Mansoura Journal For Computers and Information Sciences

Distributed ScrumFall Model: A hybrid software process model that copes with COVID-19 pandemic

Doaa Elghamrawy, Amal I Abou Eleneen, Ahmed Abou Elfetouh Saleh
[1]*Faculty of Computer Information sciences, Mansoura University, Mansoura, Egypt, Emails: doaa.alghamrawy@gmail.com, amalcs2006@yahoo.com, elfetouh@gmail.com*

ABSTRACT
In the past, whenever work starts on software development project a methodology must be selected to be applied to produce the final product. Each methodology has been designed to be applied on specific project's conditions to get the best results. Due to the rapid changes and problems arise like COVID-19 pandemic; there are sudden changes in the working processes and communication methods, which lead most companies resorting to software development from home and changing the work environment from on-site to be distributed. Therefore, it's no longer practical to continue working with the same development process models. Because of that, most of the software industry companies resorted to use the hybrid models that combine practices from more than one methodology in order to obtain the best results and to keep up with the ongoing changes at the work environments. This paper presents a practical model that is inspired from incorporating geographically distributed agile scrum with traditional waterfall model that works well for several software houses. The model has been applied on a distributed organization during the COVID-19 pandemic and shows better results. After analyzing the model results, it shows that the model improves the productivity around 40%, decrease the delivery delay 24%, and decreases number of issues on client environment 46% during the COVID-19 pandemic.

General Terms
Software Engineering, Software Development Life Cycle.

Keywords
COVID-19, agile software development, Hybrid software process model, geographically distributed software development.

1. INTRODUCTION
Traditional development methodologies are now not coordinated with today's commerce situations since they need the adaptability required for most of today's software products [1]. In response to this problem, so called "Agile" software development methodologies have been appeared and put to use. Agile approaches have achieved great success on the industry in the recent years, but they also have drawbacks[2]especially during the COVID-19 pandemic[3,4] and that was a motivation to present solutions for most of problems on the proposed model. In next subsection we will concentrate on the waterfall model and agile scrum model as it is the base of our case study.

1.1 Waterfall Model
The waterfall model is a traditional model used in system development life cycle to create a system with a linear and sequential approach as shown in Figure 1. There are problems with the waterfall model as does not meet the needs of new businesses because the model does not adjust to new change. A working version of the system is often not seen until late on in the project's life. There is where the client is not able to see the project until it is done. It is difficult to integrate risk management .There is no feedback and bad communication between the teams, meaning that they cannot deliver fast.

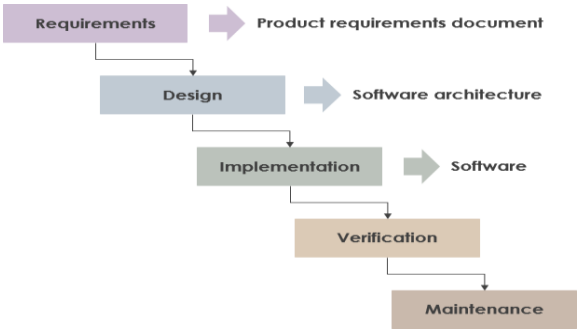

Figure 1:Waterfall Model [24]

1.2 Agile Scrum Model
The Scrum approach is developed to manage the systems development process. The main idea of Scrum is that systems development includes many environmental and technical variables (such as requirements, time frame, resources, and technology) that are likely to

change during the process as shown in Figure 2. This makes the development process unpredictable and complex, which requires flexibility in the systems development process in order to be able to respond to changes [5]. There are some problems with agile model like defects in the understanding and managing the requirements, agile methods have a problem on dealing with distributed teams [2].some of the agile methodologies are more difficult to understand than linear ones.

Based on all the above drawbacks of agile and waterfall models. This paper will present a hybrid model of both Scrum and waterfall with the distributed software development to cover the limitation on the previous attempts and improve the quality of the final product.

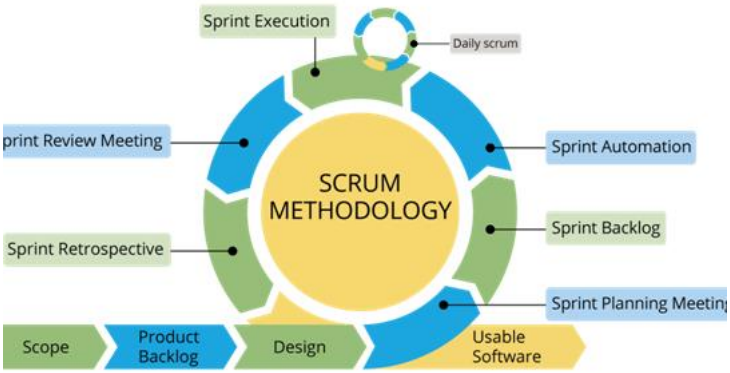

Figure 2: Agile Scrum Model [25]

2. PROBLEM DEFINITION

Due to the problems which appeared recently on the agile field, our proposed model tried to find solutions for most of these problems and improve the overall agile model by integrating both Scrum and waterfall models with distributed software development.

One of the main problems on agile scrum is the human dependency, agile depends heavily on humans[6], and it considered a risk to depend on individuals heavily [7]especially on projects that have large time span because any of the project members may be absent or changed during the lifetime of the project, especially in the recent period with the COVID-19 pandemic, it has become very likely that any stakeholder will be absent, temporarily or permanently which will cause time and cost problems.

Self-assignment of tasks also have challenges, the agile scrum methodology uses the method of self-allocation to distribute tasks among team members [8], which results in team members' lack of specialization and costs additional time because at the project progresses some team members may become more familiar with a specific set of project items, so that it's better to allocate such tasks to the most familiar members rather than letting every team member select the tasks to be performed.

Project handovers are considered to be a source of delays in agile software [9], especially after the COVID-19 pandemic, where all team members are at risk of being absent at any time. Beside that due to the rapid changes on the team members because of the COVID-19 pandemic, new members usually face a lot of problems while involving the agile teams because agile less documentation, shared conceptualization and templates on the process model [2],.

In large projects, when the project serves multiple categories of users and each category awaits delivery of its system items first, it is difficult to satisfy all categories of users at early stages, because the project is treated as one unit and the delivery process is subjected only to the priority of each item. Also agile relies on customer's participation with the team during the lifetime of the project [10] while it has been difficult in many cases [2] to do so. And Failure to identify customer representatives may lead to disagreement and different views on a variety of issues.

Another problem is that most organizations have concerns about applying agile because customers may be used to the traditional methodologies and are unable to trust agile[11]. Also because lack of the management role or limited their role to make some decisions with giving the team the responsibility to be self-organized [12]. But in fact research argues that the success of group

Activities depend on the maturity level of the team. Thus, it is important to give agile teams time to form a common mental model[13]. On agile scrum while the sprint is started, no more changes will be submitted resulting in wasted time on performing tasks that may be changed later.

Agile considered one to one and face to face communications are a cornerstone of agility to the extent that the agile manifesto devotes one of its four values and two of its twelve principles to this type of communication [2,10] . Agile communications can be simple in a small environment where the teams can be co-located in an open setting, but this type of communication is difficult in the case of COVID-19 or in the far-flung software environment that exists in most medium-to-large companies. So in the recent years most of companies start applying geographically distributed agile development (GDAD) [15] to reduce the product cost. Based on previously mentioned problems this paper presents a practical process model that inspired from merging between distributed agile and traditional waterfall model. The target of this model is to provide solutions for most issues [16] that appeared recently specifically with COVID-19 as shown in Figure 3, improve the overall results during COVID-19 and satisfy new market needs.

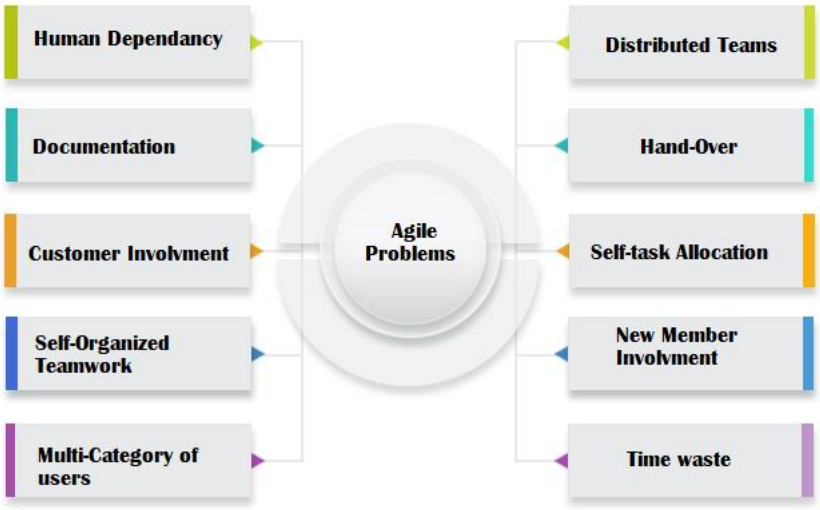

Figure 3: Agile Recent Problems

3. LITRATURE REVIEW

This section will include the recent literature on the agile hybrid models, distributed agile and distributed agile problems, the failure factors or challenges that appeared recently on agile and the COVID-19 researches on agile.

There are several attempts for merging agile with traditional methodologies. In [1], Seyam and Galal-Edeen presented Tragile framework that combines guidelines and practices from agile and traditional development methodologies, to help information systems development teams to do better work in their information systems development projects. In [17], Mateen et al. presented a survey conducted to check the result of integration of both agile and traditional v model and systematic review of literature conducted to check out the results of this integrated approach. In [18], S. Rahim, et al. introduced ScrumFall model to avoid problems of both scrum and Waterfall model. In [19], Bhavsar et al. proposed integration of Scrum and Kanban with Waterfall in the form of hybrid framework for SEM (Software Engineering Management) practices, to cover the drawbacks of Scrum and Scrumban; and reinforcement the strength of software development organization by merging required characteristics of Scrum, Kanban and Waterfall into Scrumbanfall which has a great strength compared to stand alone framework.

A lot of researches appeared to study the area of distributed agile development. In [20], Singh et al. proposed a framework for moving from traditional software development to distributed agile software development and the framework addresses time - zone based problems in distributed environment. In [21], Calefato and Ebert presented most successful tools used by companies to support collaboration in distributed agile teams. In [22], M. S. Chaves and A. Majdenbaum investigated the constructs that lead to the promotion of social interaction in Distributed Software Development (DSD) environments. In [15], M. S. Chaves and A. Majdenbaum empirically examined the relationships between agile enterprise architecture, active communication, and performance (on-time completion, on-budget completion, software functionality, and software quality). The results indicate that communication efficiency has positive effects on on-time and on-budget completion; while communication effectiveness has positive effects on functionality, quality, and on-budget completion. In [14], A. Khalid, et al. discussed the agile-scrum methodology and its limitations when used for large-scale project organization.

In order to provide a hybrid agile model, we had to study the problems that appeared in previous models. In[16], A.M.Chandrashekhar discussed the different challenges involved in various agile methodologies. The challenges are not restricted to particular role. It may be with respect to developer, customer or project managers or contractors etc. In [2] Dhir et al. presented the success and failure factors in agile development projects. A case study was conducted based on the development of five projects. Projects were developed by the students as five different teams, every team applied different approach of waterfall, spiral, and agile approach. The outcome was that in the scrum methodology results were better than waterfall and spiral model, where efficiency, product quality, and time management are higher.

Because of COVID-19, scientific research began to advance many researchers in this direction in an attempt to keep pace with the pandemic and deal with it. In[23], Asare et al. outlined some practical and theoretical recommendations of business intelligence strategies for organizations and their service supply chain network on how to be adaptive, flexible, and innovative to survive and stay competitive during these challenging times by leveraging agile dimensions, artificial intelligence systems, and data analytics to make informed decisions to enhance business operations amid COVID-19. In[4], NG ET al provide an attempt to give knowledge to get ready for any future pandemics and other emergencies from an IT management point of view. In[3], Camara et al. examined the effect of COVID-19 in a software startup setting to get it how they have responded against vulnerabilities caused by COVID-19.

According to the limitation on the previous attempts for merging agile with other methodologies and the issues that appeared in agile recently we are seeking to present a new model that copes with COVID-19 and will cover the limitation on the previous attempts and present solutions for most of issues appeared recently.

4. RESEARCH METHOD

We proposed a new method that called Distributed ScrumFall model. It is an agile integration of Scrum and Geographically Distributed Software Development (GDSD) with Waterfall model using the mixture of traditional Software Development Life Cycle (SDLC) protocols with the empiricism, agility and workflow management as shown in Figure 4.

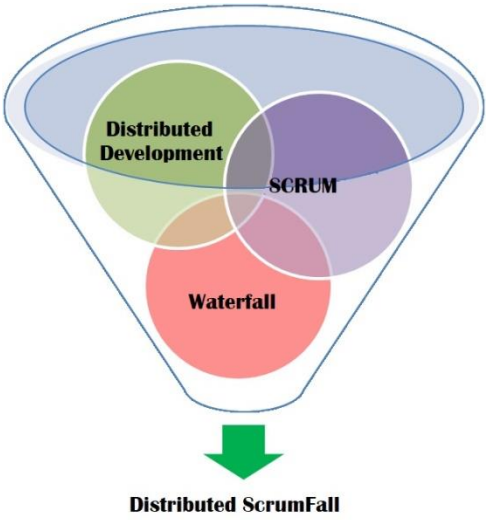

Figure 4: Distributed ScrumFall Combination.

Distributed ScrumFall Lifecycle Model

Distributed ScrumFall model consists of 3 main phases which are pregame phase, development phase and postgame phase as shown in Figure 5

4.1.1 The pregame phase

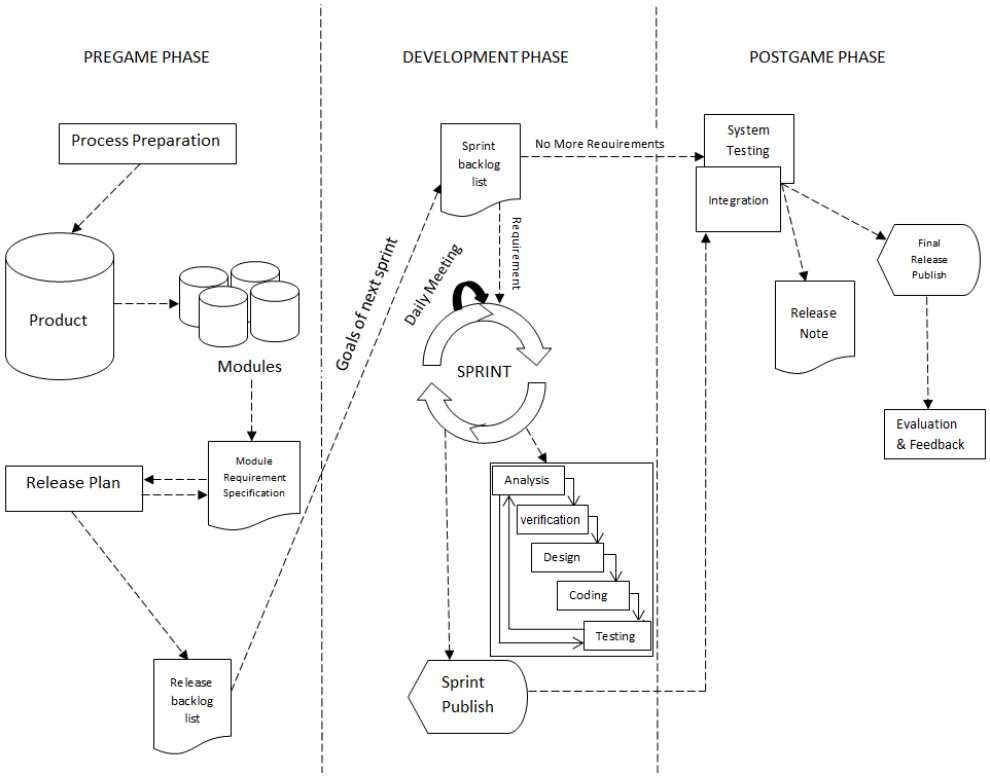

Figure 5: Distributed ScrumFall Process Workflow.

It includes three sub- phases: Process preparation, Project initiation and Release planning. Process preparation aims to establish the appropriate process for the project, after a thorough study to all aspects of the project such as the characteristics of the project (size, type, time, location, etc.), characteristics of the team (size, experience, etc.), and the nature of the client (nationality, language, technical background, etc.). Creating a process may require combining practices of different methodologies, not just applying a particular methodology or modifying it. At this stage, the types of documents that will be used in the project are chosen such as use cases, user stories, system requirement specification, user manual etc.

Project initiation phase is specifically designed for large-scale projects and aims to “split the project into smaller modules” to facilitate the process of requirements gathering from the client and to solve “multiple category of beneficiaries” problem. In large-scale projects which include multiple sub-products and each sub-product serves a different class of beneficiaries or clients, we must divide the product into modules. Because in agile we have only one product backlog[5], it will be difficult to satisfy all categories of beneficiaries early, although agile relies on providing frequent releases to the customer. Therefore, the model divides the product into modules, each of which includes its own backlog, so that we can implement more than one module simultaneously. This will help satisfy all product categories at an early stage. At this stage, the general requirements for the project will be determined, the modules' priorities will be determined, and the higher priority module requirements will be determined with approval.

Release planning phase is the first phase of the implementation of the project, where this phase aims to identify the tasks to be handed over in the next release, the distribution of tasks to the team and the work plan to implement the tasks according to their priorities. Noting that the release may divide into sprints and the release period mustn’t exceed 2 months.

4.1.2 The development phase

It includes two sub-phases: requirements processing and implementation. Development phase repeats iterations until reaching the final release. The first iteration will be for the sprint with the highest priority on the release plan. After finishing the Use Cases creation, the Use Case must be verified. "Documents Verification" or testing before coding as mentioned in requirement engineering [11]. It considered being a solution for one of the “Documentation” problems, where the Use Cases are sent to the test team to check and review before development to ensure that the requirement has been properly formulated and there are no issues or conflicts in the document. It saves the wasted time on developing incorrect tasks and it helps the test team to estimate testing effort needed and to start creating his test cases and test scenarios more early.

The design team creates the design of user interface, and then the developers implement the code of the sprint tasks. Developers create sprint test publish to be tested with the testing team. Daily meeting will be held to follow up tasks of the teamwork. These steps will be repeated until the last sprint in the release then developers create the final publish.

4.1.3 The Postgame phase

It includes two sub-phases: Testing and Evaluation. In Testing phase testing team will test every sprint publishes, makes system testing and integration testing, and then reporting issues for the developers. So that the developer must fix and solve this issues until the last sprint publish. PM will request for the final publish and create the release note.

The evaluation should be repeated after each release by evaluating the process and teamwork and obtaining customer feedback about the product and team feedback about the process to be able to identify and resolve the problems we faced in the recent release so that we avoid them in future releases. It also may be performed if we require a rapid update on the process like what happened on the COVID-19 pandemic. The benefit of this phase appeared clearly during the COVID-19 pandemic while the project team was working from home and there are changes on a lot of working strategies, so the process master updated the process rapidly with the new strategies of work. All sub phases of the Distributed ScrumFall model is explained in Figure 6

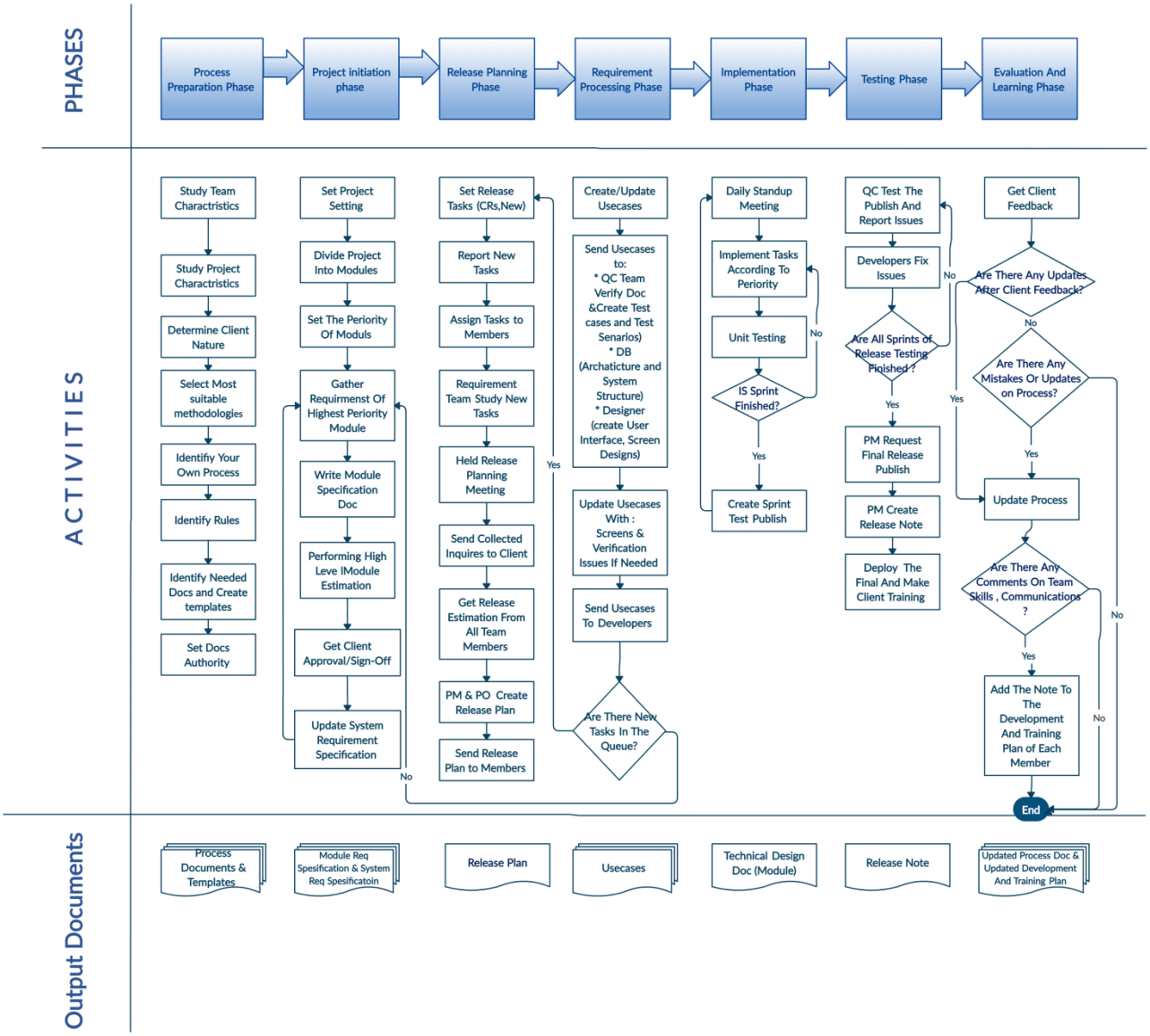

Figure 6: Block Diagram of Distributed ScrumFall phases and activities

4.2 Distributed ScrumFall Roles and Responsibilities

As the Distributed ScrumFall model is a combination of both Scrum and Waterfall models, so the proposed model inherits its roles from Scrum and Waterfall models. Roles and responsibilities will be selected according to the project need at the initiation phase of any project, and it will be set by the process master and the project manager. Distributed ScrumFall model present a new role which called "Process Master". The model also emphasized the importance of the "Project Manager" role.

Process Master must have experience on all development methodologies. S/he is responsible for:

- Selecting the most suitable practices and creates the process which matches the characteristics of the team and the project.
- S/he is dependable for guaranteeing that extend is carried through agreeing to the hones, values and rules of the method.
- S/he solves any problem related to the process and updates the process during the project timeline if needed.
- S/he must provide the support for all team members.
- S/he is responsible for developing and improving the process continuously. We can describe his role as "Process facilitator".

Project Manager is responsible for:

- Makes the choices. In arrange to be able to do this, he communicates with the teamwork to decide the current circumstance, and to recognize any troubles or insufficiencies within the process.
- S/he also responsible for assigning the product items on the team and assign tasks to members according to the item distribution.

- S/he is responsible for managing the time, defining the scope of each release or sprint, setting the priority and creating the release note. We can describe his role as "Business management facilitator". This role will help in solving the problem of "Self Organized Team".

4.3 Distributed ScrumFall Practices

These practices considered being the best practices while performing the proposed model. Some of this practice inspired from agile and traditional methodologies and the others are self-inspired as solutions to the problems that faced us while the project lifetime.

4.3.1 Practice 1 "Using documents and shared guidelines"

Because of agile face to face communication the concepts are built and stored in each individual's memory through communication and collaboration during requirement activities which considered one of agile drawbacks [11]. Using documents and guidelines mean that if there is any part of the process that more than one member can deal with it must be subject to structure to make it easy to deal with. Because if there is many of team members deal with the same part of the process without a predefined guidelines or document template about how to using it then everyone will make it from his perspective and it may cause a time waste.

This practice will help the distributed teamwork in communicating with each other and perform the process correctly because every part on the process is predefined and dedicated. Process with predefined guidelines helps new members to understand and involve to the work process easily. For example Mails templates, code guidelines, different documents templates.

4.3.2 Practice 2 "Welcome changing requests anytime even during the sprint"

On Scrum once the scope of a sprint is committed, no change requests can be added[10]. This sometimes leads to a loss of time and effort working in a wrong way within the sprint. However receiving change requests during the range can correct the direction of work and save a lot of time and effort. It also helps on customer satisfaction and save the costs that may be wasted if we continue working on incorrect requirements.

4.3.3 Practice 3 "Setting the system items and distributing the items ownership"

System items can be identified by studying the business requirements well and setting every set of related features as an item. Items must be assigned on all team members according to business requirements which mean each set of related items are better to be owned by the same member. The items ownership make every member feels responsible with his own items and thus does his best to provide them with the highest quality. In the case of errors in a specific item it will be easy to identify the responsible person and take appropriate action and thus maintain the efficiency of the system.

This practice will help also on avoiding problems and time wasted in the case of any member handover which is very possible matter. Through providing an accurate and professional handover plan by shifting the items assigned to the absence member to the new member and that's all because every member has a specific set of items. This is better than agile where the member could involve in most system items and therefore the absence of any member causes a decline in the process progress.

4.3.4 Practice 4 "Assigning tasks must be according to business requirements"

Task management must be thoughtful process that depends on the project scale .in large-scale projects assigning tasks on each team member must be according to the business requirement. After distributing the items ownership, the tasks will be assigned to each member according to his items ownership.

This practice helped the team members to understand the business of their items very well, so that the tasks completed with high quality on less time.

4.3.5 Practice 5 "Having a backup member or second owner for each item on the project"

On the department level each item on the project items distribution must be owned to two members, one member as an item owner and the other as an item backup. This means that each item must have requirement owner, requirement backup, developer owner, developer backup, tester owner, tester backup.

This practice will save time lost due to the temporarily absence of the item's owner member as the item's backup member will perform his duties. It also saves time that can be wasted with the permanently absence of the item's owner member, because in this case the backup member of the item will be the owner of the item, after which a new member will be appointed as a backup member of the item, so that we do not waste time while the new member understanding the business for the item and participate in the workflow and keep the deadline as well. Also, having two members working on the same item improves its efficiency and reduces any errors.

4.3.6 Practice 6 "Create your own Use Case template"

We always must choose between using either Use Case or User Story both of them have its valuable using. In fact, M. A. Awad [12] made a comparison between agile and traditional and he found that on traditional methodologies managing requirements using Use Cases and scenarios were very effective in capturing functional requirements and helping to monitor expected behaviors of the system. Creating your own Use Case template helps to perfectly illustrate tasks in a way that has a positive impact on results.

4.3.7 Practice 7 "Deliver working software frequently, from a couple of weeks to a couple of months"

Prototyping or delivering frequent releases is one of the agile practices [11]. Release is iterative cycle where the functionality is developed or enhanced to produce new increments. Each release includes one or more sprint each sprint includes the traditional phases of software development: requirements, analysis, design, evolution and delivery phases. One sprint is planned to last from one week to one month. Client must have valuable parts of the system. Each increment must have valuable features that could be presented to the client.

So we can take customer feedback early and frequently and this can help on improving the quality of the product. Frequent releases will help on satisfying the customer. Fast delivery for the product releases is better than being late to deliver the final product as Poppendieck said [10] "We need to figure out a way to deliver software so fast that our customers don't have time to change their minds."

4.3.8 Practice 8 "Using project management tools and online communication methods"

Due to the COVID-19 pandemic and online working, we need online communication methods to facilitate the communication among the team members. We suggest using Zoom mobile application or Google Meet.

On the software projects we need tools to manage tasks among the team members. We suggest using Team foundation Server (TFS) as "Microsoft Azure", Project management tool as "JIRA" or bug trackers as "Mantis". Also using instance messages and mails instead of face to face communication.

4.3.9 Practice 9 "Customer participate only if needed"

Agile relies on customer participation with the team [10] while it has been difficult in many cases [2] to do so. On scrum the sprint planning meeting attended by customers [5] noting that some product items may requires more than on sprint and it will be very hard to ask stakeholders to attend every sprint planning meeting, Also customer participation can have a drawback which is the frequent meetings will lead to informal communication between the customer and the team[11]. And the failure to identify customer representatives may lead to disagreement and different views on a variety of issues [11], so it's better for customer to participate only if needed and also to predefine the ways of communications with the customer as mentioned previously on practice 1 , there are a pre-defined templates for each communication way such as inquire mails , minutes of meetings mails and approval mails.

4.3.10 Practice 10 "Training for both new members and end users"

Training is a very important part of our model for either new members or end users. New members must obtain business training to understand project requirements before obtaining technical training. Training the business requirements of new members sometimes leads to new ideas that help promote the project because every new member has its new ideas, and training users will also help them to use the system in the right way.

4.3.11 Practice 11 "Providing the support after deploying the system"

Providing support to the customer after delivering the system by appointing a well-trained technical employee on the system, this employee will be called "Customer Support". This technical support will assist end users in using the system, and in most cases will be responsible for training end users. He will be responsible for communicating any issues or problems on the system to the development team. Supporting your product after deployment will boost customer confidence.

4.3.12 Practice 12 "Process Enhancement"

The team must be aware of the importance of the process enhancement because the team is the basis for the process enhancement. If the team is skilled in identifying problems and providing suggested solutions, we have skipped the first steps in enhancing the process. Now comes the role of the process master in the process enhancement by helping the team to choose the best solution to problems through the provision of previous practices appropriate to the situation or provide solutions proposed previously tried in similar situations. This makes the teamwork feel comfortable and easy, reduces wasted time in problems and Increases productivity.

5. DISCUSSION

To judge this model and to know if its application affects the workflow in a positive way, we had to define a case of study to test the model on it.

5.1 Case Study

The case organization is an IT company that has a globally distributed organization. The company's main activities are in Saudi Arabia with three offices, and it has another office in Egypt. The company is experienced in carrying out government projects, but using agile methods with them is a new experience. The product which selected for studying is a large product development system with a history spanning more than nine years called "Electronic Treatments System". ET system is adaptable as it serves multiple beneficiary users all over Saudi Arabia. ET system is scalable as it started working from 9 years ago and bears with growth until today. In previous releases of ET project it was released almost twice a year, but now it releases from one month to two months maximum. The product organization has grown over the years from 18 persons to a current size of 190 employees. The product organization consists of two major groups: the service organization, which is distributed all over Saudi Arabia, and the product development organization, which is centralized in Egypt. The service organization sets up new releases for customers, e.g. by configuring the system, supporting the system and in some cases making minor modifications. The product development organization has approximately 40 persons in Mansoura Egypt. This study focuses purely on a software

development organization. The organization has been applied the traditional waterfall model for many years, and after that it was applied Distributed ScrumFall model for 3 years including the period of COVID-19 pandemic.

5.2 Teamwork satisfaction results

We have relied on measuring the teamwork satisfaction on a questionnaire of 37 questions; the questionnaire main components are Transparency between employee and organization, job satisfaction, relationship with management, work environment, motivation, and communication as shown in Table 1. Two questionnaires have been conducted, both of them filled by the teamwork or employees. A random sample of 14 employees was selected to fill the questionnaires. The sample selected on the employees who have been on their jobs at both questionnaires time and they have made both questionnaires. The first questionnaire was to measure the teamwork satisfaction before applying Distributed ScrumFall model and the other was to measure the teamwork satisfaction after applying Distributed ScrumFall model. As shown in Figure 7, the results of both questionnaires were 64% of employees were satisfied more after applying the DSF model, 28.5% of employees have less satisfaction after applying the DSF model and 7% of employees have the same satisfaction.

Table 1: Sample of the teamwork satisfaction questionnaire

Sample of the teamwork satisfaction questionnaire questions					
1) Team communication is good	1	2	3	4	5
2) Work process is good	1	2	3	4	5
3) No contradiction between written instruction and practices	1	2	3	4	5
4) Good working environment for you	1	2	3	4	5
5) Well informed about targets, working progress and future plans	1	2	3	4	5
6) Managers make decisions clearly	1	2	3	4	5
7) You receive useful feedback about your performance	1	2	3	4	5
8) Your direct manager motivate you	1	2	3	4	5

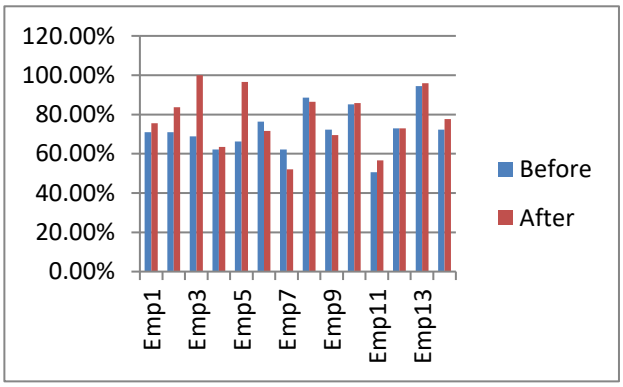

Figure 7: Team work satisfaction before and after applying DSF model

5.3 Productivity

To measure how the model affected the productivity factor we have to compare the productivity on three different releases versions, so I select 3 random releases versions of the project; the first release version took about 6 months to delivery and didn't apply our proposed model, the second release version took about 3 months to delivery and applied the proposed model before COVID-19 and the third release version took 1 month to delivery and applied our proposed model during the COVID-19 pandemic. The tasks on each version have a weight of Large, Medium and small where Small(S) =x, Medium (M) =2x and Large (L) =3x, where x is the point.

I relied to the simple productivity output formula as the following

$$Labor\ productivity = \frac{Total\ output}{Total\ input} \quad (1)$$

On the case of our project we will measure the productivity of each version through the number of delivered tasks per time as the following

$$productivity = \frac{producted\ tasks}{production\ time} \quad (2)$$

1) THE PRODUCTIVITY FOR EACH TYPE OF TASKS SEPARATELY

As there are three types of tasks, so we will calculate the productivity for each type of task within the different three version of the product as shown in Figure 8 to display how many large tasks, medium tasks and small tasks are performed before applying Distributed ScrumFall model, after applying Distributed ScrumFall model and during the COVID-19 pandemic.

Hence: We will refer to the productivity of large tasks per day with α, and the productivity of medium tasks per day with β, and productivity of small tasks per day with Ω.

Release version before Distributed ScrumFall (DSF) includes 46 tasks (13 large, 20 medium, and 13 small)

Total time per days = 110 days

$\alpha = \frac{producted\ large\ tasks}{total\ production\ time} = \frac{13}{110}$ = 0.11 task per day

$\beta = \frac{produced\ medium\ tasks}{total\ production\ time} = \frac{20}{110}$ = 0.18 task per day

$\Omega = \frac{producted\ small\ tasks}{total\ production\ time} = \frac{13}{110}$ = 0.11 CR task per day

Release version after Distributed ScrumFall (DSF) Includes 52 tasks (12 large, 0 medium, and 40 small)

Total time per days = 80 days

$\alpha = \frac{producted\ large\ tasks}{total\ production\ time} = \frac{12}{80}$ = 0.15 task per day

$\beta = \frac{producted\ medium\ tasks}{total\ production\ time} = \frac{0}{80}$ = 0 task per day

$\Omega = \frac{producted\ small\ tasks}{total\ production\ time} = \frac{40}{80}$ = 0.5 CR task day

Release version after Distributed Scrumfall (DSF) during COVID-19 Includes 39 tasks (3 large tasks, 9 medium tasks, and 27 small tasks)

Total time per days = 20 days

$\alpha = \frac{\text{producted large tasks}}{\text{total production time}} = \frac{3}{20}$ = 0.15 task per day

$\beta = \frac{producted\ medium\ tasks}{total\ production\ time} = \frac{9}{20}$ = 0.45 task per day

$\Omega = \frac{producted\ small\ tasks}{total\ production\ time} = \frac{27}{20}$ = 1.35 task per day

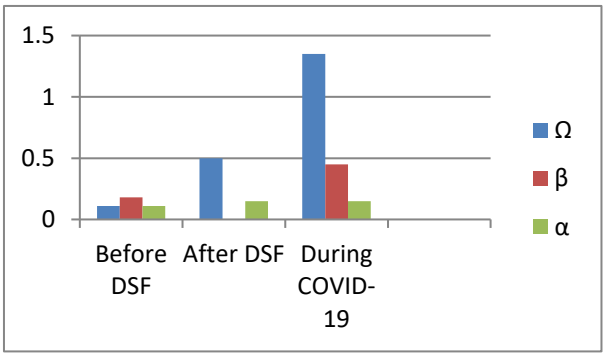

Figure 8: The productivity of small tasks, medium tasks and large tasks per day

2) THE PRODUCTIVITY WITH POINTS (X)

Since each type of tasks has a number of points (x) so, we can calculate the productivity for each version by another way through getting the productivity for points per day as shown in Figure 9.

Hence: we will refer to the total productivity of the release version before Distributed ScrumFall with ω1, total productivity of the release version after Distributed ScrumFall with ω2, total productivity of the release version after Distributed ScrumFall and during COVID-19 with ω3.

Release version before Distributed ScrumFall includes 46 tasks (13 large, 20 medium, and 13 small)

Total time per day= 110 days

Total number of produced tasks per points (x) = number of large tasks *3+ number of medium tasks*2+number of small tasks *1=13*3+20*2+13=92 points

$\omega 1 = \frac{producted\ tasks\ per\ point}{total\ production\ time} = \frac{92}{110}$ = 0.83 point per day

Release version after Distributed ScrumFall Includes 52 tasks (12 large, 0 medium, and 40 small)

Total time per day= 80 days

Total number of produced tasks per points (x) = number of large tasks *3+ number of medium tasks*2+number of small tasks *1=12*3+0+40=76 points

ω2= $\frac{producted\ tasks\ per\ point}{total\ production\ time}$=$\frac{76}{80}$= 0.95 point per day

Release version after Distributed Scrumfall during COVID-19 Includes 39 tasks (3 large tasks, 9 medium tasks, and 27 small tasks)
Total time per day= 20 days
Total number of produced tasks per points (x) = number of large tasks *3+ number of medium tasks*2+number of small tasks *1 =3*3+9*2+27=54 points
ω3= $\frac{producted\ tasks\ per\ point}{total\ production\ time}$=$\frac{54}{20}$= 2.7 points per day.

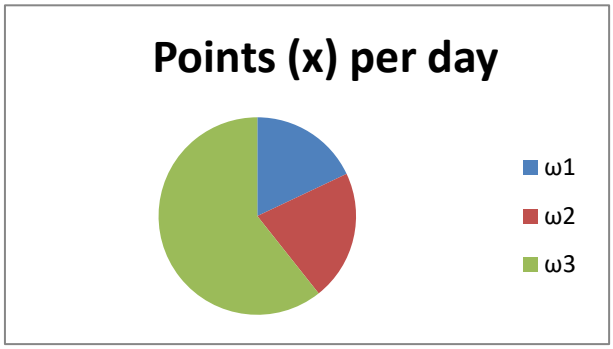

Figure 9: The productivity of points (x) per day

We can summarize all of the above that applying and developing the proposed model has a positive impact on productivity.

5.4 Quality

In calculating the effect of the proposed model on the quality of the final product, it was based on three main factors

- Delay in delivery dates.
- Number of incidents (Incident defined as an error which appeared in the product after delivery to the client)
- Number of clients accepted the release version

We measure the quality of the final product by selecting a sample of three random versions. The three random versions were the same that used previously in the productivity measurement. The first release version was delivered to the customer before applying the DSF model, the second one was delivered after applying DSF model and the latest was delivered during the COVID-19 pandemic. The planned date, delivery date, no of clients and no of incident of the three versions are shown in Table 2.

Table 2: The three random versions data

	Planned Date	Delivery Date	No of Clients	Incident
Before DSF Model	05/06/2016	19/07/2016	8	6
After DSF Model	20/10/2019	18/11/2019	9	4
During Covid-19	26/03/2020	30/03/2020	11	1

1) DELAY IN DELIVERY DATES:
The delay on the delivery dates of the three versions was calculated according to the next equation.
Delay days = delivery date – planned date.
According to table 2 and by applying the previous equation:
Delay days on the first release version before applying DSF model = 19/07/2016-05/06/2016 = 44 days
Delay days on the second release version after applying DSF model =18/11/2019-20/10/2019= 29 days
Delay days on the third release version during COVID-19=30/03/2020- 26/03/2020= 4 days
As shown in Figure 10 we conclude that the implementation of Distributed ScrumFall model makes delivery times more accurate and reduces the delay in delivery times especially during COVID-19.

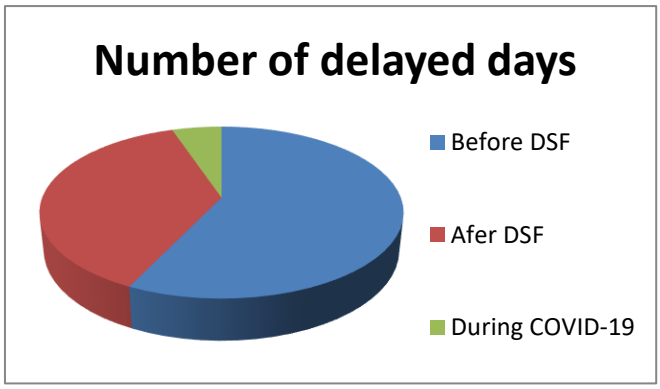

Figure 10: Number of delayed days on each version

2) NUMBER OF INCIDENTS:

Incident in software projects means number of issues which appear in the final product on the client environment after deployment. Here we will compare the previous three versions of the final product by number of incidents which appeared on each version of them in the client environment.

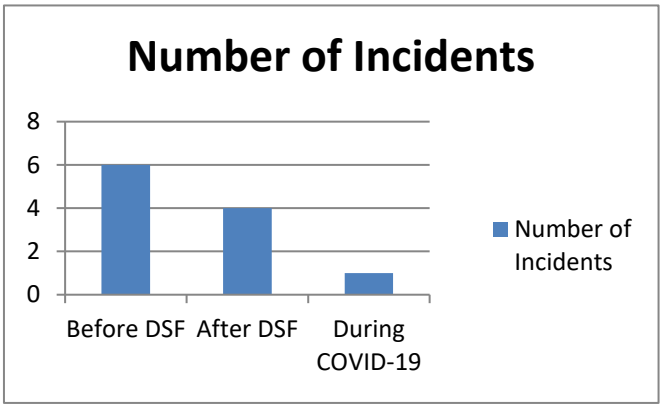

Figure 11: Number of incidents which appeared on the product versions

As shown in Figure 11 only one incident appears in the project version that applies Distributed ScrumFall model and this confirms that the model affects well the low rate of errors in the final product, which raises the level of efficiency on the final product and increases customer satisfaction.

3) NUMBER OF CLIENTS ACCEPTED THE VERSION:

Since the project serves more than one client, therefore, the number of clients who accepted and were appropriate for the product release version is an important factor in measuring the product quality, so we will compare the previous three versions by numbers of clients that accepted the versions.

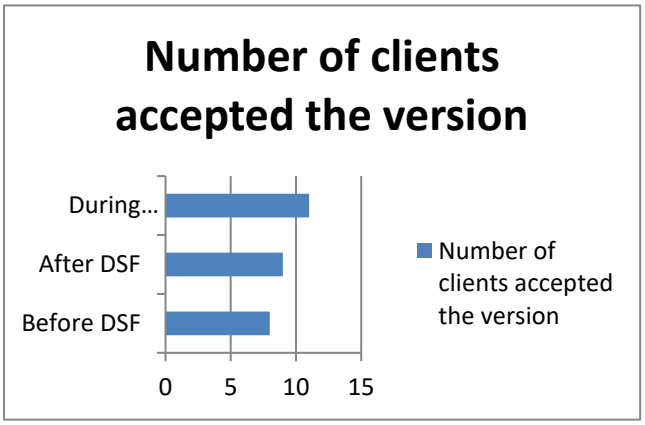

Figure 12: Number of Clients Accepted the Release Version

Figure 12 shows that the release version that applies Distributed ScrumFall model has been more accepted than the other two versions, which means that the application of the proposed model helped in increasing the number of clients who accepted the release.

5.5 Distributed ScrumFall model VS. Agile model

As shown in Table 3, which explain advantages of hybrid DSF model over agile with respect to specific parameters. This comparison shows that DSF model overcome most of agile problems which explained before in problem statement definition and explains how the DSF model overcomes those problems.

Table 3: DSF Model Vs. Agile

Parameter	Agile	DSF Model	Discussion
Individual dependency	High	Low	One of agile manifesto is depending on individuals heavily, but in DSF depends on documentation and backup members.
Documentation	Low	High	Agile gives importance to work over documents, but DSF model have documents.
Customer involvement	High	Low	In agile customer participate with the team in all the project phases, but in DSF customer participate only if needed.
Multi- category of users	low	High	Agile can't satisfy all categories of users at early stages; on other hand DSF can do that because it divides the product into sub-products.
Distributed teams	Low	High	Agile can't deal with distributed teams, but DSF can deal with distributed teams because of using instance messages and mails.
Handover	Low	High	In Agile handover cause time waste, but in DSF handover save time because it uses backup members.
Self-task allocation	High	Low	Agile Use self-task allocation to assign tasks, but DSF assign tasks according to item distribution and business requirements.
New member involvement	Low	High	New member involvement is a problem in agile, but in DSF provides guidelines and templates that helps new member during the training.

6. CONCLUSION

COVID-19 is already leading to a massive transformation in the business world. Therefore, we have a need for more flexible software development models. We concluded that it's not logically to have a static model that can be performed on specific work conditions, so software organizations must have the skills to customize and improve any model to suit with all circumstances and also to build our own model if it required. This paper presents Distributed ScrumFall (DSF) model that integrates Distributed Software Development (DSD) with Agile Scrum and Waterfall Model to cover the limitation on the previous related attempts and present solutions for most recent issues. A case study was conducted to check the results of Distributed ScrumFall model and it proved that the results after applying Distributed ScrumFall well improved most project scaling factors especially during COVID-19. For further research to enhance this research applying the model on large scale organizations which have more than 4 large offices, as in this work the organization has only four offices on different countries and the size of each office was medium. If the number of offices and the company size is more it will help in understandings results more effectively and clearly.

7. REFERENCES

[1] M. S. Seyam and G. H. Galal-edeen, "Traditional versus Agile : The Tragile Framework for Information Systems Development," *Int. J. Softw. Eng.*, vol. 4, no. 1, pp. 63–93, 2011.

[2] S. Dhir, D. Kumar, and V. B. Singh, *Success and failure factors that impact on project implementation using agile software development methodology*, vol. 731. Springer Singapore, 2019.

[3] R. da Camara, M. Marinho, S. Sampaio, and S. Cadete, "How do Agile Software Startups deal with uncertainties by Covid-19 pandemic?," *Int. J. Softw. Eng. Appl.*, vol. 11, no. 4, pp. 15–34, 2020, doi: 10.5121/ijsea.2020.11402.

[4] J. J. NG, N. Sathesh, and J. Lee, "Considerations for IT Management in a COVID-19 World," *IEEE Eng. Manag. Rev.*, vol. 8581, no. c, pp. 1–5, 2020, doi: 10.1109/EMR.2020.3014777.

[5] P. Abrahamson, Outi Salo, Jussi Ronkainen, and Juhani Warsta, "Agile software development methods: Review and analysis," *VTT Publ.*, p. 112, 2002, [Online]. Available: http://www.vtt.fi/inf/pdf/publications/2002/P478.pdf.

[6] A. Chagas, M. Santos, C. Santana, and A. Vasconcelos, "The Impact of Human Factors on Agile Projects," *Proc. - 2015 Agil. Conf. Agil. 2015*, pp. 87–91, 2015, doi: 10.1109/Agile.2015.11.

[7] K. Beck *et al.*, "Agile Manifesto," *Softw. Dev.*, vol. 9, pp. 28–35, 2001, [Online]. Available: http://agilemanifesto.org/.

[8] K. Kuusinen, P. Gregory, H. Sharp, L. Barroca, K. Taylor, and L. Wood, "Xp2017," vol. 283, no. May, pp. 135–150, 2017, doi: 10.1007/978-3-319-57633-6.

[9] C. J. Stettina and E. Kroon, "Is there an agile handover? An empirical study of documentation and project handover practices across agile software teams," *2013 Int. Conf. Eng. Technol. Innov. ICE 2013 IEEE Int. Technol. Manag. Conf. ITMC 2013*, 2015, doi: 10.1109/ITMC.2013.7352703.

[10] A. S. Requirements, S. D. Manager, D. Tire, and A. Consultant, *[Dean_Leffingwell]_Agile_Software_Requirements_Le(BookFi.org).pdf*. 2011.

[11] I. Inayat, S. S. Salim, S. Marczak, M. Daneva, and S. Shamshirband, "A systematic literature review on agile requirements engineering practices and challenges," *Comput. Human Behav.*, vol. 51, pp. 915–929, 2015, doi: 10.1016/j.chb.2014.10.046.

[12] M. Gechman and M. Gechman, "Software Development Methodologies," *Proj. Manag. Large Software-Intensive Syst.*, pp. 49–66, 2019, doi: 10.1201/9780429027932-4.

[13] E. Zavyalova, D. Sokolov, and A. Lisovskaya, "Agile vs traditional project management approaches: Comparing human resource management architectures," *Int. J. Organ. Anal.*, vol. 28, no. 5, pp. 1095–1112, 2020, doi: 10.1108/IJOA-08-2019-1857.

[14] A. Khalid, S. A. Butt, T. Jamal, and S. Gochhait, "Agile Scrum Issues at Large-Scale Distributed Projects: Scrum Project Development at Large," *Int. J. Softw. Innov.*, vol. 8, no. 2, pp. 85–94, 2020, doi: 10.4018/IJSI.2020040106.

[15] Y. I. Alzoubi and A. Q. Gill, "An Empirical Investigation of Geographically Distributed Agile Development: The Agile Enterprise Architecture is a Communication Enabler," *IEEE Access*, vol. 8, pp. 80269–80289, 2020, doi: 10.1109/ACCESS.2020.2990389.

[16] A.M.Chandrashekhar, "Challenges in Agile Software Development," pp. 227–232, 2017.

[17] A. Mateen, M. Tabassum, and A. Rehan, "Combining Agile with Traditional V Model for Enhancement of Maturity in Software Development," vol. 7, no. 2, pp. 280–296, 2017, [Online]. Available: http://arxiv.org/abs/1702.00126.

[18] S. Rahim, A. E. Chowdhury, D. Nandi, and M. Rahman, "ScrumFall: A Hybrid Software Process Model," *Int. J. Inf. Technol. Comput. Sci.*, vol. 10, no. 12, pp. 41–48, 2018, doi: 10.5815/ijitcs.2018.12.06.

[19] K. Bhavsar, V. Shah, and S. Gopalan, "Scrumbanfall: An Agile Integration of Scrum and Kanban with Waterfall in Software Engineering," *Int. J. Innov. Technol. Explor. Eng.*, vol. 9, no. 4, pp. 2075–2084, 2020, doi: 10.35940/ijitee.d1437.029420.

[20] M. Singh, N. Chauhan, and R. Popli, "A Framework for Transitioning of Traditional Software Development Method to Distributed Agile Software Development," *IEEE Int. Conf. Issues Challenges Intell. Comput. Tech. ICICT 2019*, 2019, doi: 10.1109/ICICT46931.2019.8977654.

[21] F. Calefato and C. Ebert, "Agile collaboration for distributed teams," *IEEE Softw.*, vol. 36, no. 1, pp. 72–78, 2019, doi: 10.1109/MS.2018.2874668.

[22] M. S. Chaves and A. Majdenbaum, "Distributed software development in agile projects: a model for the promotion of social interactions," *Rev. Gestão e Proj.*, vol. 11, no. 1, pp. 17–35, 2020, doi: 10.5585/gep.v11i1.16165.

[23] A. O. Asare, P. C. Addo, E. O. Sarpong, and D. Kotei, "COVID-19 : Optimizing Business Performance through Agile Business Intelligence and Data Analytics," pp. 2071–2080, 2020, doi: 10.4236/ojbm.2020.85126.

[24] "problems of waterfall model." https://warren2lynch.medium.com/what-is-the-problems-of-waterfall-model-38de858f1058 (accessed Jan. 28, 2021).

[25] "What is SCRUM?" https://www.educba.com/what-is-scrum/ (accessed Jan. 28, 2021).