A Critical Review of the Use of Spikes in Agile Software Development

Hussein Al Hashimi[1] and Andrew M Gravell [2]

[1,2] University of Southampton, Southampton, UK
[1]King Saud University, Riyadh, KSA
email: hah2n17@soton.ac.uk, amg@ecs.soton.ac.uk

***Abstract* – Spikes can be an essential component in the agile development cycle since they assist teams in both technical and functional issues in order to obtain the information required to reduce technical risk, understand requirements or enhance the accuracy of a story estimate. Spike is a time-boxed activity to explore and investigate a significant uncertainty and various technical approaches in order to obtain a demonstrable and estimable user story. This paper reviews the uses of spikes through findings in different software development domains to showcase the implementation of spikes in agile software development and their impact on the understanding, consistency, and reliability of the story estimate. The paper provides a critical review of the use of spikes in various software projects and it concludes that limited studies have been conducted on the use of spikes in different software development domains.**

Keywords - Agile; Spikes; Risk management; Uncertainty.

I. Introduction

Spike usage in agile software development was initially defined in the Extreme Programming (XP) approach because spikes represent prototyping, exploration, and investigation, design, and research activities [14]. At the end of the iteration, the spikes in agile are demonstrated and estimated as similar to other stories [26]. Furthermore, they are also responsible for providing the workflow and protocol that are used by the Agile Release Trains (ARTs). The ARTs were developed by Scaled Agile Framework (SAFe). The ARTs are virtual institutions (between 50-125 people) where all persons constitute a self-organising team of experts needed to determine and deliver value by preparing, engaging, and implementing. ARTs help determine the viability and feasibility of epic user stories [23]. The major goal of integrating spikes in agile is to increase the possibility of user story estimation and to minimise technical problems [26]. Spikes can be defined as a particular type of story for the purpose of performing several activities such as research, investigation, exploration, prototyping, and design with the purpose of reducing or driving out uncertainty or the technical risk associated either with the user story or with a project facet [26]. For instance, spikes are required when the agile team needs to resolve a specific technical problem or when they do not have enough information for user story estimation [26].

There are two major types of spikes, functional and technical spikes [26]. Functional spikes are utilised particularly when there is a significant degree of uncertainty. This uncertainty is defined in terms of a lack of understanding of how the system should interact with the user to attain the required benefit. The best way to evaluate the functional spikes is to utilize the prototyping levels by using the user interface mockups, page flows, and the wireframes. However, the other techniques are also utilized in order to get the potential feedback from stakeholders or customers. On the other side, the technical spikes are used for the purpose of conducting technical approaches with respect to the solution domain [26].

Spikes are used for driving out risk and uncertainty in agile software development. There has been a marked shift from traditional software development towards agile methodologies. However, it is important to note that spikes are never really followed in their original forms. Most software developers agree that there is a need for advanced professional skills before a spike can be implemented in software development [26].

At the level of the agile team, spikes are important for extending the runway and thus prioritising other stories. This makes software development visible, accountable and demonstrable at every iteration boundary. This is significant because spikes are used by system architects, product owners, and agile tech leaders to determine what needs to happen and when. Spikes are also important in situations where the story is too large or complex, or the implementation is poorly understood. A technical or functional spike can be built to resolve this impasse. Then based on the result, the stories are split.

Spikes were invented by XP, and are used to remove risk and uncertainty in a user story. Spikes may be used for research, educating the team about new technology, analysing implied behaviour in a large story before it is split into manageable parts, or for prototyping to identify significant risks in a story before committing the user story to some future time box [26]. However, spikes do not provide direct user value, and therefore should be adopted with caution and used carefully in software development. Moreover, it is important to study the use of spikes because the output from spikes is used for information rather than the working code that a typical story will derive. Spikes are important for delivering sufficient, useful information about the story to the agile team. They give visibility to the software development effort, are demonstratable, build collective ownership, and the responsibility for key decisions taken is shared. Therefore, a

ISBN: 978-1-61208-752-8

spike story is usually reserved for larger critical unknowns. A spike is an uncertainty in one or more potential stories; a problem may arise when the spike and the accompanying story are planned in the same iteration. This should be avoided [26][41].

Agile software development is an approach that is currently being adopted by technological- and interactive-based software programming languages. In the course of examining spike usage, this paper begins with a discussion of relevant background and related works in Section II. In Section III, the paper focuses on the application of spikes within the domains of the agile software development approach. Section IV reflects on the influence of spikes on quality in agile software development. In Section V, IT-related security issues are outlined, followed with a discussion about the identified facts in Section VI alongside recommendation and potential future work in Section VII.

II. Background and Related Works

Agile is a philosophy and a method for building and releasing software products. Agile is a group of software engineering methodologies made on similar rules and principles and provides a platform for helping teams, giving a continuously changing functional and technical environment. A focus on fast delivery of business value is maintained [29]. As a result, this approach reduces the risks associated with software development. Agile software development is a blanket term for a group of processes and techniques based on the principles and values described within the Agile Manifesto [5].

A. *Agile Methodologies*

There are several agile methodologies. The most widely used agile methodologies are Extreme Programming (XP), Scrum, Dynamic System Development Methodology (DSDM), Lean, Crystal, Feature Driven Development (FDD) and Kanban [29].

XP is a disciplined approach to producing high-quality software instantly and consistently. Extreme programming supports high customer engagement, instant feedback loops, consistent planning, consistent testing and production of working software at regular intervals and within very short periods of time. Clients work in partnership with the development team to define and prioritise user stories [6].

Scrum is a framework in persons which identify and resolve adaptive problems that are complex, while creatively and effectively making products that have supreme value. The Scrum framework is usually a Scrum Team with its affiliated functions, rules, artifacts and regular events that put them together with each other [17].

Kanban is a Japanese word and its meaning linked to a time theory, “just in time” created by Toyota in its manufacturing process and applied to software development. The basic concepts of the Kanban method are visualising the flow of work, reducing work in progress, and enhancing the flow of work. The Kanban method matches Scrum in many ways. Both of them are agile, having transparency across the development, and use pull scheduling. Both of them limit the amount of work in progress, the Kanban method at task level and Scrum at sprint level. Both Scrum and the Kanban method focus on early delivery of the releasable software built and require splitting the work into pieces, which is divided into self-organising teams. In both methods, the release plan is also continually optimised depending on empirical data [28].

B. *Roles in Agile*

An agile team is a multi-functional team of professionals with the necessary resources for the development of a working, properly tested release of a product. In Scrum, the second most significant role is that of the Scrum Master; this individual is responsible for the proper execution of the Scrum methodology as well as removing any potential obstacles that may be encountered by the development team. Accordingly, the Scrum Master is a servant-leader for the team and is not only responsible for promoting and supporting Scrum, but assisting everyone on the team to understand the Scrum theory, practices, value and rules [42]. The Scrum master tracks the project’s progress and each individual’s input and ensures the conduct of Scrum meetings, planned meetings, demos, reviews, and retrospective meetings [17]. Another role—the product owner—drives the product from the business angle by defining the requirements, evaluating their priority, and determining the date and contents of that release. This person takes an active role in planning the iteration and release meetings as the client’s voice. The product owner also accepts and evaluates user stories that meet the defined acceptance criteria and those definitions of done [27]. An agile team is self-sufficient with five to nine members who have an average working experience of around 6 to 10 years. Typically, a team comprises three or four developers, a tester, a technical lead, a product owner, and a Scrum master. The agile team uses its expertise to work on tasks and to decide and plan the scope of work [17].

C. *Agile Spikes*

Spikes in agile are stories that are estimated only after a development team completes a time boxed investigation. A time box is a defined period of time to accomplish a task in agile. Defined initially in XP, these spikes represent activities such as prototyping, design, research, investigation, and exploration. The purpose behind using spikes is to gather the information needed to better understand the requirement, lower the project risk, and make the story estimate more reliable. Spikes also provide a mutually decided workflow and protocol. At the end of each iteration, spikes are estimated and demonstrated like any other agile story [26]. Spikes are put into the team backlog to fit in to the iteration. The output of a spike is demonstrable, acceptable, and quantified for both the team and any other stakeholders [26]. Planning a spike and the resulting story in the same iteration is risky; however, every user story has uncertainty and risk, so it is necessary for an agile team to learn how to address uncertainty in each iteration using spikes [26][37].

A spike can be described as an investigation that aims to gather information about a project that would otherwise be unavailable. This makes the project more predictable and therefore easier to plan. However, spikes are short in duration, and as a result, spikes may lead to an outcome that is different from what was predicted. A Proof of Concept (POC) is similar to spikes in that it is also an activity that aims to provide more information to developers prior to planning. The only difference between the two is that a spike is focused on discovering complexities, while a POC is used to determine whether a project is worth undertaking [24]. Similar to spikes, the POC also runs for a short period of time because of the deadline that limits it in order to show that the system will successfully run. A POC is also important because it provides information that gives assurance that more investment in the project is not a waste of resources and will lead to positive results. Lastly, the Minimum Viable Product (MVP) aims to test the most basic business hypothesis. This makes MVP the product while Spikes and POCs are tools and techniques used to create the product. The MVP can also be described as a prototype or the first functional version of the product, in most cases, the MVP is used to provide evidence that the product is viable and that it has the potential to solve the problems that it was meant to solve. Most importantly, an MVP can be used to get user opinions and insights about how the product needs to be improved so that it can be perfected and made ready for use. When considering spikes, POCs and MVPs in software development, it is worthwhile to note that the spike is more related to the principles of agile development than either of the other terms; it can be considered a reconnaissance mission to gain a better understanding of a particular aspect of the software development process [43][44].

D. Agile Risk Management

In the software world, risk is the factor that influences the project's success. Due to the many risk factors that can contribute to the working of software, risk management is required in software engineering and development [10].

For projects on agile approaches, formal documentation and meetings are not required for Risk Management. Instead risk management is split into Scrum roles, artifacts and events. It can easily eliminate so many risks in the agile projects by following the principles of agile. These principles significantly mitigate and eliminate the risks that can lead to project challenges and future failures [29].

In the Agile methodology, the spike is used for identifying an issue and providing a short confirmation of an idea to examine an issue further. Spikes also incorporate testing distinctive strategies to accomplish a similar outcome, just like testing to affirm that the ideal outcome is achievable through the present ventured approach. For instance, a group may play out a spike to check whether they can code an application in one language rather than another. A risk-based spike is completed in light of a known risk or openings for project risks [30]. The group may discover that using an alternative programming language can accelerate advancement, or they may discover that they cannot use the one they originally wanted to use. These spikes are added to the backlog as alleviation activities [12].

Risk-based spikes help the agile team eliminate or minimise major risks. Risk-based spikes are used with consideration of fast failure, i.e., if any spike fails for every available approach, the project goes into fast failure, which costs much less than failing late [12].

III. SPIKES IN AGILE SOFTWARE DEVELOPMENT DOMAINS

In agile development spikes are used by the software teams for investigating, closing gaps, and reducing risks. In agile development, the spike is the story that cannot be estimated specifically when the development team is running the time-boxed investigation. The result of the spike is the estimate with respect to the original story. The spike is the "time-boxed" technical investigation that is responsible for generating the answers in accordance to some acceptance criteria on Product Backlog Items (PBIs). These product backlog items are prioritised in the upcoming sprints [34]. In addition, spikes originated in extreme programming where they are considered the special story type that is used specifically for driving out uncertainty and risky elements in either a project facet or user story [26]. Table I shows the agile spike uses in various software domains. It makes evident the limited information available about spike usage.

A. Agile Development for Big Data

Big data is converting business landscapes as the most significant technology in academic ecosystems and business since the dramatic growth of the digital economy and the Internet. Big data refers to data sets that are too enormous or complex for traditional data-processing application software to deal with adequately [12]. According to the Big Data Software Engineering (BIGDSE) workshop, big data systems pull out high-value information to enhance decision-making in Business, science, and society [8]. Performing big data analytics by using agile development methodologies is fairly new and needs wary adaptation as big data analytics are intensely different from "small" data analytics. The requirements of big data analytics require capabilities beyond traditional relational data warehouses [12].

In agile development, spikes can play a major role in data development as it integrates guidelines to make sure that user stories and data are quantifiable, demonstrable, and acceptable. Big data represents parallel processing and data distribution to make sure that data analytics, algorithms, storage lifecycle are not separated from the big data technologies. For such a reason, architecture-supported agile spikes were introduced to make sure that rapid technology changes and new requirements are addressed effectively [15].

B. Agile Development for Data warehouse

The traditional data warehouse projects follow the waterfall development model where they have to complete all six phases—gathering requirements, designing, developing, testing, deploying, and stabilising. In this model, both the technology and business requirements are complex and critical in nature, and approximately six to nine months are required to complete the traditional waterfall model to ensure full implementation. After advances in technology or changes in requirements, it becomes difficult and challenging for the software team to incorporate all the changes without changing the basic architecture of the software system, resulting in frustrated development teams and disappointed stakeholders. On the other side, agile development integrates the solution in an iterative fashion for which it is also known as the "60% solution" [32]. The agile approach assists in delivering the client needs in previous releases with refinements on the subsequent routine release series. To accommodate this task, the agile data warehousing approach improves successful implementation within budget and on time. For data warehouses, the incorporation of spikes may boost agile development by increasing efficiency and functionality.

C. Agile Development for Computer Science Education

Just like software project teams use agile spikes to reduce risks, close gaps, and investigate in software development, computer science education can benefit by doing the same while incorporating the techniques of agile development. When practicing the agile development approach stages, software development teams can identify and define the concealed aspects to help determine the steps that need to be covered for related gaps. Spikes include the production and development of the working piece of the software project to address the particular issue. The spike metaphor represents the very specific complete or end-to-end solution. Software projects produced this way are either discarded or simply not used as the final product [38].

In information and communication technology, computer science education is considered a fundamental platform for enhancing the programming skills of students [38]. The agile software development process has a collective nature in which the software team resolves the typical issues in conventional software development processes. In the software industry, the Agile Manifesto is considered the appropriate introduction to the agile approaches [8][38].

Woodward, Montgomery, Vasa, and Cain [38] suggested that spikes are not suitable for freshman students in campus because, at a pre-existing stage, the overhead of the compound skills (e.g., the software development approaches) with respect to spikes prevails over the possible potential benefits.

D. Agile Development for Blockchain

The agile methodologies are integrated in adaptive planning because it provides potential support in continuous improvement, which allows the life cycle to respond to changes easily and quickly. These procedures depend on key principles that are organised in different phases [27]. The decentralised technologies and blockchain approach together allow for new possibilities in offering the value of digital and software products to users. Agile development involves integration and transition, whereas blockchain offers a wide variety of possibilities especially when it involves system designs [19]. This creates a platform of uncertainty when a company incorporates the new technology; however, the involvement of spikes may be revoking the level of insecurity. Agile development involves conducting different tests to ascertain how different components involved in the implementation stage would work with software. The involvement of spikes could verify the uncertainty by providing information about the risk a company is bound to encounter through the integration of blockchain and agile development.

E. Agile User Experience Design (UX)

The agile and user experience (UX) methods can coexist well only if the management of the organisation supports and understands user experience work, user experience practitioners spend time and show leadership in reaching out to their colleagues, the agile workflows provide flexibility for accommodating user experience needs, and the product teams are composed of user experience professionals, where they can build rapport and respect with the developers [13]. On the other hand, lean UX is an essential technique that is used in projects which utilise the agile development method. Lean UX depends on teamwork; its main aim is to ensure that feedback is received at the earliest possible time. This makes it possible to make quick decisions that match the rapid nature of agile development. Lean UX differs from traditional UX as it focuses on detailed deliverables; the developer's priority is to make changes that make the product better at the current time [45].

Spikes in agile can be used to incorporate user experience design work or monitor user research; however, their main purpose is to handle risk issues in the implementation solutions. Normal planning in spikes should easily predict and anticipate design and research events. Spikes should be responsible for managing risk issues, such as a design task that needs an enquiry into available technology before it can be estimated or any uncertainty that occurs and requires user research to be well defined [9].

F. Agile Development in Cloud Computing

The agile methodology is used in software development to cope with the issues of traditional project management where the entire project is preplanned regardless of scope for changing requirements and the assumption that cost and time variables are fixed issues. Agile methodology succeeds by focusing more on collaborative effort to achieve results rather than a predefined process. It is an iterative effort.

The concern solved by agile methodology is the provision of a safe framework for the sharing of information. In cloud computing, there is massive data generation, and through software development, applications are developed to manage these data. Therefore, by design, the aspect of information sharing is considered. The technology provides mechanisms for easily blocking the interception of information while it is in transit, courtesy of the inbuilt application security control capabilities [46]. Effective implementations of the agile methodology help development teams respond to sprints [39]. With the latest advancements, cloud computing has gained popularity, and many entrepreneurs and organisations have adopted cloud hosting services because computing has the advantage of providing a platform for organisations to manage their activities more efficiently [21]. By adopting cloud hosting services, the organisations have been able to respond to the evolving needs in the information technology sector accordingly.

TABLE I: AGILE SPIKES IN DIFFERENT DOMAINS

Spike's Theme	Research paper	Purpose of Study	Conclusion
Risk-based spike	Chen, Kazman, and Haziyev (2016)	To use spikes for addressing the risks of a project by breaking down the story into smaller components	Using an alternate programming language can accelerate advancement, or reveal that the language chosen cannot be used. These spikes would be added to the excess as alleviation activities.
	Albadarneh (2015)	To use spikes in testing to affirm that the ideal outcome is achievable	Spikes incorporate testing distinctive strategies to accomplish a similar outcome, just like testing to affirm that the ideal outcome is achievable through the present ventured approach.
Spikes in agile development for big data	Arndt (2018)	To implement agile approaches on big data for better handling and mutual enrichment	Big data analytics and techniques can help improve the software engineering process.
	Chen, Kazman, and Haziyev (2016)	To effectively implement Agile approaches on big data projects for reducing the risk exposure	Existing agile analytics development methods have no architecture support for big data analytics, but they can help to tame project complexity, reduce uncertainty, and hence reduce project risk.
	Larson and Chang (2016)	To apply agile methodologies and principles to BI delivery and explore how agile has changed with the evolution of BI	Spikes are applicable as the principles of agile can be implemented to BI because agile addresses many problems found in BI projects.
Spikes in agile development for data warehousing	Rahman, Rutz, and Akhter (2013)	To follow agile approaches on data warehousing projects and incorporate all the changes without changing the basic architecture of the software system	Agile methodologies are best known for identifying the inefficient areas for each phase in a data warehousing project. Finding different ways for reducing redundancy, wasted time, and inefficiency can be best serviced with system metrics development.
Spikes in agile development for computer science education	Woodward, Montgomery, Vasa, and Cain (2013)	To take a potential benefit for computer science education by incorporating the techniques of agile development	Spikes are not suitable for fresher students due to the overhead of compound skills at that early stage.
	Bergin, Kussmaul, Reichlmayr, Caristi, and Pollice (2005)	To emphasise the formal processes and detailed documentation that are linked with compliance for computer science education	For application of agile in computer science education, students need to be introduced to agile practices in the right way while preparing them for all kinds of projects.
Spikes in agile development for blockchain	Lenarduzzi, Lunesu, Marchesi, and Tonelli (2018)	To incorporate the potential advantage of the strengths of a blockchain to augment the vulnerability of the Agile/Lean approaches	Agile blockchain might be a good way to record the workflow and to track the enhancements of the product under work as well as the productivity of developers by using Smart Contracts as a payment support.
Spikes in agile development for UX design	Da Silva, Silveira, Maurer, and Hellmann (2012)	To establish a framework for incorporating agile and user experience.	The framework proposed aims at addressing different aspects of this integration, providing alternatives to the UX designer inserted in the agile context.
Spikes in agile development for cloud computing	Younas, Jawawi, Ghani, Fries, and Kazmi (2018)	To ascertain the methods used in a cloud computing platform that are appropriate for agile development using systematic literature review.	Of the studies in the SLR, the techniques using existing tools were reported in 35%, simulations in 20%, and applications developed in 15%.
	Kalem, Donko, and Boskovic (2013)	To illustrate the association between agile methods for software development with the cloud computing platform.	Software development with agile methods is compared with software development with agile methods using cloud computing. All advantages of the second approach are pointed out.
Spikes in agile development for IoT	Cheng, Zhao, Niu, and Chen (2018)	To showcase service communication of agile IoT and orchestration platform using an event-driven service-oriented architecture (SOA) paradigm	The demo shows that the IoT service communication and orchestration platform responds quickly to the dynamic changes in the physical world.
Spikes in agile development for security implementation	Rindell, K., Hyrynsalmi, S. and Leppänen, V.(2017)	To understand benefits and drawbacks of using agile software methodologies in security sensitive development environments.	In order to reduce on overhead costs and uncertainties during agile software development, proper security engineering planning, mechanisms and measures should be put in place and incorporated with various methodologies best suited for implementation, in order to assist with software development and provide a robust and secure end product.
	Siponen, Baskerville, and Kuivalainen (2015)	To incorporate automated techniques to ensure that secure programming practices are implemented to ease the burden by building efficient, effective, and secure systems	Although several issues of integrating security into agile are solved, those methods have many limitations. The combination of related methods can eliminate some weaknesses and improve the existing methods.
	Baca and Carlsson (2011)	To implement Microsoft SDL procedures to reduce the drawbacks associated with agile development	For a reasonable security practice, an organisation should implement an integrated approach to all processes, including agile development processes.
Spikes in agile development for testing	Hooda and Chhillar (2015)	Quality assurance of software applications by carrying out particular test techniques and optimising the processes of software testing.	Most software failures happen due to a lack of security and performance testing. Therefore, a proposed right mix of testing (functional, performance, and security) can be applied for better software quality, where the spike considers one of performance testing.
	Hellmann, Sharma, Ferreira, and Maurer (2012)	To integrate agile development testing in the SDLC	An analysis of the tools used for agile testing showed a focus towards unit and acceptance testing tools.

The use of cloud computing in organisations has brought changes in the way these organisations run their applications and store data. Cloud computing helps organisations manage large volumes of data and provide prompt responses to their users. Integrating agile spikes into cloud computing may provide a solution to the problems of quantifying risks and timing uncertainty [22].

G. Agile Development in Internet of Things (IoT)

In Internet of Things devices, customer responsiveness is considered the main factor, but with fast-growing technology, the responsiveness to the customers' responsiveness is considered the end solution for all associations. The changing requirements are potentially supported by agile approaches. However, the alignment of the agile framework is relevant to the value stream of the agile process [11]. Additionally, continuous improvement and the sustainability of practices are the major inputs needed for scaling through regulations; competition; and volatility, uncertainty, complexity, and ambiguity [1]. Furthermore, it is important to consider the involvement of spikes in agile development as a factor that can benefit the merge of Internet of Things during software development. Although agile development aims to deliver a high-quality product rapidly, the involvement of spikes may offer a wide possibility to reduce the risk in an agile development environment for internet of things.

IV. Quality in Agile software development

The agile processes are best known for delivering results in short time intervals; however, it is important to ensure that they are meeting the quality requirements of the particular project as well. Quality is important when it comes to software development because it determines customer satisfaction. The same is true in agile software development where the spike is incorporated in the fulfilment of some fundamental components involved in unit testing. For instance, testing may determine whether the assembling life cycle of an agile environment and its supportive components are complete or whether the incorporated techniques will provide good quality.

A. Quality Assurance for Agile Software Development

Quality assurance also involves the management of future risk, and the incorporation of spikes may create a platform on which to test the riskiness of software usage in the future. Two major characteristics of agile development are its ability to handle unstable requirements that are executed throughout the development life cycle and its ability to deliver a product in defined budget constraints and shorter time frames. Agility is related to strategy, release iteration, and continuous and daily working software. Agile development allows users to develop software products in small increments or releases, which are then approved by the customer. The spike solution is the operational prototype [36].

Software quality assurance is also used to govern the processes for building the desired quality into software products. Quality assurance is divided into two main types: dynamic and static [18]. The organisation, objectives, and selection of the specific technique depending on the nature and requirements of the software project. Additionally, the selection of these methodologies depends on the project criteria. The static technique includes the examination of project documentation by groups or individuals via different tools, such as project inspection of the requirements and reviewing the code technically. However, the dynamic technique includes the execution of code and is generally used in agile development and processes [18].

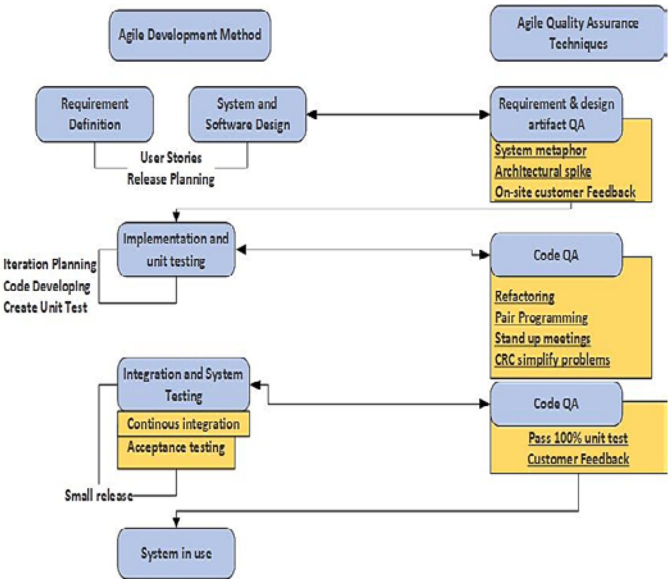

Figure 1. Quality Assurance in Agile Methods

In Figure 1, the generalised development life cycle approach of the agile methodology shows where some of the agile stages overlap one another. In agile development, some of the techniques have integrated both the quality assurance ability and agile functionality, which means the agile approaches have moved some of the quality assurance responsibilities to the developers.

In agile development, a small amount of output is sent for quality assurance in order to get fast feedback. The quality assurance practices and development practices integrate with one another to exchange the results quickly that helps in maintaining the processing speed. This approach enables two-way communication in agile development [18].

The architectural spike is a fixed variable/time scope PBI that is incorporated to inform the software team that more investigation is required for maximising velocity. The effective implementation of architectural spikes helps the software team get maximum estimates. The spikes are composed of a series of investigations that are created to find the solutions to maximum problems. The architectural spike technique is integrated to reduce the risks posed by XP [18][40].

B. Testing in Agile Software Development

In agile software development, testing is considered the cornerstone as most of the agile practices depend completely on effective software testing. The effectiveness and efficiency of the agile methodology help in determining agile software development outcomes. In agile development, the test plan is updated and written for every release. The agile test plan involves different testing types that are executed in a specific iteration, such as test results, test environments, infrastructure, and test data requirements. The general test plan in agile development includes the following: testing scope, new functionalities that need to be tested, types or levels of testing depending on complex features, performance and load testing, infrastructure consideration, risks or mitigation plan, resourcing, and milestones and deliverables [20].

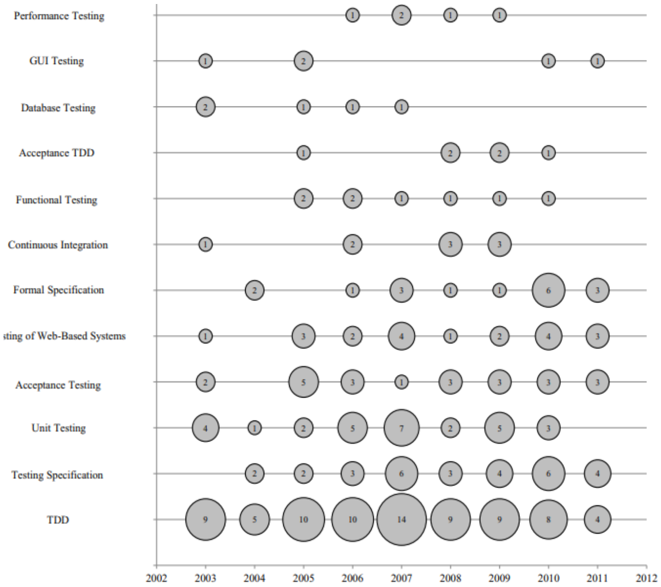

Figure 2. Testing Use over Time

Figure 2 provides a better understanding of how agile testing techniques are used over time. Interest in Test-Driven Development (TDD) continued to increase, highlighting its central role in agile testing. However, on the other side, the other agile testing use represents the “distinct spikes”. For instance, there are gaps in performance, publication database record, acceptance, and graphical user interface (GUI) testing. This specific point highlights the subfields with respect to more distinct agile testing fields. For example, in agile development, the GUI testing was not included from 2006 to 2009; however, database testing interest was shown in the same time period [16].

Although TDD has been the most used testing method in recent years, there are other forms of agile testing that are regarded as more efficient by software developers; many developers suggest that TDD does not lead to an increase in quality or faster development [48][50]. Some of the new techniques include Acceptance Test-Driven Development (ATDD) and Behaviour-Driven Development (BDD). Even though these three agile testing methods are related, ATDD is primarily a communication tool for the three amigos (customer, developer and tester) to ensure that all requirements are properly defined. Unlike TDD, ATDD does not require test automation. Often, the tests that are used in TDD are derived from ATDD, and while the ATDD test should be understood by the customer, the TDD test does not need to be readable by the customer. The BDD goes a step further by combining practices found in TDD and ATDD. In BDD, tests are initially written but focused on describing behaviour rather than tests used in TDD to test a unit of implementation [47]. Exploratory testing is another agile testing method in which the test design and test execution phases occur simultaneously, while in session-based testing, although similar to exploratory testing, the software is tested comprehensively and in a more orderly fashion [48]. Furthermore, TDD has continued to mature over the years in its use and acceptance by software developers because writing the test first allows you really understand what you want to code, receive faster feedback, reduces the time spent on rework and debugging. TDD leads to a greater understanding of the software being developed. The rise of other testing methods, albeit similar to TDD, was hinged on the preference of developers for writing tests after the code was already written. This preference is reflected in Test Last Development (TLD) and Iterative Test Last (ITL) [47] [49]. The Incremental Test-last Development (ITLD) is another closely related process to TDD. Both testing methods only differ in the order of activities that are involved in each increment and follow the same iterative steps, such as decomposing the specifications into smaller programming tasks, testing, coding and refactoring. However, while TDD requires the test to be written before the code, ITLD goes for writing the code first before the test. When TDD and ITLD were compared via a simple greenfield task, the difference was not significant. It was concluded that TDD did not appear to improve external quality. Furthermore, it was discovered that when TDD was used for simple tasks it was more productive than ITLD. The productivity of TDD significantly dropped when applied to more complex brownfield tasks [50].

V. AGILE IT SECURITY IMPLEMENTATION

In all software products, there are various potential vulnerabilities that can cause a lot of damage. Therefore, software developers are required to develop more efficient and secure systems by conducting all the phases of the software development life cycle. Developers must consider and incorporate all the security aspects in each phase to ensure that there is no vulnerability. The security is considered the most important component when developing any software product. Agile IT security goes beyond safeguarding the software as it also focuses on efficiency in performing tasks. This is facilitated by involving spikes, for instance, which boosts the testing process to abolish uncertainty and inconsistencies, boosts the software life cycle and ability to function efficiently, and thus sorts security issues that might arise from unresolved complexities. Furthermore, the involvement of spikes covers issues or challenges that may arise as a result of software developers lacking knowledge regarding futuristic implications from a particular feature. Lack of knowledge about the implications of feature can leave developers unaware of which areas they need to cover to secure the software. The involvement of spikes covers this uncertainty, giving a boost to agile IT security [25].

To reduce the drawbacks associated with agile development, five-step models are considered the best solution. In the first phase, all the security activities are extracted from the existing guidelines and processes, and the agile activities are defined to measure their level of agility. The integration problems and security issues related to agile activities are handled by implementing algorithm techniques and strategies. This way, the agility reduction tolerance parameter and its optimum value are taken into consideration to ensure the system's security. It is important to include the latest security advancements in all the stages of the Software Development Life Cycle (SDLC) to develop an efficient software system. The Comprehensive Lightweight Application Security Process (CLASP) and the Microsoft Security Development Lifecycle (SDL) are considered the two main processes for improving and enhancing a software product's security. However, some disadvantages and advantages have been compared and analysed in order to identify potential vulnerable areas. With respect to the agile methodologies, the Microsoft SDL procedures are implemented [4].

VI. Conclusion and Future Work

This paper provides a critical review of the use of spikes in various agile software projects. Spikes has also been illustrated as playing a critical role in agile software development by minimising unforeseen risks and uncertainties in the development cycle. This can be achieved by getting the agile team to understand the risks involved in the software development cycle and finding a viable solution that guarantees the best output in the form of software. However, there were challenges experienced in gathering relevant data because there is no sufficient information to back up the quality and effectiveness of the software produced related to spikes. This has been warranted by the limited information available for study, more research and experiments need to be carried out to ascertain the nature and effectiveness of software produced since there is varying information.

Professionals and industry partners will be recruited to assist in the use of spikes in agile software development because they will play a critical part in the entire process. In this regard, some case studies will be used for authentication purposes and also further research employed in order to clearly showcase the spikes in agile software development in the future.

References

[1] M. Akem, "Agile and The Internet of Things (IOT)" [Online]. Available: https://www.projecttimes.com/articles/agile-and-the-internet-of-things-iot.html. [Accessed: 18-AUG-2019].

[2] A. Albadarneh, I. Albadarneh, and A. Qusef, "Risk Management in Agile Software Development: A Comparative Study," In 2015 IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT), pp. 1-6. IEEE, 2015.

[3] T. Arndt, "Big Data and Software Engineering: Prospects for Mutual enrichment," Iran Journal of Computer Science., vol. 1, no. 1, pp. 3–10, 2018.

[4] D. Baca and B. Carlsson, "Agile Development with Security Engineering Activities," In Proceedings of the 2011 International Conference on Software and Systems Process, pp. 149-158. ACM, 2011.

[5] K. Beck, et al, "Manifesto for agile software development. 2001," [Online]. Available: http://www.agilemanifesto.org/. [Accessed: 12-SEP-2019].

[6] K. Beck and E. Gamma, *Extreme Programming Explained: Embrace Change*. Addison-Wesley Professional, 2000.

[7] J. Bergin, C. Kussmaul, T. Reichlmayr, J. Caristi and G. Pollice, "Agile Development in Computer Science Education: Practices and Prognosis," Proceedings of the 36th SIGCSE Technical Symposium on Computer Science Education, pp.130-131. ACM, 2005.

[8] BIGDSE '16, "Proceedings of the 2nd International Workshop on BIG Data Software Engineering," [Online]. Available: https://sse.uni-due.de/bigdse16/. [Accessed: 08-AUG-2019].

[9] D. Brown, *Agile User Experience Design: A Practitioner's Guide to Making it Work*. Newnes, 2012.

[10] B. Boehm, "Risk Management in Agile Software Development: A Comparative Study," IEEE Software, vol. 8, no. 1, pp. 32 - 41, 1991.

[11] B. Cheng, S. Zhao, M. Niu and J. Chen, "Agile IoT Service Communication and Orchestration Platform Using Event Driven SOA Paradigm," In IEEE INFOCOM 2018-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), pp. 1-2. IEEE, 2018.

[12] H-M. Chen, R. Kazman, S. Haziyev, "Agile Big Data Analytics Development: An Architecture-Centric Approach," In 2016 49th Hawaii International Conference on System Sciences (HICSS), pp. 5378-5387. IEEE, 2016.

[13] TS. Da Silva, MS. Silveira, F. Maurer and T. Hellmann, "User Experience Design and Agile Development: From Theory to Practice," Journal of Software Engineering and Applications, vol. 5, no. 10, pp. 743-751, 2012.

[14] A. Fuqua 2016, "What's A Spike, Who Should Enter it, And How To Word it?" [Online]. Available: https://www.leadingagile.com/2016/09/whats-a-spike-who-should-enter-it-how-to-word-it/. [Accessed: 23-SEP-2019].

[15] N. Grady, J. Payne and H. Parker, "Agile Big Data Analytics: AnalyticsOps for Data Science," In 2017 IEEE International Conference on Big Data (Big Data), pp. 2331-2339. IEEE, 2017.

[16] T. Hellmann, A. Sharma, J. Ferreira, and F. Maurer, "Agile Testing: Past, Present and Future," In Agile Conference (AGILE), pp. 55-63. IEEE, 2012.

[17] R. Hoda, J. Noble and S. Marshall, "Self-Organizing Roles on Agile Software Development Teams," In IEEE Transactions on Software Engineering, vol. 39, no. 3, pp. 422-444, 2012.

[18] M. Huo, J. Verner, L. Zhu, M. Babar , "Software quality and agile methods," In Proceedings of the 28th Annual International Computer Software and Applications Conference (COMPSAC), pp. 520-525. IEEE, 2004.

[19] Ibba 2019, "Agile methodologies and blockchain development," [online]. Available: https://iris.unica.it/handle/11584/260671#.XQx52RZKjIU. [Accessed: 29-AUG-2019].

[20] M. Isaacs, "What is Agile Testing? Process, Strategy, Test Plan, Life Cycle Example," [Online]. Available: https://www.guru99.com/agile-testing-a-beginner-s-guide.html. [Accessed: 12-AUG-2019].

[21] N. Jain and S. Dubey, "Agile Development Methodology with cloud computing," International Journal of Engineering and Computer Science, vol. 3, no. 4, pp. 5373-5378, 2014.

[22] S. Kalem, D. Donko and D. Boskovic, "Agile methods for cloud computing," In 2013 36th International Convention on

Information and Communication Technology, Electronics and Microelectronics (MIPRO), pp. 1079-1083. IEEE, 2013.

[23] R. Knaster and D. Leffingwell, *SAFe 4.5 Distilled: Applying the Scaled Agile Framework for Lean Software and Systems Engineering*. Addison-Wesley Professional, 2018.

[24] D. Larson and V. Chang, “A review and future direction of agile, business intelligence, analytics and data science,” International Journal of Information Management, vol. 36, no. 5, pp. 700-710, 2016.

[25] J. Laskowski, *Agile IT security implementation methodology*. Packt Publishing Ltd, 2011.

[26] D. Leffingwell, *Agile software requirements: lean requirements practices for teams, programs, and the enterprise*. Addison-Wesley Professional, 2010.

[27] V. Lenarduzzi, M. Lunesu, M. Marchesi and R. Tonelli, “Blockchain applications for Agile methodologies,” In Proceedings of the 19th International Conference on Agile Software Development: Companion, p. 30. ACM, 2018.

[28] G. Matharu, A. Mishra, H. Singh and R. Tonelli, “Empirical study of agile software development methodologies: A comparative analysis,” ACM SIGSOFT Software Engineering Notes, vol. 40, no. 1, pp. 1-6, 2015.

[29] A. Moniruzzaman and S. Hossain “Comparative study on agile software development methodologies,” Global Journal of Computer Science and Technology, vol. 13, no. 7, pp. 1-25, 2013.

[30] A. Moran, “Agile Big Data Analytics: AnalyticsOps for Data Science,” In Agile Risk Management, pp. 33-60. Springer, Cham, 2014.

[31] I. Perera, “Impact of using agile practice for student software projects in computer science education,” International Journal of Education and Development using ICT 5, vol. 5, no. 3, pp. 85-100, 2015.

[32] N. Rahman, D. Rutz and S. Akhter, “Agile development in data warehousing,” In Principles and Applications of Business Intelligence Research, pp. 286-300. IGI Global, 2013.

[33] K. Rindell, S. Hyrynsalmi and V Leppänen, “Case Study of Agile Security Engineering: Building Identity Management for a Government Agency,” International Journal of Secure Software Engineering (IJSSE), vol. 8, no. 1, pp. 43-57, 2017.

[34] K. Schwaber and M. Beedle, *Agile software development with Scrum*, vol. 1. Upper Saddle River: Prentice Hall, 2002.

[35] M. Siponen, R. Baskerville and T. Kuivalainen, “Integrating security into agile development methods,” In Proceedings of the 38th Annual Hawaii International Conference on System Sciences, pp. 185a-185a. IEEE, 2005.

[36] I. Stamelos and P. Sfetsos, *Agile software development quality assurance*. Igi Global, 2007.

[37] Tanner, “Spikes Purpose and Usage,” [Online]. Available: https://www.spikesandstories.com/spikes-purpose-and-usage/. [Accessed: 15-SEP-2019].

[38] C. Woodward, J. Montgomery, R. Vasa and A. Cain, "Agile development spikes applied to computer science education," In Proceedings of 2013 IEEE International Conference on Teaching, Assessment and Learning for Engineering (TALE), pp. 699-704. IEEE, 2013.

[39] M. Younas, I. Ghani, D. Jawawi and M. Khan, "A framework for agile development in cloud computing environment," Journal of Internet Computing and Services, vol. 17, no. 5, pp. 67-74, 2016.

[40] S. Ambler, *Agile modeling: effective practices for extreme programming and the unified process*. John Wiley & Sons, 2002.

[41] A. Solinski and K. Petersen, "Prioritizing agile benefits and limitations in relation to practice usage," software quality journal, vol. 24, no.2. pp.447-482, 2016.

[42] K. Schwaber and J. Sutherland, “Scrum Guide,” [Online]. Available: https://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-US.pdf. [Accessed: 30-SEP-2019].

[43] Eric, “How to use (and difference of) Spike vs Proof of Concept vs MVP,” [Online]. Available: https://www.agileview.com.au/how-to-use-and-difference-of-spike-vs-proof-of-concept-vs-mvp/. [Accessed: 30-SEP-2019].

[44] L. Garland, "Spikes, POCs, Prototypes and the MVP," Medium, 2014. [Online]. Available: https://medium.com/studio-zero/spikes-pocs-prototypes-and-the-mvp-5cdffa1b7367. [Accessed: 03- OCT- 2019].

[45] The Interaction Design Foundation, "A Simple Introduction to Lean UX," 2019. [Online]. Available: https://www.interaction-design.org/literature/article/a-simple-introduction-to-lean-ux. [Accessed: 02- OCT- 2019].

[46] C. Schmidt, T. Kude, A. Heinzl and S. Mithas, "How Agile practices influence the performance of software development teams: The role of shared mental models and backup," Association for Information systems. 2014.

[47] L. Alberto, "Analysis of the impact of test-based development techniques (TDD, BDD, AND ATDD) to the software life cycle," [Online]. Available: https://iconline.ipleiria.pt/bitstream/10400.8/3699/1/Dissertation_2160085_LuisGomez.pdf. [Accessed: 04- OCT- 2019].

[48] L. Cisneros, C. Reis and M. Maximiano, "An Experimental Evaluation of ITL, TDD and BDD," In ICSEA 2018, The Thirteenth International Conference on Software Engineering Advances (pp. 20-24). ICSEA,2018

[49] A. Čaušević, D. Sundmark and S. Punnekkat, "Impact of test design technique knowledge on test-driven development: A controlled experiment" In International Conference on Agile Software Development, pp. 138-152. Springer, 2012.

[50] A. Tosun, et al, "An industry experiment on the effects of test-driven development on external quality and productivity," Empirical Software Engineering, vol. 22, no. 6, pp. 2763-2805, 2017.

[51] I. Hooda and R. Chhillar “Software test process, testing types and techniques,” International Journal of Computer Applications, vol. 111, no. 13, pp. 10-14, 2015.