

Easycomm: A Framework and Tool to Solve Client Communication Problem in Agile Development

Tarek El-Najar, Imtiaz Ahmad, and Mohammad Alkandari

Abstract— Agile methodologies rely completely on effective and continuous communication between clients and development teams, however, failing to sustain such communication will result in project failure or overdue. Current available communication methods and tools are either costly, not effective or not designed for Agile software development process (SDP). This paper introduces EasyComm communication management framework that aims at solving client communication problem in agile development projects. The framework uses famous social network websites looks and feel in-order to include users that would normally get excluded from the software development phases due to their technical knowledge. EasyComm provides a centralized open-communication hub that allows both developers and clients to communicate regardless of their geographical locations or technical level. Several interviews with Agile development experts were conducted to validate the effectiveness of EasyComm. All interviewees liked EasyComm and would consider using it in their upcoming projects. Validation results showed that the effect of miscommunication is greater in the requirement elicitation (RE) phase, less effect during the development phase, and medium effect in the testing or release phase. Results also showed that face-to-face meetings and instant messaging applications (IM) were ranked as the best and worse communication methods, respectively.

Index Terms— Agile, Communication, Communication Problem, Communication Tool, Software Development Process.

I. INTRODUCTION

Agile SDP eliminated the need for excessive documentation and reduced the time wasted on finalizing all project's requirements before the team starts working on it. Agile SDP achieved this by relying on continuous communication between developers and client during all project phases [1, 17]. There are four communication layers in each Agile cycle, requirement elicitation, prioritization, negotiation and feedback [2]. The dependency on communication with client created the client

communication problem; failing to sustain and keep effective and continuous communication in any layer can lead to project failure or at best can cause project overdue [3] because it is one of most important success factors of Agile projects [1].

Agile methodologies assume that customers are available for day-to-day project activities which is almost impossible for many clients, which leads to incomplete and inaccurate requirements [4]. Traditional communication methods depended on physical meetings and interactions between the client and the development team either by assigning a dedicated employee to work with the development team or by giving a regular employee this responsibility on top of his/her regular work responsibilities. Both approaches are hard to achieve in most environments; for many customers, hiring a dedicated employee to work with the development team is costly approach that will not be accepted, and assigning such responsibility to regular employees adds availability and cost bottlenecks to the project [2].

In big organizations, project knowledge and information can be scattered and distributed among many employees [14]. Failing to include any of these employees can lead to project delays or project failure in the worst case.

Reasons behind poor communication in Agile based projects can be summarized as follows [2].

- Availability of client representatives: when client and development team cannot discuss urgent project tasks or issues due to client availability.
- Geographic location difference between clients and development teams: when it is impossible to have an onsite customer or to have an urgent meeting with the client because of the geographic location differences specially in global software development (GSD) environments [16].
- Change of user groups responsible for the project: when the responsible people of the project from the client's side get changed multiple times then it can lead to a lot of change requirements that were missing from previous communication sessions.
- Exclusion of some key users from the communication in different layers: when the actual users who will be using the system are not present in the project communication sessions.
- Informal communication: when communication is not properly documented and not all parties are aware of what happened in other communication sessions or meetings with the other party.

Manuscript received December 11, 2017; revised October 17, 2018.

Tarek EL-Najar is with the Computer Engineering Department, Kuwait University, Kuwait P.O. Box 5969, Safat 13060 e-mail: tarek.el.najar@gmail.com).

Imtiaz Ahmad is with the Computer Engineering Department, Kuwait University, Kuwait P.O. Box 5969, Safat 13060 e-mail: imtiaz.ahmad@ku.edu.kw).

Mohammad Alkandari is with the Computer Engineering Department, Kuwait University, Kuwait P.O. Box 5969, Safat 13060 e-mail: m.kandari@ku.edu.kw).

There are two types of solutions available, informal face to face meetings, and software tools [2]. Different solutions are available under each type like the Group Solve that uses an enhanced technique for face to face meetings to insure all information have been acquired by the developers, and WikiWinWin that uses both Wiki and WinWin frameworks to organize the communication between clients and developers in a way similar to Wiki pages [2].

Current available solutions are either costly, require users with technical backgrounds, hard to use or doesn't cover all project phases [2]. The proposed solution aims at fixing all the reasons behind poor communication problem mentioned above while being easy to use, require very little to no technical experience, covers all project phases and not costly in terms of time, money or other resources. It uses Facebook's looks and feel in-order to require little to no prior training or technical background to use it. EasyComm is an enhancement over Winbook which was built to support requirement elicitation phase in non-Agile projects.

This paper introduces EasyComm framework that aims at solving the client communication problem. Section II discusses the related work. Section III discusses how EasyComm follows the Winbook framework and how they are different from each other. Section IV discusses EasyComm and its components. Section V describes how EasyComm works and how it will be used. Section VI compares EasyComm to other solutions. Section VII discusses the validation process and its outcomes. Finally, conclusions and future work are described in Section VIII.

II. RELATED WORK

A. WinWin Framework

This framework was introduced as a requirement capturing and negotiation framework [5, 6]. The framework facilitates the following:

- Location independent requirement capturing and negation.
- Allow easy and convenient way to communicate their problems, issues, concerns and problems.
- Communicating solutions and ideas for raised issues and concerns.
- Making sure that requirements are agreed upon by all users.
- Requirements-Conflict management.

As shown in Figure 1, the framework works by following set of rules and steps for requirement capturing. Steps can be summarized as follows:

1. Project is created by the Project owner and all members get assigned to it.
2. Project taxonomy is created and is accessible by all users.
3. Users start adding requirements as Win Conditions.
4. All users review the added win conditions and raise issues when needed.
5. All users review the raised issues and offer solutions as options.
6. Users agree on proposed options to proceed.

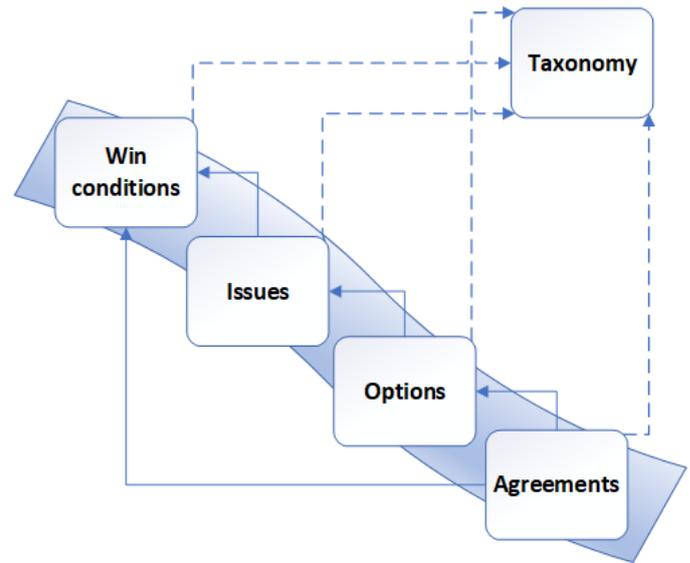


Fig. 1. WinWin artifacts.

WinWin framework was adopted by many tools and other frameworks. Some frameworks enhanced it like EasyWinWin that integrated group productivity and collaboration tools [7]. Following sections discuss two of those tools which are Winbook [5, 15] and WikiWinWin [6].

B. Winbook

Although that there are some researches that studied the possibility of using popular social network websites like Facebook as a RE tool to benefit from the fact that almost everyone knows how to interact with these websites and how to use them [8] because of their prevalent popularity [15, 18]. However, these websites are not designed to function as a RE tool and therefore they lack a lot of features that both clients and development teams will require [2]. Winbook solved the issue of utilizing existing social media framework by integrating the framework's features into a specialized RE tool.

One of the major causes of software failures is the exclusion of non-technical users from requirements elicitation and negotiation processes, and Winbook was designed to solve this issue. Winbook is an RE tool that is based on WinWin framework [5]. The authors were inspired by the WikiWinWin toolset which is based on WinWin framework as well [5].

After studying user interaction with two popular platforms (Facebook and Gmail) that almost everyone is using and comfortable with, they integrated some of features and ideas of these platforms into Winbook. By integrating those features, Winbook overcame the problem of excluding non-technical users [8]. Some of the features that were integrated into Winbook are the way of adding conditions and commenting on them, user profiles and the color-coded labels [2].

Winbook uses a special user role called "Shaper" that is responsible of shaping the discussions, take care of overlapping artifacts, shortening wide discussions, rejecting and agreeing on win conditions. Winbook was not designed for Agile methodologies, however, the authors suggested

the possibility of changing conditions to user-stories along with some other modifications to work with Agile.

III. EASYCOMM OVERVIEW

Although EasyComm follows the Winbook framework, more enhancements and modifications had to be added so that the framework could be applied to Agile SDP, as shown in Figure 2. EasyComm uses user-stories that describe user needs and vision better than the traditional long complicated requirements. In order to cover the iterative nature of Agile development process, EasyComm had to employ a type of grouping for user-stories to track each one according to its stage or status in the framework itself. EasyComm deals with the customer in a different way than Winbook, where it defines a complete team from members of the customer's side where each member has a specific role, rather than dealing with unspecified members or a single person. The introduced framework also extends the adaptation of social media websites by introducing features such as groups, user tagging and hashtags which makes the new framework even easier to use by non-technical users. EasyComm also updated the interface of Winbook system in order to comply with the new changes and updates, as shown in Figure 10 in Section IV.

The following points summarize the differences between EasyComm and Winbook which will be discussed in following section in more details.

- EasyComm deals with two main teams: the customer and the development team.
- EasyComm defines a role or multiple roles for each member of the customer team where each role determines how the system will interact with that member.
- EasyComm employs user-stories instead of Win conditions.
- EasyComm uses a predefined template for user-stories to enhance the quality of produced user-stories and to make it easy for non-technical users to use the system without much knowledge about user-stories and how to write one.
- On top of the regular project categories available in Winbook, EasyComm introduces relevance types which let the user choose the relevance of his/her user-story to one or more of the defined relevance types such as (user interface, technical, etc.).
- EasyComm uses groups like the ones available on Facebook and other social media websites. Each group has its own wall. EasyComm has 4 types of groups: Backlog, Ready, In-Progress and Finished.
- EasyComm employs user tagging similar to the tagging functionality of social media websites such as Facebook. This functionality allows users to tag other users in their comments or posts to notify them directly and ask for guidance, advice or opinion.
- EasyComm uses a visual dictionary that allows descriptive media files to be attached with dictionary entries of business or development-specific words and definitions that one of the teams may not be aware of, which can provide more information about the entry.
- EasyComm helps in producing green software.
- EasyComm covers more project development aspects as it is a real-time collaboration framework that allows both teams to collaborate on the project in its different phases, starting from requirements gathering up to delivering and discussing prototypes.

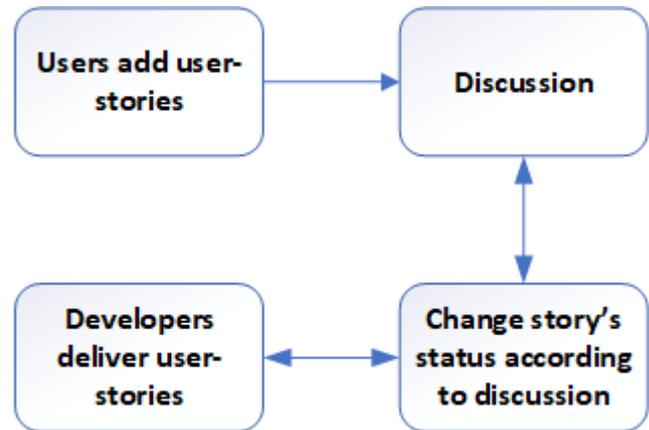


Fig. 2. EasyComm overview.

IV. EASYCOMM

EasyComm is a framework that applies the Winbook framework on Agile development environment, as well as covers more project phases and layers. EasyComm has two main teams assigned for each project: the customer team and the developers' team. Each team member has a single or multiple roles that determine the member's contribution to the project and how the system will interact with each member.

One way of balancing the work is to assign the right task to the right person. Bothering non-technical users with technical issues or technical team members with design or legal issues will only create a more stressful environment for everyone and will result in incomplete or inaccurate requirements. For this reason, EasyComm is a role-based framework; all levels and actions are tied to specific roles.

A. Customer Team

The authors in [9] proposed that, in Agile development projects, a single client representative was not enough and that a single person would be overwhelmed by the number of tasks and responsibilities; they showed how successful Agile projects had a customer team, instead of an individual representative. A customer team consists of 9 different roles where each role has specific responsibilities and is part of the project. Following are the nine roles.

- Geek Interpreter: A person who can help customers understand programmers.
- Technical Liaison: A person who knows customers' infrastructure very well.
- Political Advisor: A person who helps in identifying key political people.
- Acceptance Tester: A person who makes sure that delivered work meets client expectations and requirements.
- (UI) Designer: A person who makes sure the delivered UI fulfills customer expectations.
- Technical Writer: A person who is responsible for all technical writings and documents.

- Diplomat: A person responsible for communicating his/her department's requirements.
- Super Secretary: A person responsible for managing administrative tasks.
- Negotiator: A representative of the client with the development team who communicates with both teams.

1) Excluded rules:

All previously mentioned roles were presented in traditional physical communication-based projects. However, EasyComm is a software framework, and thus a software tool can replace some of these roles and make others not necessarily needed. Excluded rules include super secretary, political advisor, geek interpreter and diplomat.

2) Customer team's roles in EasyComm:

Following are the customer team roles included in EasyComm, along with a brief description of why they have been included in the framework and their functions in the system.

a) Technical liaison:

Software projects must deal with the existing technical infrastructure of the organization at one point or another [9]. Whenever the development team have some questions or concerns that are related to the technical infrastructure of the project, they ask users with the technical liaison role from the customer team. Technical liaison should have complete knowledge about the organization's technical infrastructure and should be able to help the development team make better decisions and deliver a software that meets the client's expectations as well as complies with their technical infrastructure.

EasyComm will notify all users with the technical liaison role whenever the development team raises an issue that is technically relevant; this will insure that all technical issues have been directed to the right person which will help keep the project going by reducing the time needed to get the right answer from the right person.

b) Technical writer:

Technical writer is added to the excluded roles section because this role will not have an effective role in the system; however, EasyComm will require this role to be defined. EasyComm will notify all users with this role whenever a requirement/user-story status is changed to finished so they can start adding the feature to their technical documentations.

c) Acceptance tester:

Acceptance testers are team members who are responsible for testing the delivered software or prototypes to ensure that all requirements are working as per the customer's vision [9]. In EasyComm, all users with this role will be notified whenever the development team post a new release so they can test it and respond with acceptance or refusal.

d) User interface designer:

Design is an essential part of any system as it is the first thing the user sees and it determines how end-users will use the system [9]. This user may provide the actual designs or

be responsible for accepting the designs provided by the development team. It is an essential role in the system that will be responsible for accepting all design-related releases to insure they meet the client's expectations. EasyComm will notify all users with this role whenever the development team post any new design-related release or prototype.

e) Negotiator:

Negotiator is usually an on-site customer who can decide which user-stories are important and the order in which they should be added to the system; also, he/she is the one whom developers usually talk to regarding their business/system issues [9].

In EasyComm, the negotiator role acts just as described before, but it will not require the user to be on-site; the negotiator will give the green light for any user-story to indicate whether it should be removed, delayed or accepted. The negotiator role will help filter and prioritize the requirements to insure that only valid, needed and important requirements are added to the system. The negotiator will have the right to remove or accept any user-story added by the customer team in the Backlog group; also, this role will prioritize the stories inside the Ready group, as shown in Figure 6.

3) Development team roles in EasyComm:

Same as the client's team, there are special roles introduced by EasyComm for the development team. First role is the shaper, which was already introduced in Winbook, and the second role is the green software executive.

a) Shaper:

Shaper is a special role in the development team; a user with that role is responsible for accepting, denying or suspending user-stories according to the development team's vision, limitations and capabilities. The shaper also has other responsibilities designated in Winbook framework which were discussed in previous sections.

b) Green software executive:

According to the nature of the project, the client's team or the development team, a green software may be a requirement. EasyComm introduces an executive role for green software engineering in the development team. This role is responsible for insuring that all requested requirements and the development process will produce a green software. Regardless of the methods applied for green software engineering and development, EasyComm facilitates the process of monitoring all requirements and the development process for the green software executive by giving any user with that role the ability to approve, raise issues and post options on posts provided by both teams. Any user from the development team can have the green software executive role even if that user has other roles in the system like a project manager or a shaper.

B. User-Stories

User-stories are used in Agile-based projects instead of regular requirements or use-cases because they define how users will use the system and how they think it should behave in a story-based form, instead of the ambiguity and

complexity of requirements [10]. EasyComm uses one of the simplest available formats for user-stories introduced by Mike Cohn [10], as shown in Figure 4. The introduced format consists of three parts: the role, the goal and the purpose or the benefit. Mike’s format forces users to add meaningful stories even if they do not have technical backgrounds or if they do not know what a user-story is or how it should be written; the introduced format is: As a <User Role>, I want to <Goal/Desire>, so that <Benefit>.

1) *User-Story building blocks:*

For a user-story to be complete, it should consist of three main parts: the user role, the story goal or desire and the benefit of that story.

a) *User role:*

User role is one of the main building blocks of the user-story format in EasyComm. User role was introduced to replace the ambiguous user “the user” that have been used in all user stories as if there will be only one user who uses the system, i.e. the user can access the posted documents. By employing the word “user” in a user-story gives the impression that all users have the same rights and goals in the system and gives a lone perspective for all user-stories. Employing proper user roles shows exactly how this user will be part of the system; for example, in a job search website, a job seeker role will describe how a user who wants a job will use the website, while an employer role will describe how employers will use the system and post job ads.

EasyComm lets the user select a user role from a predefined list of roles that have been entered by the administrator or the negotiator in the project initialization phase, as shown in Figure 4. This will help users select the appropriate user role for their story without the fear of duplicating roles or employing the general user role.

b) *Goal:*

A story should have a goal in order to be clear and have a purpose in the system. EasyComm makes it easy for non-technical users to write their user-story’s goal by providing fill-in fields that specify how the user-story should be formed, as shown in Figure 4.

c) *Benefit:*

The benefit part of the format describes how the user-story is beneficial for the system which may help in prioritizing the user-stories after their approval.

C. *EasyComm Groups*

EasyComm is based on Winbook that uses a social media framework to ease usage and to require less learning time for a user to start using the system. EasyComm extends the adoptability of the social media framework by adding groups that are similar to Facebook groups. Each group will have its own wall that will have all groups’ related posts on it available for all users to check and interact with.

Groups might not have been needed in Winbook as it only handles requirement elicitation, but EasyComm handles more project phases in an Agile-based environment which requires user-stories to be categorized and differentiated according to their status in the project.

One type of Agile methodologies is Scrum, in which the product owner adds user-stories to the backlog and prioritize them, and then the development team pick the most important user-story from the backlog and start working on it in a sprint that lasts for a certain amount of time, as shown in Figure 3. During the sprint, the development team might have some questions or require some clarifications which they ask the product owner to clarify and answer. After each sprint, the product owner reviews the outcome and then the development team moves to the next user-story. In Scrum, a user-story moves from the backlog to a sprint and finally to the review.

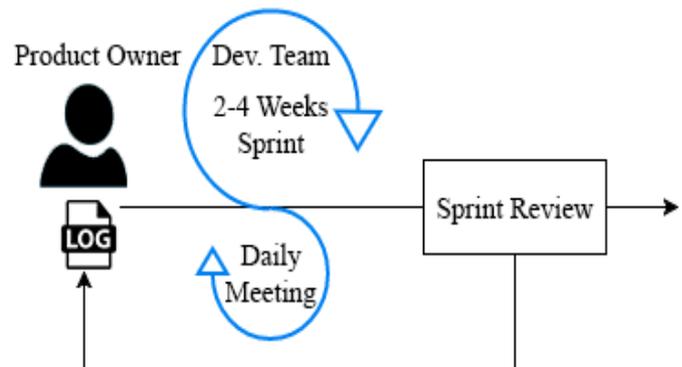


Fig. 3. Simplified Scrum Diagram.

EasyComm has 4 types of groups: Backlog, Ready, In-Progress and Finished. EasyComm groups have been

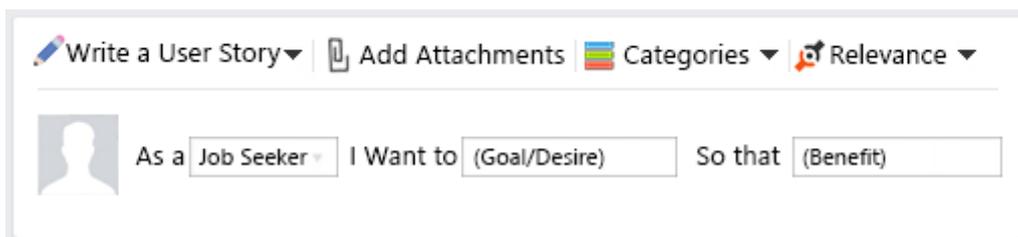


Fig. 4. EasyComm adding new user story screen.

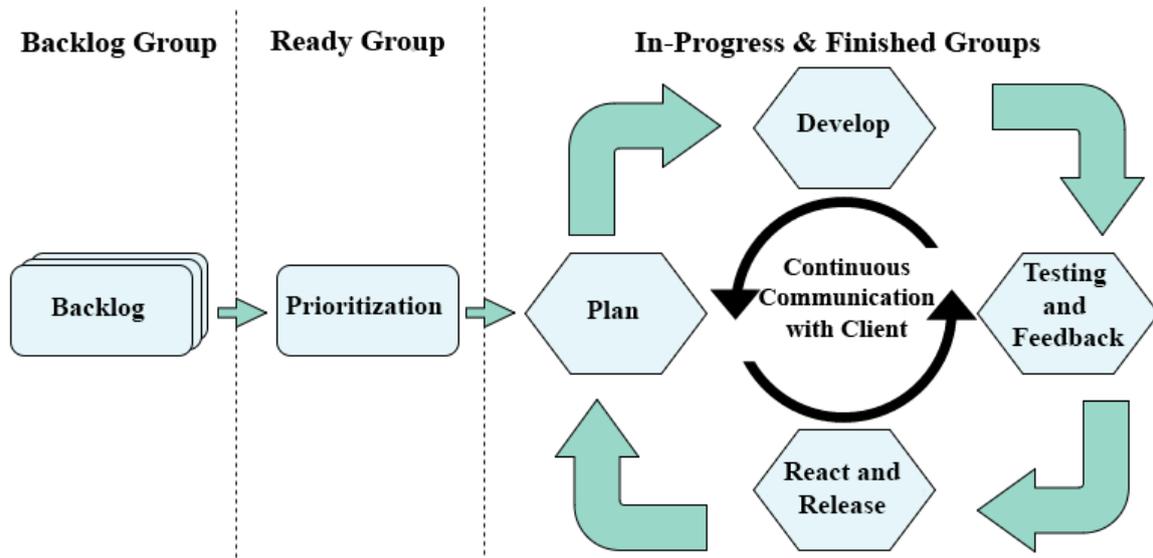


Fig. 5. EasyComm groups and agile lifecycle.

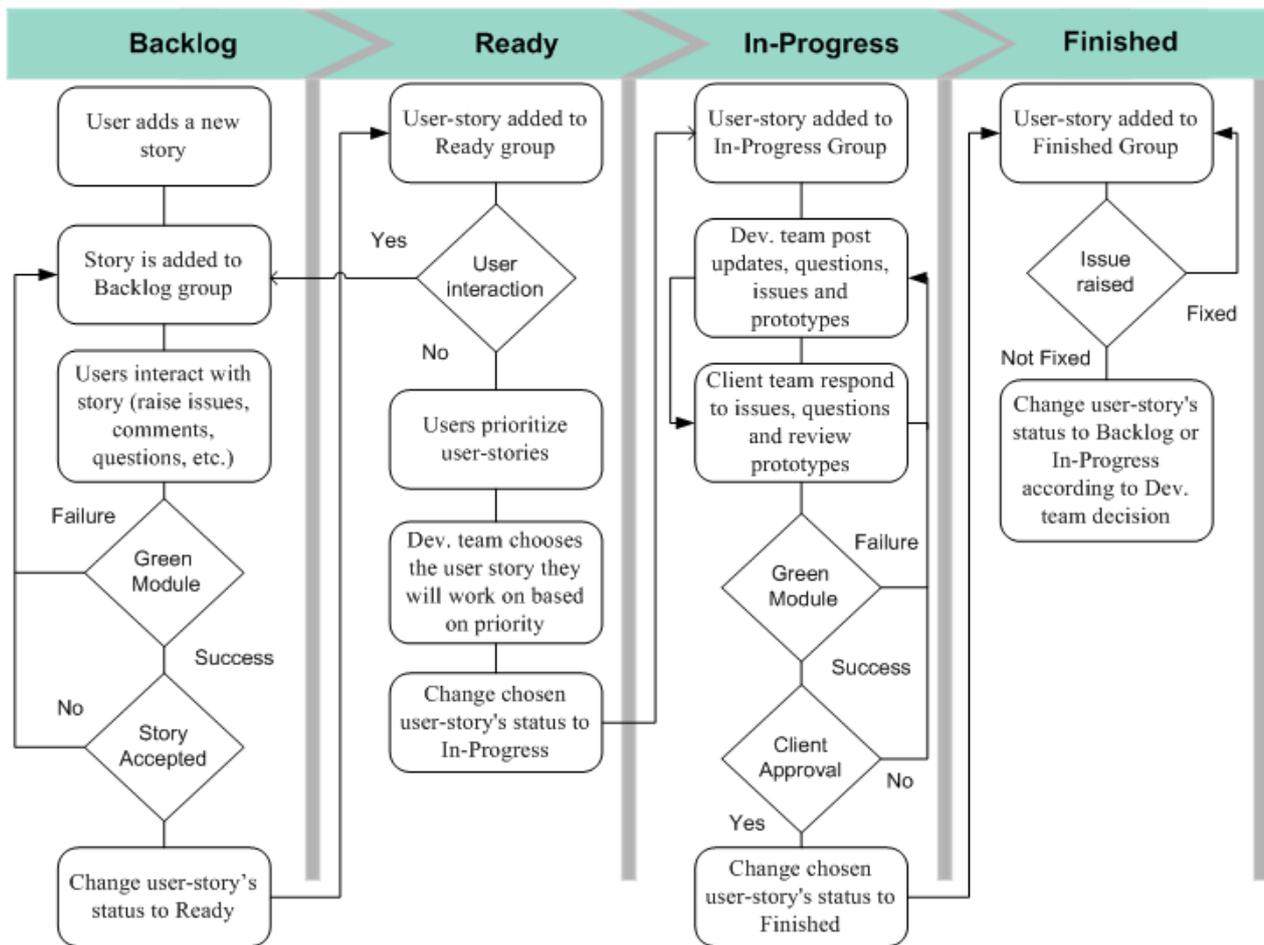


Fig. 6. User-story movements in EasyComm groups.

selected to comply with the general lifecycle of the Agile development process, as shown in Figure 5. With a design that follows the general lifecycle of the Agile development process, EasyComm should be usable with all Agile methodologies.

User-stories move between the groups as users interact with them and change their statuses, as shown in Figure 6. Users can access each group and check all stories inside it and interact with them, as shown in Figure 6. Groups' definitions and purposes will be discussed further in the paper.

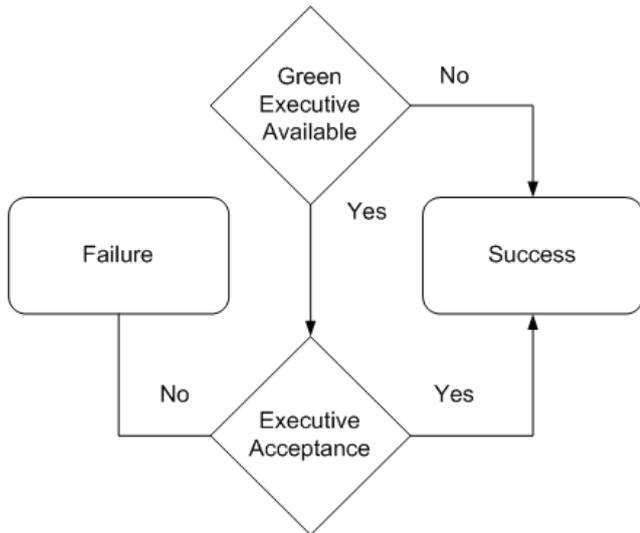


Fig. 7. Green module test in EasyComm.

1) Backlog:

Backlog is the main group in EasyComm and it holds all new entries by the client's team. The client's team add all required user-stories, answer all questions and issues, and provide options for issues raised either by their team members or the development team. After both teams agree on the user-story, negotiator acceptance will be needed to move the user-story to the Ready group. In case the green SE executive is involved in the project, a user-story must have the executive's agreement also to be moved to the Ready group, as shown in Figures 6 and 7.

2) Ready:

Ready group holds all user-stories that have been marked as ready. All ready user-stories are agreed on by both the client's team and the development team. Both teams will

have the ability to raise issues or ask questions on all ready user-stories. Whenever an issue is raised on a user-story that cannot be fixed, it moves back to the backlog group and all users get notified.

All users with the negotiator role can prioritize the user-stories to determine which ones should the development team start working on first according to their importance and relevance to the system and the client. After the prioritization process, the development team move the user-stories from the Ready group to the In-Progress group whenever they start working on the user-story; in a Scrum-based project, this will mean that the development team have started a sprint on this user-story.

3) In-progress:

In-Progress group holds all the user-stories that the development team are working on; the development team might have multiple sub-teams working on different tasks concurrently. Users will be able to check progress on all in-progress tasks as the development team will be adding progress updates, screenshots, videos or prototypes on each in-progress user-story. Development team members can ask questions or require clarifications from the customer's team members while they are working on a particular user-story. In-Progress group simplifies communication between developers and client team members by removing availability, location and cost constraints that would have delayed the delivery of the user-story if communication had been made using traditional methods. When the negotiator approves the submitted work, the user-story's status changes to "finished" and it moves to the finished group. In case the green SE executive is involved in the project, the submitted work must be approved by the executive first before being approved by the client to ensure that the submitted work will result in a green software according to the followed standards, as shown in Figures 6 and 7.

4) Finished:

Finished group holds all finished user-stories that have been submitted by the development team and approved by the client's team. All users can access this group and view all finished stories along with any submitted prototypes, screenshots, videos or attachments.

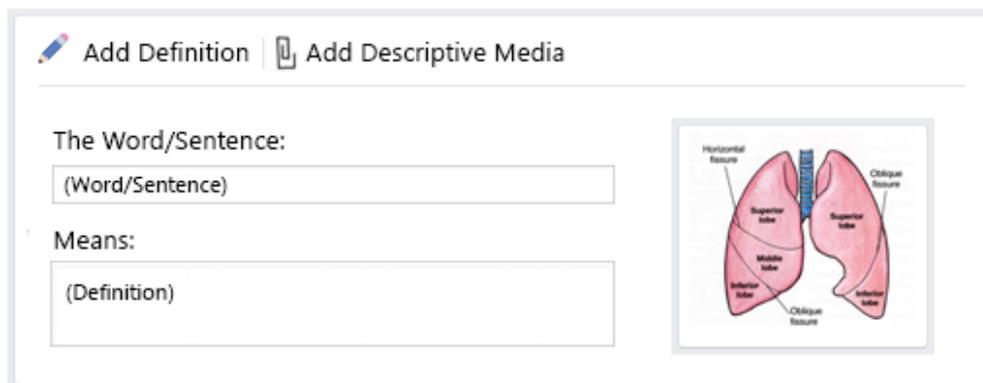


Fig. 8. EasyComm new visual dictionary entry.

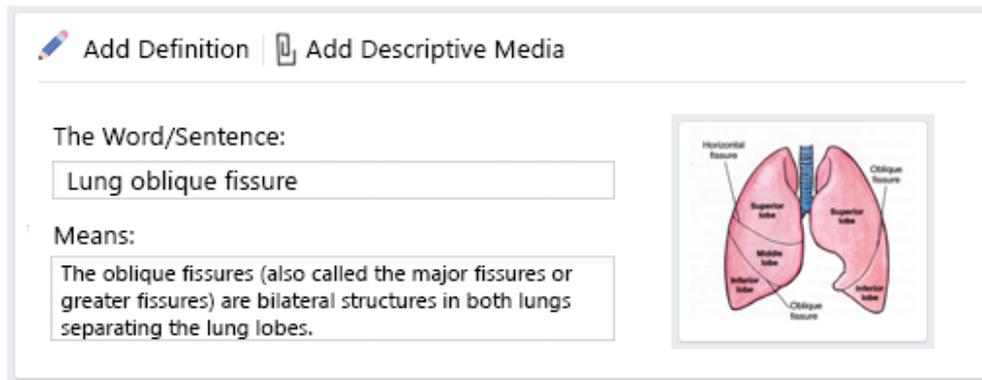


Fig. 9. EasyComm visual dictionary entry example.

D. Visual Dictionary

Both clients and developers are almost from two different worlds; both teams work in different businesses, environments, and professions. Each business and profession has its own set of words and definitions that are not present in other types of businesses or professions. In order to facilitate communication between developers and clients when both use their own set of words and definitions, EasyComm uses a visual dictionary.

Visual dictionary allows descriptive media files to be attached with dictionary entries of business- or development-specific words and definitions that one of the teams may not be aware of, which can provide more information about the entry. During the initial project set-up process, the administrator can add some dictionary entries that were gathered in the project analysis phase. Users can also add dictionary items as they use the system whenever they employ business- or profession-specific words inside their posts. Explaining a new definition follows the same concept of adding a new story; EasyComm uses a format that facilitates the addition of new definitions, as shown in Figures 8 and 9.

E. EasyComm and Green Software Development

Some may think that traditional communication methods do not affect the environment or people themselves. However, this means that they are not looking at the whole picture; they are not considering all the wasted energy and pollution resulting from transportation or other means of traditional communication. Also, they are not considering how continuous physical meetings or phone calls are exhausting and stressful for people in Agile-based projects. On the contrary, EasyComm solves this issue by providing a green solution that reduces physical meetings, on-site visits and phone calls to the minimum which in turn decreases wasted energy and pollution produced by these methods. Additionally, it is a web-based system that requires minimum power consumption on users' machines as all calculations and handling are performed on a centric server.

EasyComm also helps in producing a green software by introducing the green software engineering executive role who is responsible for insuring that all the development process and user-stories are following the green software engineering standards. EasyComm fulfills one of the three requirements of a green software according to the definition

suggested in [11]: the engineering process of a software must save resources and reduce waste.

V. USING EASYCOMM

The following sections describe how the EasyComm framework can be used to solve and enhance communication between customer and development teams in Agile-based projects.

A. Setup a New Project

Before using EasyComm in a project, the administrator should first set up a new project. Project set-up includes defining the following points.

- **Project name:** Defining the project name allows users to switch between projects if they are assigned to multiple projects on EasyComm.
- **Categories:** Setting categories at this stage is optional as they might not be clear at this stage of the project; however, setting up some basic categories can help users start using the system right away.
- **Project Dictionary:** Setting up some basic technical and business-related definitions gathered from both teams at the project analysis phase.
- **User roles:** Defining user roles is very important at this stage as it is one of the main components of the user-story template in EasyComm.
- **Development team members:** Defining development teams along with their members allows the framework to notify only the right members according to their corresponding teams.
- **Client's team members along with their roles:** Defining the client's team members along with their roles allows the system to insure that all communications are routed correctly to the corresponding users according to the selected relevance at the creation of the post.
- **Project taxonomy:** Defining the project taxonomy includes defining all project details, rules and infrastructure information. Project taxonomy allows users to get all information about the project.

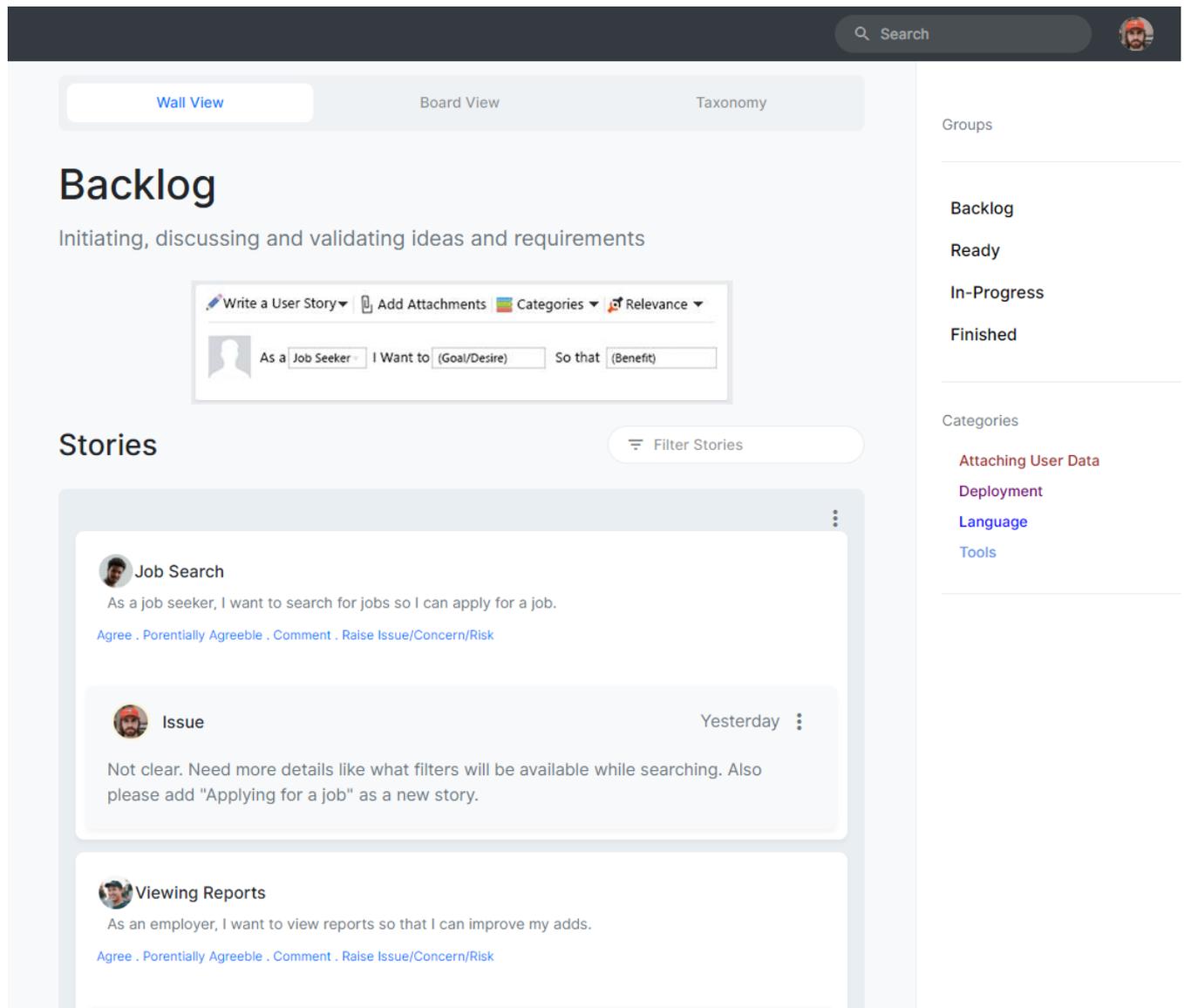


Fig. 10. EasyComm backlog group wall view with groups and categories menu.

- **Project dictionary:** Defining the project dictionary is an ongoing task that lasts until the project ends. Both the client and development teams add dictionary entries that help the other team understand industry- or business-related terms.

B. Adding User-Stories

After setting up the project, the client's team can directly log in the system and start posting user-stories and browsing other posted ones. After login, users will be directed to the backlog wall view as the default page, as shown in Figure 10. Users can post new user-stories directly in the backlog group in order for other users to check and discuss them.

Both teams can access the backlog group to check all user-stories posts, and can comment, raise issues or provide options on any posted story. This type of discussions is similar to that available on social media websites such as Facebook, where people argue about a post by commenting, liking or reporting it.

Users can assign relevance to each posted comment or issue; EasyComm uses relevance to notify users that are relevant to the selected relevance to check that posted issue or comment according to their role in the system; for example, if users want to raise a technical issue or ask a technical question related to the story, users must select "Technical" from the relevance menu of their response so that EasyComm notifies all members with the technical liaison role.

Tagging is one of the options adopted from social media websites including Facebook and Twitter where users can directly tag someone in their response which works just like choosing the relevance; however, tagging allows users to select specific people to get notified, regardless of their role in the system.

For a user-story to be accepted and moved to the Ready group, it should be marked as "agreed" by both the shaper from the development team and the negotiator from the client's team.

C. Managing the Ready Group

After the negotiator and shaper users agree on a particular user-story, it is moved to the Ready group. In this group, user-stories are ready to be prioritized; the negotiator starts prioritizing the user-stories according to their importance to the client and relevance to the project. Team leaders of each development team open the Ready group to select the user-story they are going to work on with their teams. All chosen stories are moved to the In-Progress group where users can keep up with all progress made on each user-story.

Users can still raise issues or post comments on stories in the Ready group. User-stories with unresolved issues can be moved back to the backlog where users can suggest options and whenever they are accepted again they can be moved back the Ready group.

This process insures that the development team work only on important and required tasks that have been approved by both teams.

D. Communication in Development Phase

No matter what Agile methodology the development team use, it will require constant communication with the client during the actual development of user-stories. Once a user-story has been moved to the In-Progress group, it means that the development team are currently working on that user-story. Whenever the development team have a question or an issue while working on a particular user-story, they can directly post a comment with the correct relevance or with user tagging, in order to get direct feedback from relevant client's team members.

The development team can post documents, screenshots, videos or actual executable prototypes or working website-page links and wait for customer feedback. If the acceptance tester approves the submitted work, then the user-story will be marked as finished and moved to the Finished group. However, if the acceptance tester does not approve the submitted work, then the development team can choose between continuing to work on the same story to fix the posted issues, moving the story back to the Ready group or moving the story back to the Backlog group according to the nature of the submitted issues, their solution and project timing and deadline.

E. Last Stage

After the delivered user-story has been accepted, it is moved to the finished group where other users can access and check it. EasyComm will send notifications to all users with Technical Writer roles so they can start writing technical documents for the delivered work.

As shown in Figure 6, finished user-stories can still have issues raised on them; sometimes a change in requirements or environments might occur after a user-story has been finished and delivered, which may require changes or modifications in already-delivered stories.

According to the development team's decision, finished user-stories that have issues raised on them can either be sent back to the In-Progress group where they can start working on them right away, or sent back to the Backlog group where they start the whole process from the beginning but without losing all of their current comments, options, issues and deliverables.

VI. COMPARISON WITH OTHER AVAILABLE SYSTEMS

Comparing EasyComm to other systems like Winbook shows how EasyComm enhanced the communication process between developers and clients; by providing an open, contemporary, easy-to-use and centralized communication channel between developers and their clients. EasyComm minimizes the time needed to start using it, maximizes the number of people participating in application development, and minimizes costs spent on loop backs which helps producing accurate applications that meet client expectations and requirements.

Considering the comparison made in [2] between the available client communication solutions, EasyComm easily takes full point in all criteria; it covers all the comparison criteria as follows:

- Easy-to-use for non-technical users.
- The learning curve is very short compared with other solutions.
- Considered as Formal as it is completely controlled by the companies themselves and no other entity can modify, remove or block any of their communications.

VII. VALIDATION

In order to validate the proposed solution, several interviews have been conducted with people from the software development field. Interview questions were reviewed by an expert in Agile software development and Agile methodologies to prove their effectiveness. Owing to the very small software development industry in Kuwait, interviews were conducted with 30 people in different positions including developers, analysts, managers, IT personnel and sales.

A. Interview questions overview

Interview questions were formed to provide experts' feedback regarding the proposed solution; results should provide information about the following.

- How much misunderstanding, misinterpretation or lack of communication between developers and clients in the project phases below affect the project.
 - Requirement elicitation.
 - Development.
 - Demo, testing and feedback phases.
- How much the exclusion of key users from the project's development phases, owing to their schedules, geographic location, language barrier or technical level, affects the project?
- Types of communication methods used by the interviewees or their teams.
- Effectiveness of the current used communication tools.
- Whether assigning each member of the client's team a specific role that insures that the developer's questions are delivered to the right person in the client's team would help the interviewees communicate better with the client.
- Whether using a tool that forces the client's team members to add their user-stories using the proposed format would enhance user-stories' quality.
- Whether having a tool that facilitates direct contact between the client's team members and developers will reduce the amount of loop-backs.

- Whether the visual dictionary tool is beneficial and useful.
- Whether the way EasyComm handles and groups communication is suitable and beneficial.
- Whether the look and feel of EasyComm that looks like Facebook will make it easier for non-technical users to use.
- Whether the interviewees are interested or willing to use EasyComm after they have known about it.

B. Results and Discussion

The choice of conducting interviews, instead of using online surveys, greatly helped us in obtaining accurate results, along with direct feedback from experts in the software development field that can significantly enhance EasyComm in the future.

Apart from introducing EasyComm, its functionalities and how it can help in simplifying the communication process between clients and developers to help solving the client communication problem, interviews discussed three main topics, effects of miscommunication on different project phases, effectiveness of current communication tools and methods and whether interviewees will consider using EasyComm. Discussed topics are summarized in Tables 1, 2 and 3.

The miscommunication term can refer to multiple types of communication issues like misunderstanding, misinterpretation or lack of communication. Miscommunication affects the project in all phases, however, according to the interviewees, miscommunication during the requirement elicitation phase affects the project the most; starting a project on a foundation based on wrong or missing ideas can only lead to project failure. It is extremely important to insure that both teams communicate their requirements, ideas, concerns and thoughts in a way that is clear and documented to insure a solid project foundation. On the other hand, miscommunication in development and release phases has low and medium effects, respectively. Having a strong foundation in project elicitation phase will result in less need for communication in development phase, while in release phase, communication is important to get client's feedback and do the necessary changes is important but not as important as building the project foundation at the beginning.

Generally, all the interviewees liked the idea of EasyComm as they all stated that they were likely to use EasyComm and give it a try because they thought it could in fact help them overcome the client communication problem. Some interviewees expressed their concerns about the constant and direct communication between clients and developers allowed in EasyComm; they fear that such communication tool can disturb the developers. This concern can be addressed in future research on communication going through communication tools such as EasyComm be regulated in a way that minimizes disturbance for both teams.

The idea of having a dedicated system for communication that is not cluttered with many features and options was one of the points that almost all the interviewees pointer out as a strong point about the proposed solution. However, it raised another concern which is integration with other systems used for other

purposes like internal project management applications. Integration is not covered in this stage but it is in the future work list.

Another point raised by two reviewers was about the resistance of change and the fear of trying something new and different; they mentioned that a lot of managers would not take the risk of trying such tool. However, this applies to any new tool or technology and these types of people will often keep using their old tools until they start losing clients or team members as a result of that.

Interviews' results showed that among all the mentioned communication methods, Instant messaging (IM) applications were the least effective communication method; communication in IM applications tends to interfere with developers' personal lives as they usually use these applications in their personal communications outside of work. Also, there is no proper documentation or history of the communication that takes place on these applications. Interviewees ranked face-to-face meetings as the most effective communication method as it allows them to capture all information they need from the client and communicate their thoughts and ideas to them. Although face-to-face meetings were considered the most effective communication method, some interviewees stated that they waste a lot of valuable time.

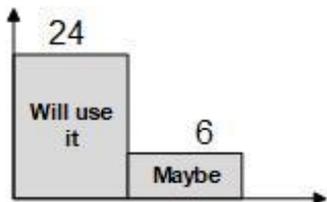
TABLE 1
EFFECT OF MISCOMMUNICATION

Topic	Effect of Miscommunication
Description	According to the interviews results, miscommunication in the requirement elicitation phase affects the project the most, while in development and demo phases, the effects are low and medium, respectively.
Summary	

TABLE 2
EFFECTIVENESS OF COMMUNICATION TOOLS AND METHODS

Topic	Effectiveness of Communication Tools and Methods
Description	IM applications were rated as the least effective and least favorable communication tool while face-to-face meetings were the most effective and favored method despite being costly in terms of time, money and other resources.
Summary	

TABLE 3
CONSIDERATION OF USING EASYCOMM

Topic	Whether interviewees will consider using EasyComm
Description	Interviewees liked the idea of EasyComm and thought that it can help them overcome the client communication problem in their Agile-based projects, however, some of them had some concerns regarding disturbing their development team and whether their clients will be able to use such tool.
Summary	 <p>A bar chart with two bars. The first bar is labeled 'Will use it' and has a value of 24. The second bar is labeled 'Maybe' and has a value of 6. The y-axis is labeled 'Number of interviewees' and the x-axis is labeled 'Response'.</p>

VIII. CONCLUSION

At the cost of solving the issues of the traditional development methods, the Agile model introduced the client communication problem; owing to the structure and nature of Agile methodologies, they depend heavily on communicating with the client. With four layers of communication that get repeated in almost every Agile iteration, failing to handle client communication is one of the main reasons behind the failure of Agile-based projects. In this paper, we introduced the EasyComm framework that aims at solving the client communication problem in Agile development projects. EasyComm follows the WinWin negotiation framework, especially Winbook which uses a social media website framework to be more user-friendly and to provide better and quicker adaptability. EasyComm aims at covering all aspects of communication that involve clients in Agile-based projects, while including all non-technical users in the process of developing their software. EasyComm adopted a social media website framework even more than the original framework did; with adding groups, tagging and visual dictionary, EasyComm will be more familiar and easier to use. The proposed solution follows the basic Agile lifecycle, while supporting green software engineering; however, there is still considerable room for improvements, enhancements and testing that can be done to both the client communication problem and the proposed solution. The proposed solution has been validated by interviewing experts in the Agile software development field, and interviews' results showed that EasyComm can in fact fix the client communication problem and that developers are willing to give it a try and use it in their projects.

EasyComm is still in its early stages and there are lots of room and opportunities for improvements and enhancements. User-stories' quality can be enhanced by following techniques found in [12] that applies 14 quality criteria on user-stories to ensure quality or using natural language processing technique in [13]. Other opportunity to improve EasyComm is to study how EasyComm can be modified to suit different Agile methodology and what features must be added to support each methodology. EasyComm supports green software development and software engineering, however, more research can be made to make it a green framework and tool itself. Last but not the

least, EasyComm doesn't generate any communication related report; research can be made to study the types of reports that can be generated from EasyComm and how they can be beneficial to both client and development teams.

REFERENCES

- [1] Hummel, M., Rosenkranz, C., & Holten, R. (2013). The role of communication in agile systems development. *Business & Information Systems Engineering*, 5(5), 343-355.
- [2] El-Najar, T., Ahmad, I., and Alkandari, M., "Client Communication - A Major Issue in Agile Development", *International Journal of Software Engineering and Its Applications* 10(12) (2016) 113-130.
- [3] Bhalerao, S., & Ingle, M. (2010). Analyzing the modes of communication in Agile practices. In *Computer Science and Information Technology (ICCSIT)*, 2010 3rd IEEE International Conference (Vol. 3, pp. 391-395). IEEE.
- [4] Cao, L., & Ramesh, B. (2008). Agile requirements engineering practices: an empirical study. *IEEE Software*, 25(1), 60-67.
- [5] Kukreja, N., & Boehm, B. (2012). Process implications of social networking-based requirements negotiation tools. In *Software and System Process (ICSSP)*, 2012 International Conference (pp. 68-72). IEEE.
- [6] Huang, L., Wu, X., & Zhang, Y. (2013). WikiWinWin: A Wiki Based System Together with WinWin Method for Collaborative Requirements Negotiation. In *AASRI Winter International Conference on Engineering and Technology (AASRI-WIET 2013)*. Atlantis Press.
- [7] Gruenbacher, P. (2000). Collaborative requirements negotiation with EasyWinWin. In *Database and Expert Systems Applications*, 2000. Proceedings. 11th International Workshop (pp. 954-958). IEEE.
- [8] Seyff, N., Todoran, I., Caluser, K., Singer, L., and Glinz, M. Using popular social network sites to support requirements elicitation, prioritization and negotiation. *Journal of Internet Services and Applications* 6.1 (2015): 1.
- [9] Martin, A., Noble, J., & Biddle, R. (2006). Programmers are from Mars; customers are from Venus: a practical guide for customers on XP projects. In *Proceedings of the 2006 conference on Pattern languages of programs* (p. 20). ACM.
- [10] Cohn, M. (2004). User stories applied: for Agile software development. *Addison-Wesley Professional*.
- [11] Taina, J. (2011). Good, bad, and beautiful software-in search of green software quality factors. *Cepis Upgrade*, 12(4), 22-27.
- [12] Lucassen, G., Dalpiaz, F., van der Werf, J. M. E., & Brinkkemper, S. (2015). Forging high-quality user stories: towards a discipline for Agile requirements. In *2015 IEEE 23rd international requirements engineering conference (RE)* (pp. 126-135). IEEE.
- [13] Lucassen, G., Dalpiaz, F., van der Werf, J. M. E., & Brinkkemper, S. (2016). Improving agile requirements: the quality user story framework and tool. *Requirements Engineering*, 21(3), 383-403.
- [14] Cruzes, D. S., Moe, N. B., & Dybå, T. (2016). Communication between developers and testers in distributed continuous agile testing. In *Global Software Engineering (ICGSE)*, 2016 IEEE 11th International Conference on (pp. 59-68). IEEE.
- [15] Ardini, A., Hosseini, M., Alrobai, A., Shahri, A., Phalp, K., & Ali, R. (2014). Social computing for software engineering: A mapping study. *Computer Science Review*, 13, 75-93.
- [16] Yagüe, A., Garbajosa, J., Díaz, J., & González, E. (2016). An exploratory study in communication in Agile Global Software Development. *Computer Standards & Interfaces*, 48, 184-197.
- [17] Buchan, J., Bano, M., Zowghi, D., MacDonell, S., & Shinde, A. (2017). Alignment of Stakeholder Expectations about User Involvement in Agile Software Development. In *Proceedings of the 21st International Conference on Evaluation and Assessment in Software Engineering* (pp. 334-343). ACM.
- [18] E.Y. Baagyere, Z. Qin, H. Xiong, and Q. Zhiguang, "The Structural Properties of Online Social Networks and their Application Areas," *IAENG International Journal of Computer Science*, vol. 43, no. 2, pp156-166, 2016.