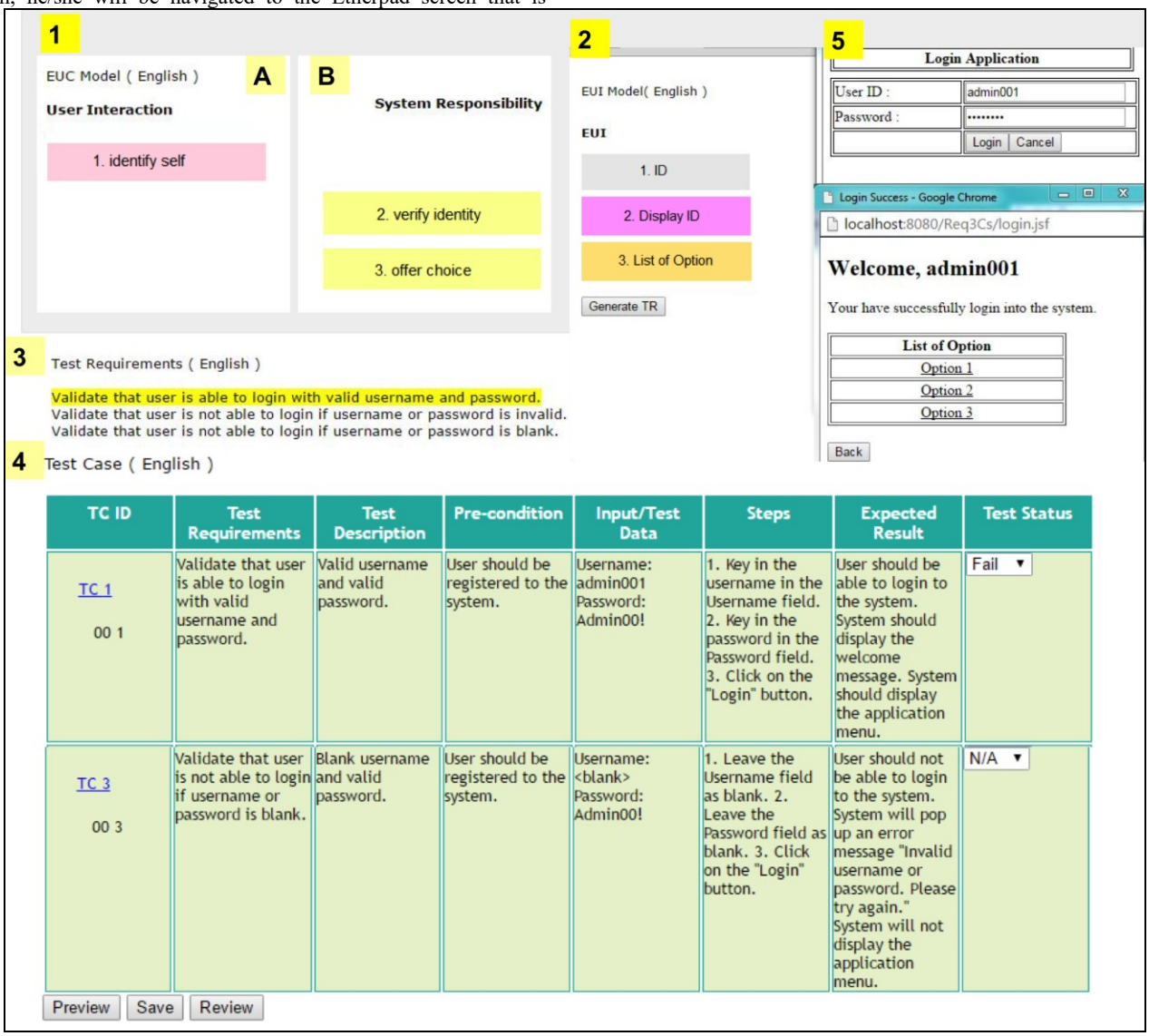


**Figure 2: TestMEReq generates EUC model, EUI prototype model, test requirements, test cases and mock-up UI prototype from natural language requirements.**

# 4. EVALUATION

This study focuses on answering a concrete research question: *Can an automated approach and tool support help to improve the communication and collaboration between requirements engineer and their clients in order to validate the requirements*? For this aim, we have conducted a small user study to observe the effectiveness of our approach in a collaborative requirements validation environment.

## 4.1 User Study

We have conducted a qualitative study with six post-graduate students who worked in pairs as the requirements engineer and the client. Some of the participants have working experience in the software industry. The participants were given an explanation and demonstration of our TestMEReq tool and task to be performed. For the experiment, every participant received a laptop or an android tablet to access the TestMEReq tool. Each of the pairs was requested to explore our tool with some requirements sample. They were given one hour to identify any missing requirements: incomplete and inconsistency between the generated EUC and the EUI model as well as the test requirements and the test cases. In this study, they were required to identify three inconsistent EUC and EUI model, two incomplete or missing test requirements and four incomplete test cases. They were also requested to discuss the requirements using the facilities in our tool, such as the comment and the chatting tool. During the workshop, they were assigned in two different rooms and not allowed to communicate verbally to each other. We observed their behaviour and transcribed their comments and chatting history to analyse their communication behaviour. Finally, they were requested to answer a questionnaire to identify their level of satisfaction when using our tool that allows them to communicate and collaborate in the requirements validation process. We also conducted a semi-structured interview to seek their opinions whether the approach helps to improve the communication and collaboration between the requirements engineer and client and whether it helps to improve the requirements validation process.

From our observation, we found that TestMEReq assisted both requirements engineer and clients to better communicate and collaborate, to discuss and to validate the intended system requirements. In evaluation 1, RE1 stated that the tool encouraged him to ask the client to confirm and validate the consistency and completeness of the requirements that he had captured in TestMEReq. An extract from the dialogue is as follows:

RE1: "Here is the requirement scenario of your requirements. From here we got the EUC and EUI models as well as the test requirements and test cases. What do you think?"

C1: "I think the EUC and EUI models are not in the correct order. The item "List of option" must be before the "Choose item"."

RE1: "Ok. Let's re-arrange the use case scenario and let see the EUC and EUI models."

*RE1 re-wrote the use case scenario as per the client's instructions and then showed to C1 the generated EUC and EUI models from the scenario. C1 was then requested to validate and confirm the modified requirements against the original requirements and responded:*

C1: "Yes. I think it is fine now."

A similar dialogue occurred in evaluation 2:

RE2: "This is the outcome of your requirements. Can you please have a look on each requirements components: EUC, EUI, test requirements and test cases to confirm that we have the right requirements."

C2: "I think the requirements scenario is not tallied with our initial requirements. It is not in the right flow of the system that we imagine. First, the user should be able to view the list of product to be added in the system, then he can choose the item from the list."

RE2: "Ok. Let's re-write the flow of the requirements."

*RE2 showed C2 the original requirements in the textual editor and refined it based on C2's instruction and then asked C2 to validate the modified requirements.*

RE2: "Here are the new requirements components. Let's see each of them. Do you agree with this one?"

C2: "Yes, this is what we want."

Similarly, in evaluation 3, C3 highlighted the incomplete test requirements and test cases generated from the textual requirements.

RE3: "Let's test your requirements. These are the four test cases generated from the requirements scenario. What do you think?"

C3: "I think the test cases are not complete. Something is missing. How about if the user key in an incorrect product id?"

RE3: "Ok, I will add a new test case for that."

*The RE3 added a new test case as requested through the test-authoring template. He entered the same requirements scenario again and showed the result to C3.*

RE3: "I have updated the test cases. Please confirm if this is right."

C3: "This is perfect."

In all the three cases, the tool helped the requirements engineer to validate the user's requirements in terms of consistency, correctness and completeness in a timely manner. The requirements engineer can get faster confirmation and agreements without face-to-face meeting with the client. This can help to save time and cost for both the requirements engineer and client. Overall, the evaluation and interview with the participants provide positive results. The requirements engineer stated that the tool helps them to communicate and discusses the captured requirements with the clients, thus it helps to speed up the requirements validation process. The requirements engineers were also satisfied with their interaction with the clients where they find them very helpful in identifying any missing requirements. This helps to avoid incorrect implementation of the requested system.

# 5. RELATED WORK

Many studies have proposed a mechanism for collaborative work in the requirements engineering process, such as the StakeRare [9] and FlexiSketch [10][11]. StakeRare uses social network and collaborative filtering to identify and prioritize requirements in large software projects. An evaluation on StakeRare shows that it is able to accurately identify a complete set of requirements and prioritize them. Meanwhile, FlexiSketch allows multiple users to sketch requirements models or notation simultaneously within the same canvas region. It uses electronic whiteboard to support a synchronous, co-located and multi-display collaboration. Both of these works contribute to support requirements engineer and client-stakeholders during requirements elicitation, except for validation of requirements for final confirmation and agreement.

M. Sourour and N. Zarour [12] have proposed CoREVDO, a methodology for a collaborative requirements validation process based on the concept of competences and multi-viewpoints to increase the quality of software product and reduce the complexity of the validation task. This approach involved the generation of Quality Function Deployment (QFD), checklist and groupware, such as the NetMeeting tool for collaborative work. It focuses more on the internal audit with customer using formal method, such as mathematical logic or algebra. There is no automation for the prototype, where the group of REs needs to make a rapid prototype to simulate the intended system. In contrast to our work, we focus on the external validation with the user using the techniques of requirements-based testing and rapid prototyping to support the automatic generation of the abstract tests and the mock-up UI prototypes. Our approach is also supported with Etherpad to allow the RE and client-stakeholders to effectively communicate and work on the same set of abstract tests that represent their requirements. A summary of our findings of the related tools is described in Table 1.

**Table 1. Related tools comparison**

| Tool | StakeRare | FlexiSketch | CoREVDO |
|---|---|---|---|
| RE Phase | Elicitation | Elicitation | Validation |
| Objective | Identify and prioritise stakeholders to support requirements elicitation in large software projects. | Support synchronous, co-located, and multi-display collaboration for sketching requirements model / notation. | Increase the quality of the software product. Reduce the complexity of the validation task. |
| Collaborative Method | Social network and collaborative filtering | Electronic whiteboard | Prototype QFD, checklist, groupware |

# 6. CONCLUSION AND FUTURE WORK

TestMEReq is an automated collaborative tool that could assist requirements engineers to effectively communicate and collaborate with the client-stakeholders virtually in real time to validate the requirements. Our tool is integrated with Etherpad, an open source tool that provides online editor for collaborative document editing in real-time. Our initial studies suggest that the automated support provided by our tool can help the requirements engineer and the client-stakeholders to effectively communicate and collaborate in real time to validate their requirements even though they are not in the same geographical location. The test-authoring template also helps the requirements engineer to write quality test requirements and test cases compliance with the test pattern libraries. This helps the requirements engineer to write correct and complete test requirements and test cases before reviewing them with the clients.

For future work, we plan to conduct further evaluation with the industry to validate our approach and tool. We also plan to embed a requirement prioritization method to our tool for prioritizing the generated tests for better organization of requirements validation based on the generated test cases. Furthermore, we intend to enhance this collaborative validation tool with better graphical annotation for making comments and function to convert the generated test to spread sheet for future use as a test plan to the testers.

# 7. ACKNOWLEDGMENTS

This research is funded by Ministry of Higher Education Malaysia (MOHE), Universiti Teknologi Mara (UiTM), Fulgent Corporation, FRGS grant: FRGS/2/2013/ICT01/FTMK/02/2/F00185 and RTC/E00038.

# 8. REFERENCES

[1] P. A. Laplante, *Requirements Engineering for Software and Systems*. Auerbach Publications, 2009.

[2] R. R. Young, *Effective Requirements Practice*. Addison-Wesley Information Technology Series, 2001.

[3] N. A. Moketar, M. Kamalrudin, S. Sidek, M. Robinson, and J. Grundy, "TestMEReq : Generating Abstract Tests for Requirements Validation," in *3rd International Workshop on Software Engineering Research and Industrial Practice*, 2016, pp. 39–45.

[4] R. Biddle, J. Noble, and E. Tempero, "From Essential Use Cases to Objects," *forUSE 2002, Second Int. Conf. Usage-Centered Des. forUSE 2002*, vol. 1, no. 978, pp. 1–23, 2002.

[5] L. L. Constantine and L. A. D. Lockwood, "Structure and Style in Use Cases for User Interface Design," vol. 1, no. 978. Addison-Wesley Longman Publishing Co., Boston, MA, 2001.

[6] M. Kamalrudin, N. A. Moketar, J. Grundy, and J. Hosking, "Automatic Acceptance Test Case Generation From Essential Use Cases," in *13th International Conference on Intelligent Software Methodologies, Tools and Techniques*, 2014, pp. 246–255.

[7] N. A. Moketar, M. Kamalrudin, S. Sidek, and M. Robinson, "TestMEReq : Automated Acceptance Testing Tool For Requirements Validation," in *International Symposium on Research in Innovation and Sustainability*, 2014, vol. 2014, no. October, pp. 15–16.

[8] "Etherpad." [Online]. Available: http://etherpad.org/. [Accessed: 16-May-2016].

[9] S. L. Lim and A. Finkelstein, "StakeRare: Using Social Networks and Collaborative Filtering for Large-Scale Requirements Elicitation," *IEEE Trans. Softw. Eng.*, vol. 38, no. 3, pp. 707–735, 2012.

[10] D. Wuest, N. Seyff, and M. Glinz, "Sketching and Notation Creation with FlexiSketch Team : Evaluating a New Means for Collaborative Requirements Elicitation," in *International Requirements Engineering Conference*, 2015, pp. 186–195.

[11] D. Wuest, N. Seyff, and M. Glinz, "FLEXISKETCH TEAM : Collaborative Sketching and Notation Creation on the Fly," in *International Conference on Software Engineering*, 2015, pp. 685–688.

[12] M. D. Sourour and N. Zarour, "A Methodology of Collaborative Requirements Validation in a Cooperative Environment," *Program. Syst. (ISPS), 2011 10th Int. Symp.*, pp. 140–147, 2011.

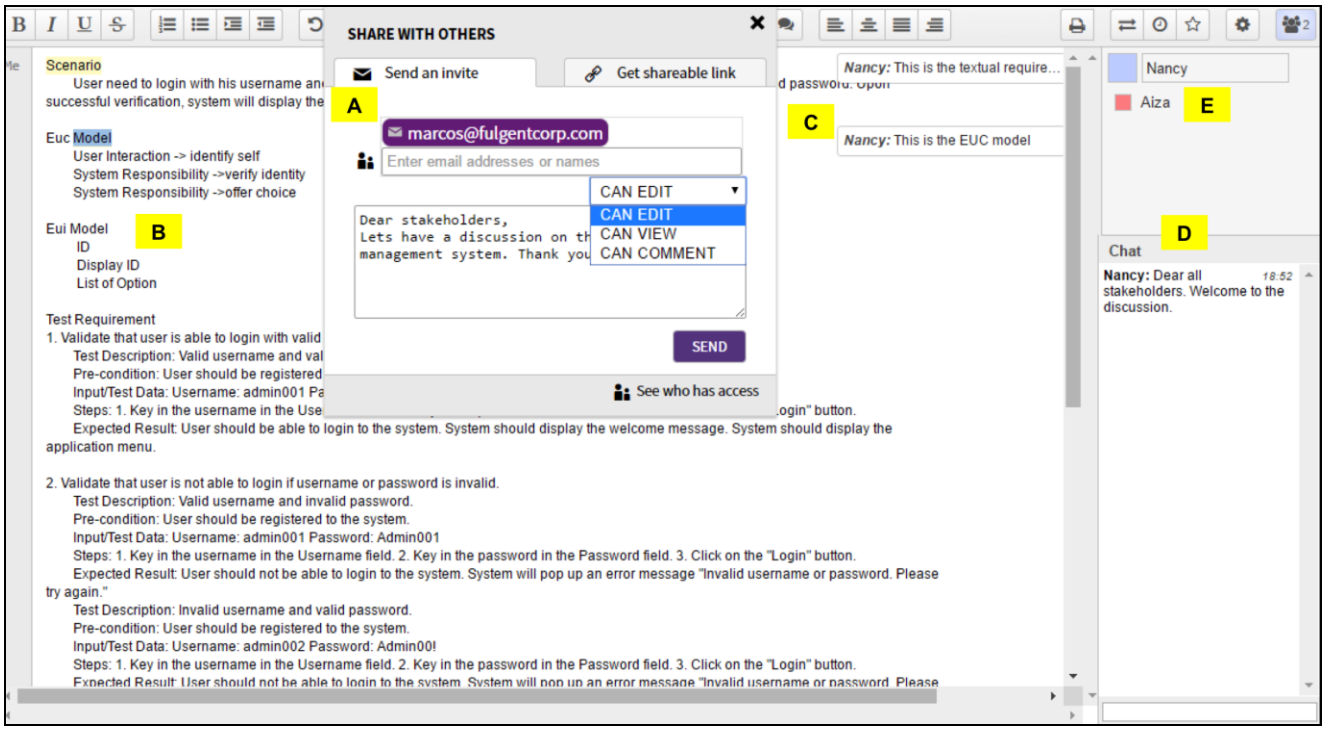


**Figure 3. Etherpad the online collaborative text editor integrated with TestMEReq**

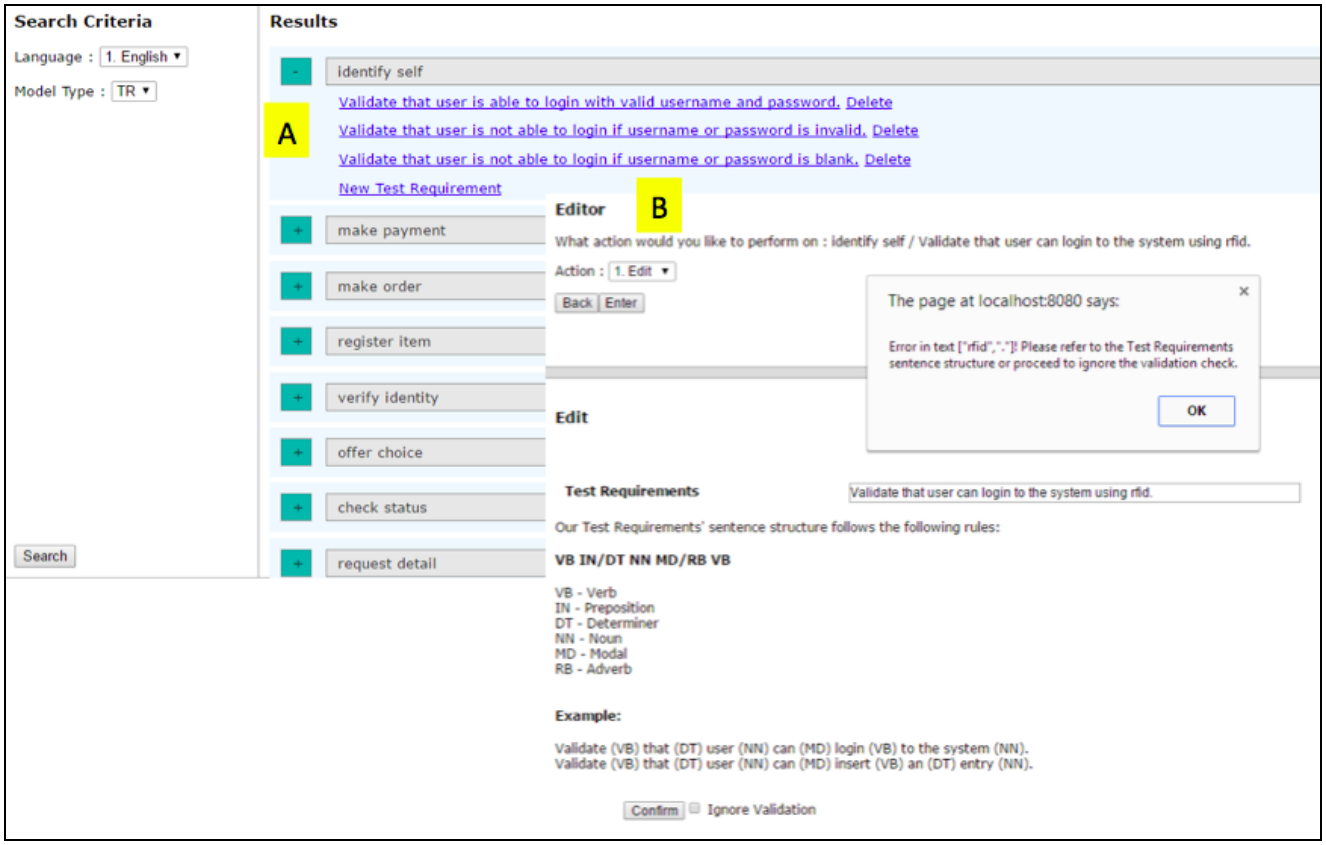


**Figure 4. Template-based authoring of TestMEReq**