2017 IEEE 25th International Requirements Engineering Conference Workshops

Automated Support to Capture Verbal Just-in-Time Requirements in Agile Development: A Practitioner View

Corey Hollis
Information Technology Laboratory
Engineering Research and Development Center
Vicksburg, Mississippi, 39180
Email: corey.e.hollis@usace.army.mil

Tanmay Bhowmik
Computer Science and Engineering Department
Mississippi State University
Starkville, Mississippi 39759
Email: tbhowmik@cse.msstate.edu

***Abstract*—Being able to quickly and accurately capture requirements is crucial when using agile methodologies. Requirements, to that end, are often captured in an as-needed and informal manner, with continuous stakeholder interaction. Techniques such as interviews, user stories, rapid feedback, and text mining are commonly used in the industry to gather such informal requirements, which are often congruent with the concept of 'just-in-time' (JIT)—informally represented and continuously refined—requirements. Oral communication is an integral part of most of these techniques, and the loss or misinterpretation of verbally communicated requirements is a daunting problem. To address this issue, we propose a novel framework to assist developers in capturing verbal requirements in an accurate manner. In particular, our framework automatically captures, transcribes, and mines verbal communication, which in turn produces a set of loosely formulated candidate requirements for further elaboration. We also conducted a survey among practitioners concerning requirements in agile methodologies and our framework. The results of this survey provide positive feedback on the prospects of the framework and also indicate the prevalence of JIT requirements in closed source agile projects.**

***Index Terms*—just-in-time requirements; agile development; audio to text transcription; text mining**

I. Introduction

Agile development has become one of the most prevalent software development methodologies, thus prompting researchers to study how requirement elicitation could be improved in agile software development. Considering the main objectives of agile, such as delivering requirements quickly, improving developer and customer communication, and reacting efficiently to changes, proficient capture of requirements is vital to ensuring continual delivery of valid, working software [1]. In fact, as Choudhary and Rakesh [1] indicate, 'time' is one of the most important factors when using the agile development methodology.

As agile calls for rapid software development, it requires rapid requirements elicitation as well. Consequently, developers capture requirements in an as-needed and often informal manner with continuous customer interaction and a high degree of transparency among all stakeholders [2]. Such activities are congruent with the inception and refinement of just-in- time (JIT) requirements characterized as requirements that are informally captured in a lightweight manner and further explained during implementation [2], [3]. Techniques, such as interviews [4], [5], user stories [5], rapid feedback [6], and text mining [7] are currently used to capture such JIT type requirements in agile development. A common issue with some of these techniques, e.g., user stories and interviews, is the loss or misrepresentation of verbally communicated requirements [8]. In addition, much of the research on JIT requirements has focused on open source software (OSS) [3], [9], whereas JIT requirements in closed source agile development are still unexplored.

In this position paper, we propose a solution to remedy the problem of capturing verbal requirements communication by presenting a framework for recording, transcribing, and mining verbal communication to assist developers in better describing and further refining requirements. We also report feedback from 13 practitioners regarding their experience with agile development and JIT requirements as well as their opinion on the effectiveness of our proposed framework

II. Background and Related Work

A. Common Elicitation Techniques in Agile

Interviews. This technique has been used in many other development paradigms besides agile, but fulfills much of the Agile Manifesto inherently. Interviews involve heavy customer interaction to ensure that valid requirements are being defined [4], [5].

User Stories. When users or customers are unsure of how to communicate their needs, user stories are often created. These short phrases or action items are used to succinctly state what a user would need for a specific task to be performed [5]. This facilitates improved communication and understanding among all stakeholders and is believed to eliminate the need to constantly update specification documents [4].

Rapid Feedback. For agile development to be most effective, customer involvement is crucial. Since there are constant incremental releases during an agile project, one useful technique for capturing new JIT requirements is to ask for quick feedback from customers on new releases. This is known as rapid feedback, and it can be captured as formally as using sophisticated issue tracking systems or as informally as a phone call [6].

Text Mining. An inherently automated technique for capturing JIT requirements is mining text from existing artifacts, such as readme documents, posted requirements, comments, and issues over GitHub, Bugzilla, and other issue tracking systems [10], [11]. Several techniques could be used to achieve such automation. For example, Raisinghani *et al.* [11] conducted text mining by searching for specific phrases within readme data sets, whereas Bhowmik and colleagues [10] applied topic modeling techniques on existing requirements and comments recorded over issue tracking systems. Lucassen *et al.* [12] proposed an automated approach based on natural language processing to extract conceptual models from user stories. Such automated approaches improve the elicitation process and reduce stakeholders' workload. Thus, text mining could be considered among the most efficient techniques for JIT requirement elicitation in agile.

B. A Common Thread and the Role of Active Listening

A common thread among the aforementioned elicitation techniques (except Text Mining) is the use of verbal communication. In fact, Knauss *et al.* [5] reported that face-to-face communication was the most frequently used technique to elicit requirements in agile development where the requirements often followed JIT characteristics (cf. paragraph 2, Section I). To deal with oral communication of requirements, most institutions request for each developer to take notes or to assign a transcriber during meetings. Although note taking is encouraged, it creates a manual burden and often represents inaccurate information [13].

Hristatche *et al.* [14], provides a study that supports this claim, and describes and encourages active listening. The study shows that those who are not actively listening forget approximately 50% of what was said, and the information that is retained is only 60% accurate. In order to retain the maximum amount of information possible when communicating, each party must be fully aware when listening (i.e., active listening). This means using active listening techniques, such as, paraphrasing what was said and asking questions to ensure comprehension. However, when verbally communicating, it is extremely difficult for team members to actively listen if they must constantly write down what is being discussed to ensure they do not miss important information [14]. Therefore, the proposed framework outlined in this position paper provides automated support to help formulate more accurate and thorough JIT requirements from verbal communication, which provides agile teams with the most accurate information.

III. Our Proposed Framework

To eliminate the necessity of constant note taking when discussing requirements verbally, we propose a framework that provides a real time application for recording the conversation and converting it into a text transcript. Next, a text mining program, specifically designed to search for key words that are used to describe requirements and customized additional words, is used to extract important items from the transcript. The text mining program creates an output of a rough set of loosely formulated requirements that could be further refined before being added to the project's requirements list. We have named this technique - Automated Audio Mining.

A. Audio to Text Conversion

In order to capture verbal discussion and translate it into a text transcript, a real time audio-to-text conversion software, such as Dragon NaturallySpeaking Premium, is used. Dragon is a software package that anyone can purchase for a one-time fee of $176.27 on the Amazon website [15]. A useful feature of this tool is the free mobile application, Dragon Remote Microphone, which is included with the purchase of this software. It allows the user to leave the desktop or laptop in the office and use the Dragon Remote Microphone mobile application to transmit the audio to the software instead [16].

B. Customized Text Mining

The text mining program is customized for requirements elicitation with the option for developers to add words or phrases that their organization frequently uses or words or phrases associated with the particular application being developed. To capture the action items that are within the text transcript, we choose to use Fillmore's case theory, which recommends using a *verb* + *noun* construct [17]. The verbs and nouns are extracted using the Python TextBlob tags functionality that associates a part-of-speech tag with each word [18]. We customize this functionality to ensure that the developer added key words were labeled appropriately. Then, we automatedly generate items from this tagged list by searching for all words in a sentence labeled (VB_[1]), or noun (NN_[1]) [19]. These words are concatenated to create a set of loosely formulated candidate requirements. To help shorten the generated list and make each item more relevant, we also check through all previously generated requirements when a new item is created to ensure that no duplicates exist. This step provides a solid basis for each requirement so that the developer can further expand upon the results, and copy it to any requirements management software the company uses.

C. Proof of Concept

Figure 1 depicts the aforementioned workflow. With this new technique, the user first opens the Dragon Remote Microphone application on a smart phone. Then, during the meeting or discussion, the application constantly sends the captured audio to the desktop application, which converts it into a text transcript. Once the translation is complete, the transcript is saved, and the text mining tool retrieves the file. Next, the text mining program outputs the extracted requirements in terms of verbs and nouns.

As a practical example of this workflow, consider a hypothetical bakery named Sandra's Sweets. The team that is

[1]The _ indicates other characters that may proceed the initial letters.

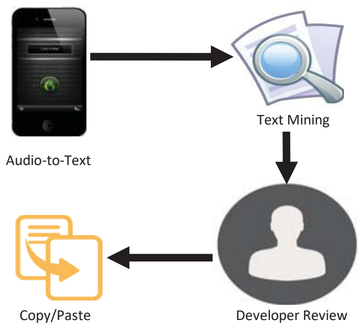
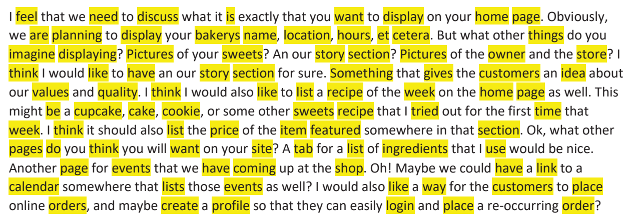


Fig. 1. Automated Audio Mining workflow.

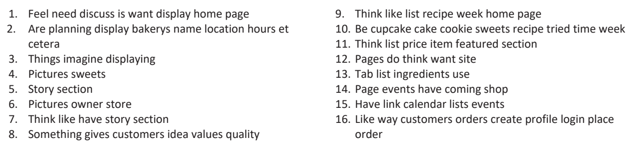
developing software for Sandra's Sweets conducts an interview with several stakeholders to discuss new requirements. During the meeting, several team members ask questions specifically related to their parts of the project in order to better understand what each stakeholder desires from the software. Diane, one of the developers, has Dragon NaturallySpeaking installed on her desktop in her office, and she has customized the text mining tool to parse for words specifically related to this project, in addition to the words and phrases commonly used for requirements. Diane is also using the Dragon Remote Microphone application on her smart phone to capture the entire discussion. A small snippet of the transcript saved on Diane's machine is presented in Figure 2. While the team



I feel that we need to discuss what it is exactly that you want to display on your home page. Obviously, we are planning to display your bakerys name, location, hours, et cetera. But what other things do you imagine displaying? Pictures of your sweets? An our story section? Pictures of the owner and the store? I think I would like to have an our story section for sure. Something that gives the customers an idea about our values and quality. I think I would also like to list a recipe of the week on the home page as well. This might be a cupcake, cake, cookie, or some other sweets recipe that I tried out for the first time that week. I think it should also list the price of the item featured somewhere in that section. Ok, what other pages do you think you will want on your site? A tab for a list of ingredients that I use would be nice. Another page for events that we have coming up at the shop. Oh! Maybe we could have a link to a calendar somewhere that lists those events as well? I would also like a way for the customers to place online orders, and maybe create a profile so that they can easily login and place a re-occurring order?

Fig. 2. Interview transcript.

escorts the stakeholders out of the building, the new text mining tool pulls the transcript from its file folder and begins parsing it for requirements. For the purposes of this example, the customized key words for Diane's project are: cake, cookie, baking, bakery, cupcake, icing, sweets, ingredients, recipe, prices, and Sandra's Sweets. Diane chose these words because she knew that the new client was interested in a website for her bakery, and these were words she assumed would be spoken throughout the meeting. Figure 3 is an example of the list that is waiting for Diane when she returns to her desk. Diane



1. Feel need discuss is want display home page
2. Are planning display bakerys name location hours et cetera
3. Things imagine displaying
4. Pictures sweets
5. Story section
6. Pictures owner store
7. Think like have story section
8. Something gives customers idea values quality
9. Think like list recipe week home page
10. Be cupcake cake cookie sweets recipe tried time week
11. Think list price item featured section
12. Pages do think want site
13. Tab list ingredients use
14. Page events have coming shop
15. Have link calendar lists events
16. Like way customers orders create profile login place order

Fig. 3. Text mining output.

decides that items 2, 4-6, 8-11, and 13-16 are useful. She further elaborates these items and places them in the team's requirements tracking system.

IV. Practitioner Survey

To determine if this technique would be useful for practitioners, a survey[2] was developed and disseminated to current agile development methodology users.

A. Study Setup

We posted the survey on two different online forums for developers. This gave approximately 50 practitioners access to the 10 question survey. These 10 questions do not include the identifying information that was collected, such as name, email address, company, occupation, or years of experience. Also included in the survey, is a brief explanation of the term "JIT requirements." Most research conducted in this area has only included studies involving students or investigations on open source systems [10], [9], [3]. Instead, we collected feedback from developers currently in the industry (practitioners) regarding their experience with JIT requirements, methods used to capture informal requirements, and their opinion concerning the technique we proposed[2]. On average, the 13 respondents have 8.5 years of experience, ranging from 1 year to 39 years, which provides a decent representation of the current workforce. These participants belong to several different companies and maintain varying occupations, which provides an array of viewpoints. Their professions include, but are not limited to, software developer, project manager, and chief technology officer, and 8 different companies were represented.

B. Survey Response and Feedback

The responses that provided the most useful feedback were associated with questions 3, 6, and 8. For question 3, the respondents were asked to indicate whether they use JIT requirements in their work. Since 11 out of 13 practitioners responded "yes" to this question, we are confident that JIT requirements are fairly common in closed source projects as well as open source projects. Question 6 asked practitioners to check or list some techniques or tools they use to capture informal requirements. This helped to confirm that the techniques listed in Section II were valid. Finally, question 8 asked practitioners to rate the automated audio mining technique, and the responses were distributed in the following manner: 8% (1 out of 13) responded Extremely Beneficial, 31% (4 out of 13) responded Somewhat Beneficial, 23% (3 out of 13) responded Might be Beneficial, 0% (0 out of 13) responded Not Beneficial, and 39% (5 out of 13) responded not applicable (N/A). This indicated that most companies would find our technique beneficial for their agile development practice.

[2]Survey questioner and collected data are available at: https://figshare.com/s/6b230ec796d89d006025

V. Discussion

On JIT Requirements in Closed Source Software. Although the term "JIT requirements" seems to be unfamiliar, 85% (11 out of 13) of the practitioners revealed that they had worked with requirements that could be categorized as JIT type. With much of the existing research on JIT requirements focusing on the open-source software development paradigm, our survey responses provides an important insight. Our study suggests that JIT requirements are not predominantly an open source phenomenon because they are also encountered in closed source agile methodologies. Such a finding indicates the necessity of in-depth studies in JIT requirements for closed source software.

On Our Framework. We designed this framework to complement note taking when capturing requirements during face-to-face meetings. Of our respondents, 62% agreed that a tool realizing the framework would be beneficial to some degree. One participant mentions "ideas, brought up during discussions, often get lost due to lack of follow-up," rating our framework as "somewhat beneficial". Another participant indicated that the proposed framework "might be beneficial," with a comment that stated "seems like the themes (of discussion) would be obvious by keywords". One of the participants, suggested an add-on or plug-in with this capability, for existing requirements management systems, instead of a stand-alone application. One practitioner expresses her concern: ".... be very expensive (as in, you pay someone to manually transcribe it)....". To that end, we found Dragon NaturallySpeaking Premium [15], which is very reasonably priced, can address this practical concern. Thus, we noticed that our framework received mostly positive and constructive feedback from experienced practitioners.

Limitations. The requirements engineer should analyze the candidate list of loosely formulated requirements (in terms of verbs and nouns) in a timely manner, specifically, while information discussed during the meeting can be easily retrieved from memory. Otherwise, there is a risk of missing the context in which those key words were discussed. It is a major limitation of our framework. Furthermore, before the meeting, each participant should give consent for the conversation to be recorded. Some participants may not be comfortable with such an agreement, thereby limiting the applicability of the framework. In addition, our study consisted of only 13 participants, which imposes further limitations on our findings. However, the study was conducted using professional developers with diverse experience; therefore, we expect our findings to be reliable and meaningful.

VI. Future Work

It is our intention to develop and deploy an integrated tool realizing the framework. The process of breaking down the transcript into meaningful action items is still primitive and needs further refinement. Also, the additional functionality to allow developers to indicate key words that must be kept, no matter what the context is, also needs further investigation. Finally, before integrating with an audio-to-text conversion tool, further research is needed to determine if Dragon NaturallySpeaking would be the most effective option.

Acknowledgment

We express our sincere gratitude to the practitioners who participated in our survey and provided insightful feedback.

References

[1] B. Choudhary and S. K. Rakesh, "An approach using agile method for software development," in International Conference on Innovation and Challenges in Cyber Security (ICICCS), Noida, 2016, pp. 155–158.

[2] T. A. Alspaugh and W. Scacchi, "Ongoing software development without classical requirements," in International Requirements Engineering Conference (RE), Rio de Janeiro, Brazil, July 2013, pp. 165–174.

[3] N. A. Ernst and G. C. Murphy, "Case studies in just-in-time requirements analysis," in International Workshop on Empirical Requirements Engineering (EmpiRE), Chicago, IL, 2012, pp. 25–32.

[4] I. Inayat, S. S. Salim, S. Marczak *et al.*, "A systematic literature review on agile requirements engineering practices and challenges," Computers in Human Behavior, vol. 51, pp. 915–929, 2014.

[5] A. Knauss, D. Damian, and K. Schneider, "Eliciting contextual requirements at design time: A case study," in International Workshop in Empirical Requirements Engineering (EmpiRE), Karlskrona, 2014, pp. 56–63.

[6] Z. R. Stephenson, J. A. McDermid and A. G. Ward, "Health modelling for agility in safety-critical systems development," in Institution of Engineering Technology International Conference on System Safety, Savoy Place, London, UK, 2006, pp. 260–265.

[7] R. L. Q. Portugal, J. C. S. do Prado Leite, and E. Almentero, "Time-constrained requirements elicitation: reusing GitHub content," in Workshop on Just-In-Time Requirements Engineering (JITRE), Ottawa, 2015, pp. 5–8.

[8] P. Heck and A. Zaidman, "Quality criteria for just-in-time requirements: just enough, just-in-time?," in Workshop on Just-In-Time Require- ments Engineering (JITRE), Ottawa, 2015, pp. 1–4.

[9] T. Bhowmik and S. Reddivari, "Resolution trend of just-in-time requirements in open source software development," in Workshop on Just-In-Time Requirements Engineering (JITRE), Ottawa, 2015, pp. 17–20.

[10] T. Bhowmik, N. Niu, A. Mahmoud, and J. Savolainen, "Automated support for combinational creativity in requirements engineering," in International Requirements Engineering Conference (RE), Karlskrona, 2014, pp. 243–252.

[11] R. Raisinghani, R. Riggen, and K. Ryan, "Adopting an agile methodology requirements-gathering and delivery." 2014, PricewaterhouseCoopers. Retrieved from: https://www.pwc.com/enUS/us/insurance/publications/assets/pwc-adopting-agile-methodology.pdf

[12] G. Lucassen, M. Robeer, F. Dalpiaz *et al.*, "Extracting conceptual models from user stories with Visual Narrator," Requirements Engineering, 2016, pp. 1–20.

[13] M. Gall and B. Berenbach, "Towards a framework for real time requirements elicitation," in International Workshop on Multimedia Requirements Engineering, 2006, pp. 4.

[14] D. A. Hristache, C. E. Paicu, and M. M. Dobrescu. "The study of communication paradigm from the perspective of active listening," Theoretical and Applied Economics XXII, 2015, pp. 13–18.

[15] Amazon. "Amazon.Com: Dragon NaturallySpeaking Premium 13 Blue- tooth (Wireless)," Amazon.com, 2017, Web. 24 Apr. 2017. Retrieved from: https://www.amazon.com/NaturallySpeaking-Premium-Bluetooth-Wireless-English/dp/B00LX4C4W4

[16] Nuance. "Using the Dragon Remote Microphone application with Dragon NaturallySpeaking," Nuance.custhelp.com, Web. 24 Apr. 2017, Retrieved from: http://nuance.custhelp.com/app/answers/detail/aid/6294//using-the-dragon-remote-microphone-application-with-dragon-naturallys\peaking

[17] C. Fillmore, "The case for case," Universals in linguistic theory, E. Bach and R. Harms, Eds. New York: Holt, Rinehart and Winston, 1968, pp. 1–188.

[18] A. Parrish, "Natural Language Basics with TextBlob: Reading and writing electronic text," Rwet.decontextualize.com, Decontextualize, Web. 31 May 2017, Retrieved from: http://rwet.decontextualize.com/book/textblob/

[19] Clips, "Penn Treebank II tag set CLiPS," Clips.ua.ac.be, Web. 31 May, 2017, Retrieved from: http://www.clips.ua.ac.be/pages/mbsp-tags