International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878 (Online), Volume-9 Issue-4 November 2020

Addressing Security Requirements through Onshore Agile Development When Packaging Software in the Distributed Development Domain: An empirical study

Abdulrahman M. Qahtani

Abstract ***Designing software for use by multiple clients has become commonplace in the software sector and has led to many vendors focusing on developing software for a specific sector, marketing the product then modifying it to a customer's requirements. To fit the software to the client's needs involves a unique form of teamwork, and it is usually an offshore team that processes the requests and implements the changes to the initial infrastructure. Unfortunately, this contravenes organizations' information security requirements, due to their multiple structures and infrastructures and their need for privacy as well as swift processing of requests at reasonable cost. This study proposes a hybrid model, the Onshore Agile Security Requirements Development (OASRD) model, which uses Agile to meet the security implications arising from the onshore team working at the client's site while it processes the customization requirements. It investigates the impact of the model on productivity, measured by the number of security and customization requirements that are processed and the estimated cost in terms of human resources. The evaluation reveals a statistically significant increase in productivity of about 40%, accompanied by a reduction in cost of more than 48% over the entire customization process, demonstrating the advantages of customizing packaged software through distributed development.***

Keywords: security requirements, distributed development, packaging software, Agile development.

I. INTRODUCTION

The software industry has undergone significant change since the 1960s, especially in its approach to and methods of software production; and in recent decades there has been a marked turning point in the form of the Agile approach, launched by the Agile Manifesto [1]. Since then, research has demonstrated the impact of Agility in every aspect of the software development life cycle, from eliciting requirements to delivering solutions and evaluating outputs[2]. Another feature is that, when multiple teams work together on software development projects across organizational boundaries and on several sites, there are associated challenges to software engineering practice, such as in the communication and coordination between teams and organizational and cultural differences [3]. One software development process that is affected by such change is requirements engineering (RE), which plays a vital role in both collocated and distributed domains [4]. Recently, the information systems of various organizations have paid a great deal of attention to cybersecurity[5]. This is reflected in the software sector, which has increased its interest in addressing security and the time and costs involved in implementing its requirements and ensuring quality. As indicated in the literature, regardless of business model or domain one aspect of the sector that has had to confront this issue is the customization of packaged software. This is because, in a distributed domain, RE is a complex intersectional phenomenon that encompasses numerous technical, social and organizational aspects [6],[7].

This study aims to investigate the impact of using Agile principles, such as close communication, in a multiple-client domain where an offshore team undertakes customization at the vendor's site, in order to implement the security requirements at the client's site. To achieve this goal, a hybrid model is proposed: the Onshore Agile Security Requirements Development Model (OASRD). This shows both how the process works and the flow of requirements and data between the multiple clients and the distributed software vendor. It potentially facilitates customization in a distributed domain to achieve security and process urgent requirements quickly and at low cost. The model is evaluated using empirical data collected from a case study of five distributed clients of a single software product that is customized to their requirements by the vendor. Data from more than 2,700 requirements over 900 working hours are used to establish the model's productivity and cost. The rest of this article is structured as follows: section 2 gives the background and reviews the research conducted on distributed development, Agile development, and the customization and security requirements in this domain. Section 3 discusses the proposed model and its components in detail. Section 4 presents the empirical study, its procedure and setting, while section 5 presents and discusses the results of the evaluation. Finally, section 6 concludes this study by looking at the findings and how they meet the study's goals.

Revised Manuscript Received on October 20, 2020.
***** Correspondence Author
Abdulrahman M. Qahtani*, Computer Science, College of Computers and Information Technology, Taif University, Taif, Saudi Arabia. Email: amqahtani@tu.edu.sa

© The Authors. Published by Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP). This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/)

II. RELATED WORK

In recent decades, research on software development has examined both the collocated and the distributed domains, and some systematic reviews of distributed software development [3], [8] have pointed to a lack of empirical studies on cases in industry to reflect the reality of research hypotheses.

Thus, this study focuses on a real-life case study and evaluates the proposed model through actual data. This section gives an overview of the literature on distributed software development, the customization process, Agile development principles and security requirements in this domain.

A. Distributed Software Development

Until the emergence of distributed software development in the 1980s [9], the software sector relied on traditional, in-house software development [10]. In the two decades since the 1990s, the sector has shown much interest in this type of development [11]. Software development may be defined as the activities, methods, practices and technologies that companies and individuals employ [12]. It is undertaken by work organization teams in software projects, and may take a variety of forms. In the collocated form, all those involved in the development are on a single site [12], while in the distributed form the actors (e.g. project team, customer and users) are physically remote [13]. Currently, the software sector decentralizes and distributes both development and maintenance in a movement known as global software development [9]. This has attracted a great deal of interest, directed at extending distributed development and outsourcing projects [14]. The motivation is to allocate development teams to particular locations in order to secure higher quality and lower cost products and services, skilled human resources and an appropriate infrastructure[15]. According to Sengupta et al. [9], global development was established in the late 1980s following identification of the most effective practices for each aspect of software development; however, the switch to a distributed project workforce poses challenges as well as obvious advantages, and these have been the focus of recent research[9]. Distributed software development projects take various forms, depending on the locality and the interactions among those involved in the project [16]. While some use multiple teams in a single locality, others use teams spread around the world [17]. The next section describes the models of practice in distributed software development and the associated issues, referring to studies that investigate these challenges. Another aspect of distributed software development that is investigated in the literature is multiple-site development and organizational boundaries. RE and project management become increasingly difficult when a project is conducted across several sites, and this is exacerbated by organizational and cultural boundaries. Much research on distributed software development has discussed the impact on the process of development. The study by Damian and Zowghi [18] identifies the inherent issues of communication, knowledge management, increased cultural diversity and multiple time zones. Similarly, Akbar et al., [19] considers the challenges associated with global software engineering, concluding that managing the requirements of distributed stakeholders in software development involves issues that relate to the cultural, time and organizational differences.

B. Customization of Packaged Software

The software sector offers many kinds of products, some built from scratch to clients' requirements and some sold 'off-the-shelf' and then customized. Therefore, the RE process is differentiated by software type and the extent of the required changes. Most studies on the processes and practices of distributed software development in terms of management of team roles and accountability, including requirement management during the software development life cycle, focus on the teams responsible for client requirements, from the selection to the delivery of the solution through decision-making and development. Xu and Brinkkemper in [20] describe the differences between the following software products, as in Figure 1. Packaged software (PS) defined by Xu and Brinkkemper [20] as ready-made products obtainable from software vendors. They are built to general requirements then sold to multiple clients, and may be ordered with the source code then customized to meet requirements. Examples are Enterprise Resource Planning (ERP) and Customer Relationship Management (CRM) systems. Over the past decade, many organizations have decided to implement PS, thus this software market is growing fast [22].

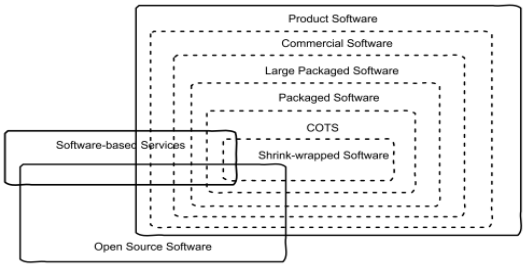


Fig. 1. Categories of software products [20].

The recent research direction has been for the software development market to specialize in products that are custom-ready, not built from scratch, in order to save time by shorting development time and to deliver high-quality, trusted software [23]. Moreover, custom-ready software such as software packages has a tremendous influence on an organization's configuration, so vendors have concentrated on building packaged software that is customized to clients' requirements, leaving the client organizations to focus on managing the environment for that software, including people and hardware, to facilitate the implementation by accurately defining its requirements [24]. Sawyer [25] considers the changes in RE processes to cater for clients' customization. The following sections discuss these challenges, such as the requirements of and opportunities in distributed software development projects. The rapid growth, huge importance and seeming predictability of PS development make it obligatory to understand both RE practice in PS implementation and the development process itself [24].

Jebreen and Wellington [24] observe that there have been few studies on RE practices for PS products, and future studies focus on RE's other aspects, due to the lack of tools to analyse, manage and collaborate with clients over the customization of their PS products in the distributed domain [24].

C. Agile Methods and Locality in Software Development

Agile software development is referred to by the software development community as a phenomenon [26], rather than an approach or methodology: it is an entire philosophy of software development and a new way of thinking about processes and project management[27]. The business community [28] has a pressing need for a development method that is swifter than the conventional plan-based approach. Agile arose in reaction to approaches such as the waterfall model, to reduce time and costs and to accommodate changes to the requirements at any stage without compromising the entire development. According to Shore and Warden [29], "Agile methods are processes that support the Agile philosophy".

Distributed software development is now the norm, and the trend is attracting attention in Agile software development with a view to adopting its methodologies [30], [31]. Distributed Agile development refers to the use of Agile principles to secure its advantages for such projects. As the practice promotes an iterative process, it helps to address the inevitable differences in culture and communication [32]. Organizations have embraced Agile methods in distributed development environments in various forms [33]. Anwar et al., [34] undertook a systematic review of the use of Agile in global software development, concluding that it has not yet been thoroughly examined. Despite the increasing number of articles, these are mainly industrial reports and a few evaluations. Therefore, in-depth research is required to investigate the advantages and disadvantages of applying Agile concepts to distributed software development projects [35].

D. Security requirements in distributed development projects

Security is an essential aspect of any information system, and is an area of concern from the initial stages of system design and development. Various aspects of software security have involved researchers, developers and engineers over the past two decades in developing approaches and methods for security requirements [4]. Many articles and systematic reviews focus on security research in software engineering and dealing with security requirements [36].

In view of the current interest in cybersecurity in research and development, during the RE stage there is explicit attention to systems security. Indeed, in any system the security requirements are the first stage in ensuring that the technical choices made during implementation promote a coherent security system [37]. Generally, in software engineering and RE, security requirements engineering allows information system practitioners, developers and stakeholders to predict attacks and threats before the system has to be deployed in protection [37]. Regardless of these stages, packaged systems that are developed on the basis of general requirements then customized to a client's setting have dissimilar security requirements and thus, in reality, face dissimilar attacks. For this reason, in some contexts applying an Agile approach is a vital move to deal with the security requirements. The methods have been deployed widely in recent decades and are practised across the sector [38].

Agile methods are also used increasingly in other development projects, such as web applications, IoT, cloud computing and network applications, as they offer benefits to the software development world and can change a development project's philosophy. Agile has features that promote security, as one of its main principles states that "Customer collaboration over contract negotiation "[39]. So, the Agile approach works better in small development teams with a private and close infrastructure. This gives the team members an understanding of the actual issues in each system. It shows them how to handle a cybersecurity attack using their close communication with the system engineers and their involvement in critical decisions on the organization's security policies and procedures. Indeed, this generally leads to prompt implementation, in a 'quick fix' approach to any security issue, as identified during the iterations. It establishes an integrated risk analysis and security management in the software development project, alongside meeting the software requirements with due consideration to cost efficiency and quality in both process and products. A valuable perspective is to consider security with attention to productivity and to refer to the production costs.

In summary, the literature review points to the benefits of using Agile in both collocated and distributed development projects. Also, it shows that security requirements have attracted much attention in recent years owing to the high number of cybersecurity attacks on information systems. The following section fills the gap in the literature review concerning the increased risks to a client arising from having an offshore team working on its site on security requirements by proposing a hybrid model, taking an Agile onshore approach to processing security requirements and a traditional approach to the distributed domain.

E. The Onshore Agile Security Requirements Development Model (OASRD)

The challenge for any organization using packaged software products and using offshore development services for security requirements is to send its desired customization requirements to the vendor's offshore team securely while, at the same time, the security requirements are processed locally through an on-site Agile development team. This article focuses on the benefits of using a hybrid approach to overcome these challenges.

The OASRD model is designed to merge two development approaches. The first approach concerns the security requirements, which originate from the detection and analysis unit. This unit responds to any cybersecurity threat to the organization, passing the requirement to the local development team (onshore).

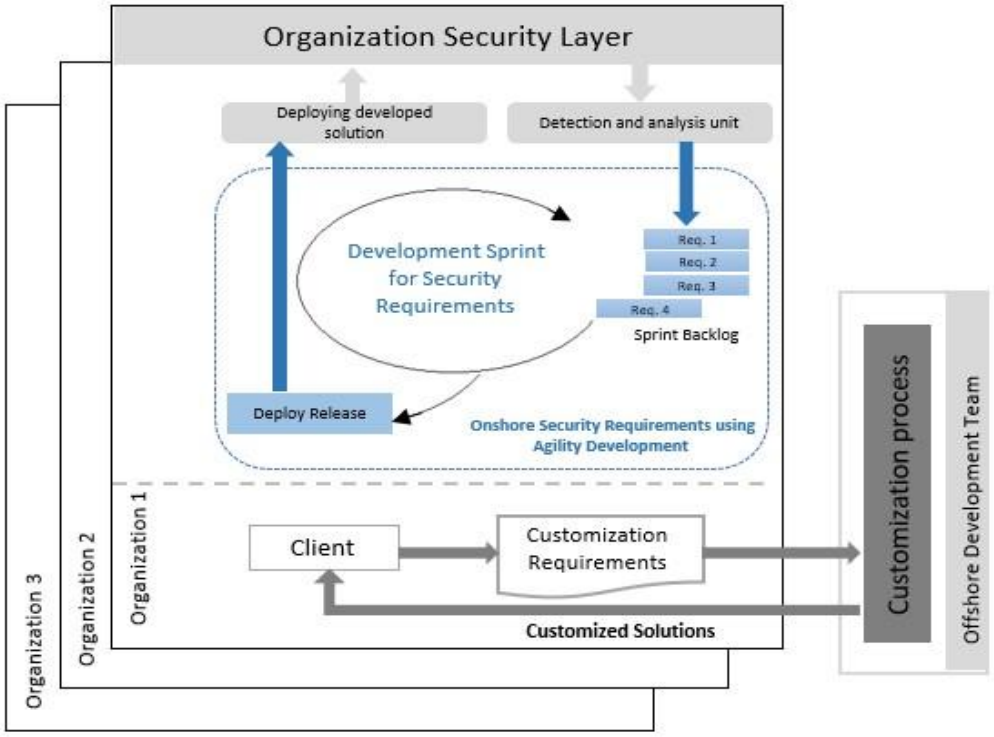


Fig. 2. Onshore Agile Security Requirements Development Architecture model (OASRD)

To develop a solution, the onshore team members use Agile, since this has the benefits of close communication and discussion with the system engineers; moreover, the approach is recommended to solve issues in distributed development [40]. The Agile team has a sound awareness of the organization's infrastructure, making its response to these security requirements accurate, rapid and efficient.

On the other hand, the organization also has to customize the software that it has purchased, so it needs representation by means of customization requirements that are put to the vendor's offshore team, which deals with multiple clients in the distributed domain. This team uses a traditional development approach to fulfilling clients' requirements and improving the new release of the software product. Figure 2 shows the architecture of an organization that uses a hybrid model to manage both its customization and security requirements at the same time. The OASRD model has two parts. The first is the customization requirements, which are derived from the organization's end-user needs, represented by a component 'client'. The requirements can be formulated and sent to the offshore customization team at the vendor's site. Usually, in this domain (customization packaged software for multiple clients), a central team serves many clients to reduce the costs and avoid any redundancy. In this model, the customization requirements are collected from the client site and combined with requirements with respect to priority, then processed using a traditional approach, such as waterfall development, in order to make everything coherent and documented so it may be tracked and used at other sites, if this is possible. The second part of the OASRD model, at the top of Figure 2, takes a lighter approach to managing and developing the security requirements. These are sent by the detection and analysis unit upon identifying a cybersecurity attack or receiving an alert about a new threat. This unit prepares a report on the attack and defines the requirements to be implemented to make the system secure and safe. The requirements arrive at the onshore development team to be split into tasks in the Sprint Backlog. From that backlog, in a prioritization process, the decision is taken by team members to trigger the development process, encompassing design, implementation, testing and evaluation activities in a short period with regular meetings and close discussions to produce a totally deployable released solution. The solution is applied with a high level of involvement by the organization's system engineers and security experts. The OASRD model offers benefits for PS customization, as any security threat in the distributed domain demands a quick and efficient solution. Also, it reduces the time taken to discuss the issue across organizational boundaries and to develop security solutions tailored to the client's infrastructure and organizational setting.

III. THE EMPIRICAL STUDY

In this article, the empirical study is discussed in detail to evaluate the impact of applying Agile principles to customization projects for only the security requirements. It assesses the OASRD model's productivity in terms of the time taken to complete the requirements and the numbers completed in a specific time. The study investigates the impact on project resources in terms of estimated costs and resources used. The study follows the guidelines laid down by Perry et al. [41].

A. Problem Definition

As discussed above, customization across distributed organizational boundaries faces specific, inherent difficulties in communication between the client and the offshore developers. Also, it involves the challenge of understanding the security of and frequent changes to the client's organizational infrastructure [42], as mentioned in studies such as [43].

B. The Hypotheses and Research Questions

A quick response to security threats and attacks is the main goal of all system engineering. This is not easy for a totally distributed team. The OASRD model reduces the challenges by making the process of developing security requirements local, using an Agile onshore team. The hypotheses defined by this study are:

H1: Adopting an Agile development method to handle security requirements in customization projects increases the productivity of the development of security requirements.

H2: Applying the locality concept to develop security requirements through an Agile development process decreases the cost of the customization.

The objectives of this study are to evaluate the OASRD model by measuring the number of completed requirements and the average time taken to complete the security requirements, and to increase development productivity alongside PS customization.

To meet the research objectives and accomplish these aims, the following research questions were formulated:

RQ1: What is the impact of developing security requirements onshore using an Agile approach on the productivity of the customization process in a distributed development domain?

What is the impact of developing security requirements onshore using an Agile approach on the cost of the customization process in a distributed development domain?

C. Study Measurements

To meet the research goals and answer its questions, this study measured the OASRD technique's productivity and cost in order to inspect the defined hypothesis. The first task was to compute the productivity in terms of numbers of completed requirements and to estimate running costs in terms of the salaries of those involved (both the onshore and offshore teams).

D. Experimental Instrumentation

1) Implementation tool

In software engineering research, software simulation is used for various purposes [44], for example process improvement control, the operational management of software engineering and the strategic development of software. According to Oden et al., [45], modelling and simulation have become popular methods in research and industry in fields such as engineering, business and medical science. Applying and controlling research studies in software engineering involves issues relating to time, cost and resources, and simulation can be an alternative, using real data to drive the experiment. In this study, the evaluation used simulation as its primary technique to test the OASRD model against a defined hypothesis.

The simulation tool used here is a simulation package, SIMUL8 [46]. The SIMUL8 Corporation offers an integrated environment, as used by many industrial and academic organizations [47] to design, build and control simulation models, as well as powerful language and model visualization.

2) *Objects of analysis*

One of the most appropriate methodologies in software engineering research is the case study, as it adds value and understanding to the subject by investigating phenomena in their natural context [48]. Also, empirical research in this field indicates the vital importance of further such studies [49]. This research started with a contextual, real-life case study to understand the customization scenario and to collect historical data to drive an appropriate simulation. The case study was conducted on an actual project run by a PS vendor of products developed to be customized to a client's requirements. The company in question has more than five distributed health-sector clients involved in the project. Figure 3 shows a conceptual model of the customization process, reflecting the reality of the selected case study. In this scenario, the customization and security requirements are specified by the user and sent to the offshore team at the vendor's site. After processing, the team duly delivers the solution to the client. This involves various difficulties, such as in communication and a lack of awareness of the client's overall infrastructure.

3) The study scenario

In this section, the setting and procedures are discussed, starting with data collection from the case study. Figure 4 builds from a reflection of the observed, real-world scenario of the case study into a simulation to construct the baseline model from historical data gathered in the contextual review. Those data document the processes involved in the customization process, involving more than 2700 requirements and over 254 security requirements (arriving as bugs and security issues), using a tracking system known as JIRA [50]. The data were collected over approximately 900 working hours on more than five distributed clients at the same time. The next stage constructed the simulation shown in Figure 2, using the same data set as the baseline model yet assigning the Agile development team to the client's site to handle the security requirements, as in the OASRD model. The third stage ran both the baseline and the OASRD simulation to evaluate them, as in Figure 4, using random data for the client requirements for customization. In both, the arrival data were generated by using a Poisson distribution of 0.365 to fit the case study's historical arrival data, while the security requirements were generated using uniform distribution, with U values of 0.75, 15. The outputs were compared by a statistical t-test to establish whether the OASRD model, by using onshore development to implement the security requirements and an offshore team to implement the customization, makes a statistically significant effect on productivity. The final stage was to conduct confirmatory interviews at the vendor company in question to gain experts' views on the results achieved.

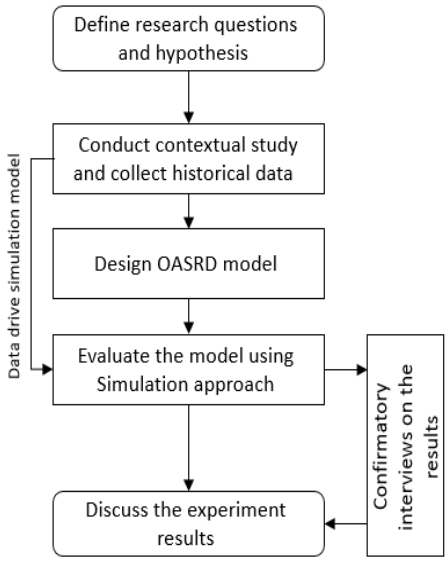


Fig. 4. Design to evaluate the OASRD model

IV. DATA AND ANALYSIS

Productivity is one of the commonest measures of the quality of a software process. Therefore, this study defined its main goal as the examination of the impact of using the OASRD model on customization in a distributed project. This was carried out by comparing simulations of the baseline and the OASRD models.

Figure 5 shows the number of customization and security requirements completed in each model, calculated by running the simulations for 480 minutes. The histogram, exported from the simulation tool, represents the high, low and average number of completed requirements after running the models 100 times.

To conduct statistical analysis of these figures, a paired sample t-test was undertaken, and the results are shown in Table I The mean numbers of completed customization and security process requirements in 480 minutes of simulation time were 918 under the baseline model and, with the same settings and conditions, 1,534 under the OASRD model. The P value was less than 0.05, which means that there is a significant difference between the models' productivity, expressed in terms of completed requirements. Overall, the results of the paired sample t-test indicate that the outputs of the baseline simulation and the OASRD model show a significant difference in productivity: the OASRD model is nearly twice as productive, with $P < 0.05$ at the 95% level of confidence. That means the null hypothesis can be rejected and the alternative hypothesis accepted; that is, there is a significant difference between the baseline simulation and the OASRD model, which uses a local Agile development approach for security requirements, in terms of their numbers of completed customization and security requirements.

Table I: Results of paired sample t-test

Model	N[1]	Mean[2]	Std dev	P value
Baseline model	20	918	0.24	0.0001
CCRD model	20	1534	0.52	(p< 0.05)

In addition, the participants in the confirmatory study' interviews emphasized the importance of discussing a client's requirements at the client's location in an Agile manner, as the following interviewee states:

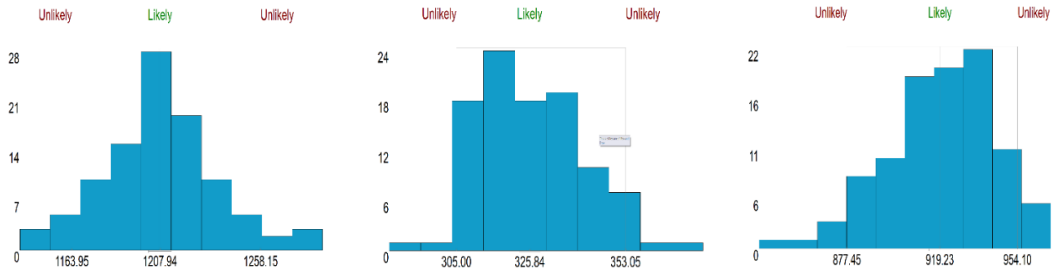
> We had an experience with a one client who hasn't an implementer at his location. We stopped dealing with them after a week due to them not understanding the business or the system either. (Int. 1)

In the same vein, interviewees emphasized the benefits of taking decisions and carrying out development at the client's location, as this made negotiations on security requirements easier. As the following interviewees mention:

> I expect that if the decision were made at the client's location it would reduce the time for decision and the entire process. (Int. 2)
>
> I agree, moving decisions from the central to the client's location in some cases and special requirements would reduce the challenges of communication. (Int. 3)

In sum, both the significant statistical results and the interview findings of the confirmatory study indicate the benefits of using Agile to address the security requirements in customization for distributed clients. This finding answers the research question (RQ1) and meets the defined hypothesis (H1).

Table II presents the estimated cost of the customization and development of security requirements in the distributed domain under the baseline and OASRD models, respectively. It shows the number of developers and vendor's representatives alongside the cost in each.



a: OASRD model – customization requirements

b: OASRD model – security requirements

c: Baseline model – security & customization requirements

Fig. 1. No. of requirement processes in the baseline and OASRD models for: (a) customization requirements; and (b) security requirements (over 100 runs)

Table II: Cost estimate in OASRD model

		Baseline model	**OASRD model**
No. of representatives at client site		5	5
No. of senior developers		8	3
Cost	Representatives:	5 * £30,249 = £15,1245	5 * £30,249 = £151,245
	Senior developers:	8 * £63,537 = £50,8296	3 * £63,537 = £190,611
Total cost		**£659,541**	**£ 341,856**

This result was collected from the case study conducted on five distributed clients, who communicated with the offshore development team in order to customize their software product. In the first column, the baseline model has five vendor representatives at the client's site. These representatives manage the client's requirements by collecting them, then sending to the offshore team for development. In a software sector environment, the representative at the client's site is generally a software engineer, as this role needs to undertake the collection of the analysis requirements and also to implement the final solution on the client's infrastructure, whether the changes originate from the offshore or onshore development team. The developers in each team should be senior software developers, so that they are aware of the client's contract and can assess the changes.

The baseline model has five representatives at client sites and eight senior offshore developers, according to the data collected. The number of representatives remains the same in both the baseline and OASRD models, whereas under the proposed model the number of senior developers drops from eight to just three, all allocated to the offshore site. On the basis of the salary for a UK software engineer shown on the website of Glassdoor [37], which anonymously collects and analyses actual salaries at a variety of large companies, the cost thus plummets from £659,541 to £341,856 (48.17%).

At this point, the defined hypothesis H2 has been confirmed and answered by the research question (RQ2), indicating a reduction in estimated costs under local Agile development for security requirements.

V. CONCLUSION

This study conducted an empirical investigation into the effectiveness, in terms of productivity, of the Agile approach to developing security requirements for customization in a distributed domain. This article proposes a model for the enhancement of Agile development for security requirements: the OASRD model.

The empirical study gauges the productivity of OASRD model through the number of completed security and customization requirements, and estimates the reduction in the total cost of developing the customization and security requirements in the distributed domain in terms of the salaries of the local and offshore developers and the local vendor's representatives. The study uses an actual case study of a vendor's five distributed clients in a customization project involving more than 2,700 requirements, collected over 900 working hours.

REFERENCES

1. E. Del Nuevo, M. Piattini, and F. J. Pino, "Scrum-based methodology for distributed software development", in Proceeding of 6th IEEE International Conference on Global Software Engineering, ICGSE 2011, 2011, Helsinki, Finland.
2. A. M. Qahtani, Gary Wills and Andy Gravill, "The Impacts of Local Decision Making on Customisation Process Speed across Distributed Boundaries: A Case Study", International Journal of Computer, Information, Systems and Control Engineering Volume 9, No:1, 2015.
3. R. Vallon, B. J. da Silva Estácio, R. Prikladnicki, and T. Grechenig, "Systematic literature review on agile practices in global software development," *Information and Software Technology*, 2018, doi: 10.1016/j.infsof.2017.12.004.
4. J. Dick, E. Hull, and K. Jackson, *Requirements engineering*. 2017.
5. S. Turpe, "The Trouble with Security Requirements," 2017, doi: 10.1109/RE.2017.13.
6. S. I. Hashmi, F. Ishikawa, and I. Richardson, "A communication process for global requirements engineering," 2013, doi: 10.1145/2486046.2486070.

7. E. B. Swanson and P. Wang, "Knowing why and how to innovate with packaged business software," *Journal of Information Technology*, 2005, doi: 10.1057/palgrave.jit.2000033.
8. S. Schneider, R. Torkar, and T. Gorschek, "Solutions in global software engineering: A systematic literature review," *International Journal of Information Management*, 2013, doi: 10.1016/j.ijinfomgt.2012.06.002.
9. B. Sengupta, S. Chandra, and V. Sinha, "A research agenda for distributed software development," 2006, doi: 10.1145/1134285.1134402.
10. R. Colomo-Palacios, C. Casado-Lumbreras, P. Soto-Acosta, F. J. García-Peñalvo, and E. Tovar, "Project managers in global software development teams: a study of the effects on productivity and performance," *Software Quality Journal*, vol. 22, no. 1, pp. 3–19, Jan. 2014, doi: 10.1007/s11219-012-9191-x.
11. Z. U. R. Kiani, D. Smite, and A. Riaz, "Measuring Awareness in Cross-Team Collaborations -- Distance Matters," *2013 IEEE 8th International Conference on Global Software Engineering*, pp. 71–79, Aug. 2013, doi: 10.1109/ICGSE.2013.17.
12. R. S. Pressman, *Software Engineering A Practitioner's Approach 7th Edition*. 2010.
13. R. Prikladnicki, J. L. Nicolas Audy, and R. Evaristo, "Global software development in practice lessons learned," *Software Process: Improvement and Practice*, vol. 8, no. 4, pp. 267–281, Oct. 2003, doi: 10.1002/spip.188.
14. V. Garousi, K. Petersen, and B. Ozkan, "Challenges and best practices in industry-academia collaborations in software engineering: A systematic literature review," *Information and Software Technology*. 2016, doi: 10.1016/j.infsof.2016.07.006.
15. D. Šmite, C. Wohlin, T. Gorschek, and R. Feldt, "Empirical evidence in global software engineering: A systematic review," *Empirical Software Engineering*, 2010, doi: 10.1007/s10664-009-9123-y.
16. W. Alsaqaf, M. Daneva, and R. Wieringa, "Agile Quality Requirements Engineering Challenges: First Results from a Case Study," 2017, doi: 10.1109/ESEM.2017.61.
17. T. Dingsøyr, K. Rolland, N. B. Moe, and E. A. Seim, "Coordination in multi-team programmes: An investigation of the group mode in large-scale agile software development," 2017, doi: 10.1016/j.procs.2017.11.017.
18. D. E. Damian and D. Zowghi, "RE challenges in multi-site software development organisations," *Requirements Engineering*, vol. 8, no. 3, pp. 149–160, 2003, doi: 10.1007/s00766-003-0173-1.
19. M. A. Akbar, J. Sang, A. A. Khan, and S. Hussain, "Investigation of the requirements change management challenges in the domain of global software development," *Journal of Software: Evolution and Process*. 2019, doi: 10.1002/smr.2207.
20. L. Xu and S. Brinkkemper, "Concepts of product software," *European Journal of Information Systems*, vol. 16, no. 5, pp. 531–541, 2007.
21. L. Xu and S. Brinkkemper, "Concepts of Product Software : Paving the Road for Urgently Needed Research," in *CAiSE Workshops*, 2005, pp. 523–528.
22. I. F. Da Silva, P. A. Da Mota Silveira Neto, P. O'Leary, E. S. De Almeida, and S. R. D. L. Meira, "Software product line scoping and requirements engineering in a small and medium-sized enterprise: An industrial case study," *Journal of Systems and Software*, 2014, doi: 10.1016/j.jss.2013.10.040.
23. A. E. Akgün, "Team wisdom in software development projects and its impact on project performance," *International Journal of Information Management*, 2020, doi: 10.1016/j.ijinfomgt.2019.05.019.
24. M. Sciences, "Packaged Software Implementation Requirements Engineering by Small Software Enterprises : An Ethnographic Study Issam Jebreen," 2014.
25. S. Sawyer, "Software Development Team Performance," *Information Systems Journal*, vol. XY, no. IS, pp. 155–178, 2001, [Online]. Available: http://www.dimap.ufrn.br/~jair/ES/artigos/SoftDevTeam.pdf.
26. M. T. Sletholt, J. Hannay, H. C. Benestad, and H. P. Langtangen, "A Literature Review of Agile Practices and Their Effects in Scientific Software Development," *Change*, pp. 1–9, 2011.
27. A. Azanha, A. R. T. T. Argoud, J. B. de Camargo Junior, and P. D. Antoniolli, "Agile project management with Scrum," *International Journal of Managing Projects in Business*, 2017, doi: 10.1108/ijmpb-06-2016-0054.
28. S. Alam, S. Nazir, S. Asim, and D. Amr, "Impact and Challenges of Requirement Engineering in Agile Methodologies: A Systematic Review," *International Journal of Advanced Computer Science and Applications*, vol. 8, no. 4, pp. 411–420, 2017, doi: 10.14569/IJACSA.2017.080455.
29. J. Shore and S. Warden, *The Art of Agile Development*. O'Reilly Media, Inc., 2007.
30. O. Sievi-Korte, S. Beecham, and I. Richardson, "Challenges and recommended practices for software architecting in global software development," *Information and Software Technology*, 2019, doi: 10.1016/j.infsof.2018.10.008.
31. M. S. Filho, P. R. Pinheiro, and A. B. Albuquerque, "Task allocation approaches in distributed agile software development: A quasi-systematic review," 2015, doi: 10.1007/978-3-319-18473-9_24.
32. R. Phalnikar, V. S. Deshpande, and S. D. Joshi, "Applying Agile Principles for Distributed Software Development," *2009 International Conference on Advanced Computer Control*, pp. 535–539, 2009.
33. S. Lee and H.-S. Yong, "Distributed agile: project management in a global environment," *Empirical Software Engineering*, vol. 15, no. 2, pp. 204–217, Oct. 2009.
34. R. Anwar, M. Rehman, K. S. Wang, and M. A. Hashmani, "Systematic Literature Review of Knowledge Sharing Barriers and Facilitators in Global Software Development Organizations Using Concept Maps," *IEEE Access*, 2019, doi: 10.1109/ACCESS.2019.2895690.
35. S. Jalali and C. Wohlin, "Agile practices in global software engineering - A systematic map," 2010, doi: 10.1109/ICGSE.2010.14.
36. A. Souag, R. Mazo, C. Salinesi, and I. Comyn-Wattiau, "Reusable knowledge in security requirements engineering: a systematic mapping study," *Requirements Engineering*, 2016, doi: 10.1007/s00766-015-0220-8.
37. M. Daneva and C. Wang, "Security requirements engineering in the agile era: How does it work in practice?," 2018, doi: 10.1109/QuaRAP.2018.00008.
38. T. D. Oyetoyan, D. S. Cruzes, and M. G. Jaatun, "An empirical study on the relationship between software security skills, usage and training needs in agile settings," 2016, doi: 10.1109/ARES.2016.103.
39. K. Beck *et al.*, "Manifesto for Agile Software Development," *The Agile Alliance*, 2001. .
40. A. Gopal, J. A. Espinosa, S. Gosain, and D. P. Darcy, "Coordination and Performance in Global Software Service Delivery : The Vendor ' s Perspective," vol. 58, no. 4, pp. 772–785, 2011.
41. D. E. Perry, A. A. Porter, and L. G. Votta, "Empirical studies of software engineering: A roadmap," 2000, doi: 10.1145/336512.336586.
42. M. Zahedi, M. Shahin, and M. Ali Babar, "A systematic review of knowledge sharing challenges and practices in global software development," *International Journal of Information Management*, 2016, doi: 10.1016/j.ijinfomgt.2016.06.007.
43. A. M. Qahtani, G. B. Wills, and A. M. Gravell, "The Impacts of Local Decision Making on Customisation Process Speed across Distributed Boundaries : A Case Study," vol. 9, no. 1, pp. 63–68, 2015.
44. M. I. Kellner, R. J. Madachy, and D. M. Raffo, "Software process simulation modeling: Why? What? How?," *Journal of Systems and Software*, vol. 46, no. 2, pp. 91–105, 1999, doi: 10.1016/S0164-1212(99)00003-5.
45. J. Oden, T. Belytschko, J. Fish, and T. Hughes, "Revolutionizing Engineering Science through Simulation," *Blue Ribbon Panel on Simulation-Based Engineering Science*, 2006.
46. A. Greasley, "SIMUL8 Simulation Package," in *Wiley Encyclopedia of Management*, 2015.
47. K. H. Concannon, K. I. Hunter, and J. M. Tremble, "Simul8-planner simulation-based planning and scheduling," 2003, doi: 10.1109/wsc.2003.1261593.
48. P. Runeson and M. Höst, "Guidelines for conducting and reporting case study research in software engineering," *Empirical Software Engineering*, vol. 14, no. 2, pp. 131–164, 2009, doi: 10.1007/s10664-008-9102-8.
49. T. Ambreen, N. Ikram, M. Usman, and M. Niazi, "Empirical research in requirements engineering: trends and opportunities," *Requirements Engineering*, 2018, doi: 10.1007/s00766-016-0258-2.
50. Atlassian, "JIRA Software," *La herramienta de desarrollo de software líder de los equipos ágiles*, 2017.

AUTHORS PROFILE

Dr Abdulrahman Qahtani is currently Assistant professor in computer science department at Taif UNiveristy. Dr. Qahtani received his PhD from Southampton University in 2015. He has extensive experience in software engineering and development process in a distributed domain. His research focused on customisation process across organisational boundaries. Recently, he applied machine learning algorithms on software engineering data to predict time and cost estimation for multi-clients projects.

.