Handling Non-Functional Requirements For Big Data and IOT Projects in Scrum

Vaibhav Sachdeva and Lawrence Chung

Department of Computer Science
The University of Texas at Dallas
Richardson, TX 75080, USA

{vaibhav.sachdeva, chung}@utdallas.edu

Abstract – Agile methodologies are gaining popularity at a lightning pace in the software industry. Prominent agile methodologies, such as Scrum and Extreme Programming (XP), help quickly deliver a system that meets Functional Requirements (FRs), while adapting to the changes in the customer requirements and feedback. However, Non-Functional Requirements (NFRs) by and large have been either mostly ignored or introduced and retrofit late in the software development cycle, oftentimes leading to project failures. With the recent, increasing shifts towards cloud and emphasis in big data in the software industry, NFRs, such as security and performance, have become more critical than ever before. This research proposes a novel approach to handling security and performance NFRs for projects involving big data and cloud, using Scrum. An industrial case study, conducted over a span of 9 months, shows that the approach helps deal with security and performance requirements individually as well as conflicts between them in an agile methodology.

Index Terms – Big Data, Internet of Things (IoT), Agile Methodologies, NFRs, Scrum.

I. Introduction

IoT, a term coined by Kevin Ashton at the Auto-ID research group at MIT in 1999 [1], is an emerging field, transforming the software industry. IoT is defined as the ability of a variety of components to connect and communicate with each other using cloud computing. Rapid advancements in cloud and IoT have led to inventions, such as smart cars [3], and smart homes [2].

Fig. 1 shows an example of IoT and how it connects the world.

IoT requires storing, modeling, querying, and processing an enormous amount of (both structured and) unstructured data termed as ‘Big Data’ [4]. Big data is described by the following characteristics: Volume, Variety, Veracity, and Velocity (4 V’s) [5] and is an integral part of IoT. There are various big data processing frameworks available, including Hadoop, which is an open-source project developed for reliable, scalable, distributed computing and storage, and has been widely adopted to support data intensive distributed applications [6].

Agile methodologies [7] have revolutionized the software industry and have provided frameworks, such as Scrum and Extreme Programming, to allow rapid delivery of working software to the customers [8][9]. For the scope of this paper, we will concentrate on scrum [10], which is one of the most common agile frameworks for developing and managing complex projects and is based on transparency, inspection, and adaption [11]. Scrum uses user stories to capture user requirements [12]. User stories portray high-level requirements from an end-user point of view and deliver incremental business value. “As a user, I want to be able to pay my credit card bill online, so that I do not incur any charge” is an example of a user story.

However, in order to reduce time to market, user stories are focused more on execution rather than requirements engineering. User stories lay more emphasis on FRs, while NFRs are either ignored, or introduced late in software development life cycle, hence leading to project failures [13][14]. As one of the agile manifesto [15] value states, “Responding to change over following a plan”, while it is important to have the ability to change, it is still important to have a good plan to start with and identify major risks at an early stage in order to minimize cost and effort.

NFRs become even more vital to projects involving cloud and big data, and therefore need to be handled in a suitable manner. Every now and then, we hear about data breaches, security hacks, and data leaks in the cloud. Industrial shift towards cloud automatically makes security more important than ever before. Customer protection becomes an integral part of customer satisfaction. In addition, big data processing needs to adhere to performance standards acceptable to the end users. Therefore, security and performance – the scope of this paper - are critical for many software projects residing on cloud and involving big data and hence need to be handled in a well-defined manner.

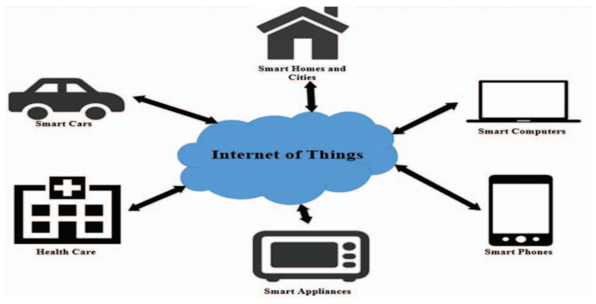

Fig. 1 Internet of Things

Some previous proposals, such as the NFR Framework [16], NORMATIC [17] and AFFINE [18], can be used for modeling NFRs. However, none of these addresses how to deal with NFRs in an agile environment for complex projects involving cloud and big data, for example, how and when inputs from the stakeholders can be obtained.

The paper is organized as follows. Section II describes a use case diagram for a banking system. Sections III and IV respectively provide an agile method for handling security and performance for the banking system described in section II. Section V presents an industrial case study describing the current issues when dealing with NFRs for cloud and big data projects using Scrum, how the solution presented in the paper mitigated the issues, and threats to validity. Section VI addresses conflicts between NFRs and the importance of maintaining the right balance. Conclusions are drawn in Section VII followed by future work involved.

II. USE CASE DIAGRAM

Let us consider an online banking system as shown in Fig. 2. Let us assume that the bank has more than one million users spread across the globe and all the data for this bank resides on a cloud. For the scope of this paper, let us focus on FR1: User Login, FR2: Online Transaction, NFR1: Security, and NFR2: Performance. Login FR will involve dealing with millions of records in the database. Online Transaction FR requires immense security measures to avoid any breaches. Performance and security are a big concern for any bank as all the data involved is highly confidential and performance lags can result in unsatisfied customers and transaction delays.

III. SECURITY NON-FUNCTIONAL REQUIREMENT

Let us consider NFR1: - The online banking system should be secure. 'Secure' is an ambiguous term in itself and needs to be defined with stakeholder's involvement. As agile focuses on "Individual and Interactions" over "Process and Tools", it is more important to have a common understanding between the stakeholders as to what 'Secure' means with respect to the particular project rather than trying to define the term by itself. In scrum, this NFR is likely to be ignored or introduced late or made a part of the Acceptance Criteria (AC) of some other user story. AC is the way to measure the success of a user story and generally consists of set of user scenarios. The solution as described in this paper is to map security NFR to a feature, a collection of similar user stories, and introduce it in the beginning phase of software development. During refinement meetings, scrum teams will work with the product owners and other stakeholders to break down the features into user stories. For example, a user story depicting the NFR will be, "As an admin of the online reservation system, I want to limit the access of the banking system to users with valid user names and passwords, so that unauthorized users cannot access the data." The AC of the user story will look like: AC1- Users with valid usernames and passwords should be allowed to login. AC2- Users with invalid username password combination should be redirected to login screen. AC3- Users after successful login should be directed to home page.

Along with the NFR to user story mapping, it is also very important that the user story is prioritized in the backlog early so that the system is designed accordingly. Sprint 0 can also be used for design discussions. Introducing the user stories at a later stage will involve a lot of design changes, which will not be cost and time effective.

One big problem with NFRs is that they are treated as soft goals and thereby there is no clear way of defining whether the NFR is met or not. This can be avoided in scrum using AC. Similar to any other user stories, NFR user stories should also have a clearly defined AC and the user story should only be accepted by the product owner once all the ACs are met. Once the user story is prioritized high in the product backlog and is made available to pick by the product owner in the sprint-planning meeting, the scrum team should pick up the user stories based on their capacity and start breaking them down into executable tasks. As user stories are conversation starters, it is essential that scrum teams working on the user story work with the product owner throughout the sprint in order to clarify the requirements and AC. Once the user story is complete, the product owner should verify the AC for each user story and go through Definition of Done (DoD) to accept the user stories. DoD is a checklist that helps verify the completeness of the user story. If there are any concerns found, either the product owner should create a new user story that is again prioritized in the product backlog and made available in a later sprint, or the scrum team should work on fixing the issues in the current sprint.

Security impacts can be introduced with any subsequent user stories as well and therefore it is vital that such impacts are monitored with each user story and appropriate actions are taken wherever necessary. Fig. 3 shows the process diagram for the same.

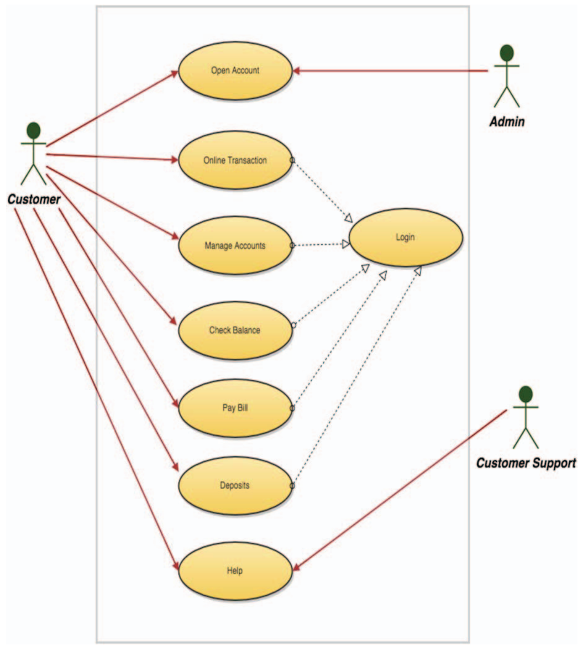

Fig. 2 Online banking system use case diagram

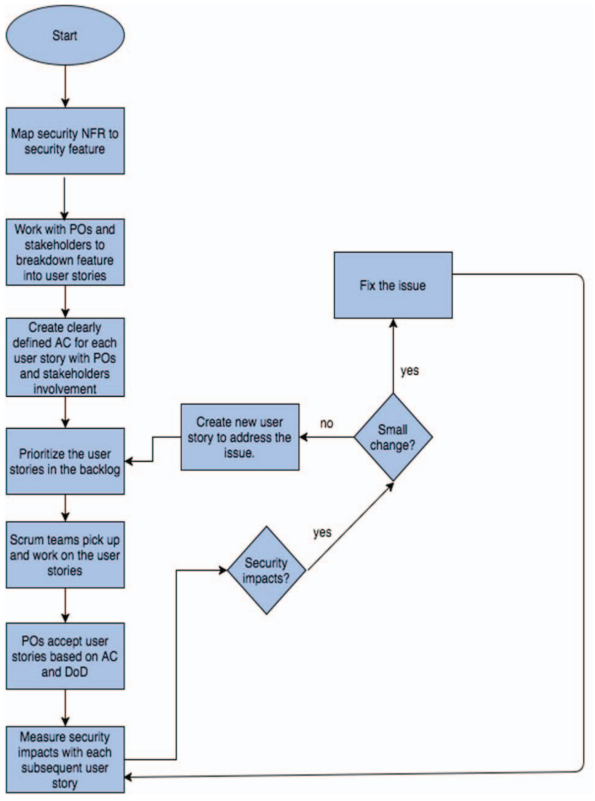
Start
Map security NFR to security feature
Work with POs and stakeholders to breakdown feature into user stories
Create clearly defined AC for each user story with POs and stakeholders involvement
Prioritize the user stories in the backlog
Scrum teams pick up and work on the user stories
POs accept user stories based on AC and DoD
Measure security impacts with each subsequent user story
Security impacts?
yes
Small change?
no
Create new user story to address the issue.
yes
Fix the issue

Fig. 3 Security NFR process diagram

IV. Performance Non-Functional Requirement

The biggest problem with performance NFR is that it is treated as a qualitative measure rather than a quantitative one. Let us consider our system in Fig. 1 and FR2: Online Transaction. We assumed that the data resides in a cloud and there are millions of records in the system. Let us assume user1 wants to transfer money to user2. User1 tries to search for user2 using email-id. On hitting the search key user1 has to wait 5-7 minutes before user2 is found. Such a system is unlikely to be used by users and is not an ideal scenario for any organization as focus is on customer satisfaction. Also saying that the search should be quick enough is not a good measure as definition of 'quick' varies from user to user. A search taking 45 seconds might be quick for one user and way too slow for another one. If the search time keeps on increasing with the addition of new FRs or additional data records (noticeable or not noticeable), such a system will not be scalable and will be difficult to maintain in the long run. Also, ignoring performance NFR in the initial design of the system and relying on the emerging architecture principle will cause additional scalability issues. Consider that the lookup system was designed with no indexing and the database ended up having millions of records, search on such a system will be very slow and trying to add indexing at that stage will require way too much cost and effort.

Therefore, it is important to consider performance constraints from the very beginning and monitor them on a regular basis. An efficient way to do this in scrum is to start with a spike to come up with initial performance constraints. A spike is an investigational user story that is time boxed to understand and scope the work involved. Performance should be quantified and measured with each user story i.e. each user story should have performance impact as an AC and this should be measured using automated test suites. The outcome of the initial spike should act as the performance threshold for all related user stories. Performance should be made a part of DoD for user stories as well as DoD for release. For e.g. in the system defined above, if the spike identifies performance threshold as 4.5 seconds, then all the user stories related to search capability for FR2 will have an AC constraining the performance to 4.5 seconds.

If at any point performance falls below the defined threshold, the code that caused the failure should be tracked down and fixed within the same sprint or the following sprint based on the discussion with the product owner. If the impact is too huge, a user story should be created to investigate the root cause and fix the issue. The user story goes back into the product backlog where the product owner prioritizes it and makes it available to pick in the sprint-planning meeting. As working on cloud-based projects requires continuous integration, appropriate metrics should be in place to measure performance impacts. Fig. 4 shows the process diagram.

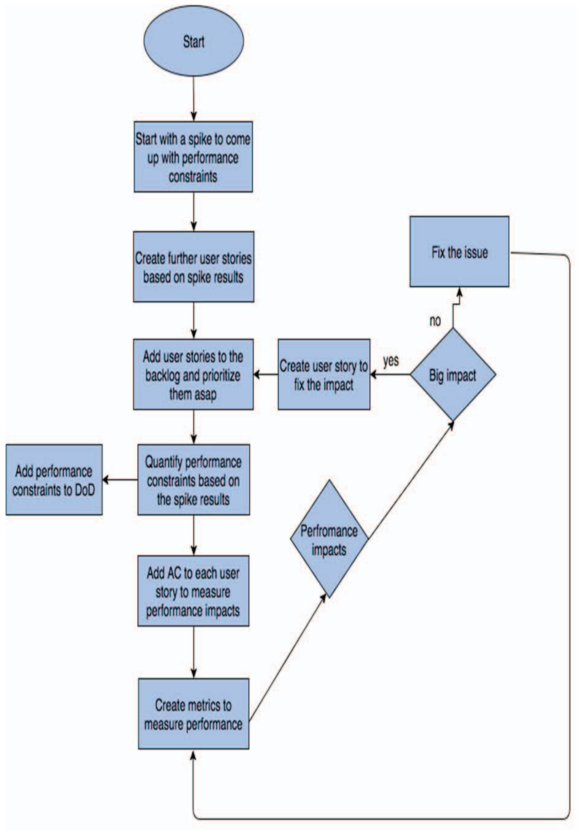
Start
Start with a spike to come up with performance constraints
Create further user stories based on spike results
Add user stories to the backlog and prioritize them asap
Quantify performance constraints based on the spike results
Add performance constraints to DoD
Add AC to each user story to measure performance impacts
Create metrics to measure performance
Perfromance impacts
Big impact
yes
Create user story to fix the impact
no
Fix the issue

Fig. 4 Performance NFR process diagram

V. Industrial Case Study

In this section, a multinational IT organization's cloud and big data related software projects are being presented to analyze the issues faced with security and performance and how the proposed solution helped mitigate the issues. The case study involves 2 projects. Project 3rd Gen Phones talks about the issues faced when NFRs are not handled in a proper manner and the impact on the project and business as a result of that. Project New Gen Phones uses the proposed solution to handle NFRs and verifies the solutions validity and positive impact on business and revenue.

A. *Project 3rd Gen Phones*

Project 3rd Gen Phones started at a time when the organization had already been using scrum for almost 2.5 years. The project was aimed at developing Voice over IP (VoIP) video conferencing phones and involved 6 scrum teams, 2 scrum masters, and 3 product owners. The phones registered to a call manager that resided on an internal cloud using MongoDB. As the company has big reputation in the market, customer satisfaction and security were the biggest goals.

The scrum teams decided to use Android platform for the 3rd gen phones and started with an emerging architecture focusing on delivering a good quality product at a rapid pace. The product was doing really well and the goal was to ship its minimum viable product (MVP) within six months. The list of features kept on increasing and the phone supported more than 35 features by the end of 5 months. Little attention was paid to performance, and security was handled implicitly and minimal work was anticipated due to androids built in security features. At the end of 5 months, the teams had met the FRs for the MVP with only security compliance remaining before the product launch. The product was sent to the security lab for compliance testing and the results turned out to be a nightmare for the organization. 6 security issues were found in the product and they had to be fixed before the product could be released. The engineering team started analyzing the issues and realized that fixing the issues required a major architectural change and the work was estimated to be 3 months. This caused the organization a revenue loss of more than 4 million dollars as one of the big customers bought the product from a competitor. At the end of 8 months, the product was finally ready. It went through major architectural changes which were initially ignored, required 3 additional months, more than 30 resources, and loss of 6.5 million dollars to the organization.

Subsequently after the release, some users complained about corporate directory searches being too slow. On investigation, the architects realized that the performance was severely impacted due to the added functionalities. As performance wasn't quantified throughout software development life cycle, this went unnoticed. The only requirement from the product owner was that the performance should be 'good' and the teams complied and the product was released. 'Good' is a qualitative term and subject to multiple interpretations. The teams realized that in order to meet the performance standards, they first need to set those standards and 'good' is not 'good enough.' Architects sat down for several weeks and came up with a quantified performance standard. Now the bigger task was to meet those standards. It took another 2 months, 3 scrum teams, and thousands of dollars to meet the set performance standards, but the issue was finally fixed. However, maintaining the product became a big issue as the architecture was not scalable or reliable and had major performance impacts from time to time.

B. *Project New Gen Phones*

After retrospection of the issues identified with the 3rd Gen Phones project, the engineering team decided to switch platforms from android and release their new generation video conferencing phones on a different platform. The MVP was set and an effort of 8 months was estimated. This project utilized the solution described in this paper to handle security and performance. Appropriate time was spent upfront on the architecture of the system and security and performance were given prime importance. Security was explicitly called out as a feature and was prioritized high in the backlog. Feature F2341 Security Support was introduced to handle any security concerns. The feature was then broken down with the help of product owners and stakeholders into user stories. Some of the sample user stories created were:- US18512: Quick fix to glibc vulnerability reported in CVE-2015-7547, US18571: Run Codenomicon vulnerability testing for 11.5, and so on. Each of the user stories had clearly defined AC and was prioritized high in the backlog. Once the framework was put into place, security impacts were measured with each user story and fixed promptly. While working on one of the user stories, a big security impact was introduced and 'US20186: BE/Gumbo security hole analysis' was created to analyze the impact and fix it as appropriate. Security compliance testing was run at the end of each sprint to measure any impacts. Fig. 5 shows use of the proposed security NFR process diagram.

Performance thresholds were set upfront and measured using automated test beds on each code commit and the results were displayed on big dashboards everywhere in the organization. The team spent extra time in sprint 0 focusing on performance and security and coming up with an architecture that is scalable in a long term. The analysis started off by creating a spike user story US16758: SPIKE to investigate search performance threshold. The spike led to creation of six more user stories. Some samples are: US16789: Add tags to code to record time stamp for performance (Need to be done after F994), US16792: Worst case performance analysis, and so on. The threshold for search was decided as 2.47 seconds and this was made a part of DoD. Each user story under feature F991: Search improvements, had the threshold as an AC. US17568: Enable predictive search broke the threshold, but the product owners did not accept the user story, as it did not meet the DoD. The teams worked on fixing the performance issue in the following sprint and met the threshold and the user story was accepted after the AC was met. Fig. 6 shows use of the proposed performance NFR process diagram.

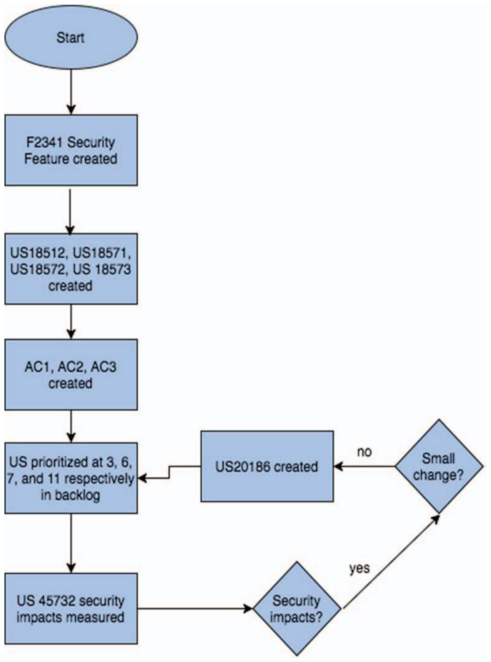
Start
F2341 Security Feature created
US18512, US18571, US18572, US 18573 created
AC1, AC2, AC3 created
US prioritized at 3, 6, 7, and 11 respectively in backlog
US20186 created
no
Small change?
yes
US 45732 security impacts measured
Security impacts?

Fig. 5 Security NFR process diagram in practice

The team was able to complete the MVP in 8.5 months and was sent to security compliance lab for final testing. The lab came back with 1 issue, which took less than 2 weeks to fix, and the product was shipped in 9 months. The project was 1 month late as per its initial schedule which led to a revenue loss of 0.25 million dollars. However, spending that extra month provided a long-term benefit to the organization. The product made total revenue of 47 million dollars. Supporting the product became painless and there were less than 10 customer issues reported. Even after the product was shipped, teams had dashboards measuring performance of the product being used by their customers to analyze the results and make enhancements in future. Table 1 analyzes the two projects and against a set of criteria's.

C. Threats to Validity

Even though Project Next Gen Phones, as described above, shows the success of the project while using the proposed solution to handle security and performance issues, the success might have been influenced by other factors as well like better project management, quality vs. quantity tradeoffs, customer requirements, and market needs. It is also possible that the security impacts might have been reduced in the project due to the existing code base that was previously tested for possible security compliance issues. The project utilized better data structures and search algorithms that adds to the uncertainty of the proposed solution. External factors like sustainable pace, team dynamics, workplace environment, organizational mindset, management pressure, and hardware capabilities can also add to the uncertainty of the hypothesis proposed in this paper. The solution needs to be applied to further industrial projects to confirm the validation of the proposed hypothesis.

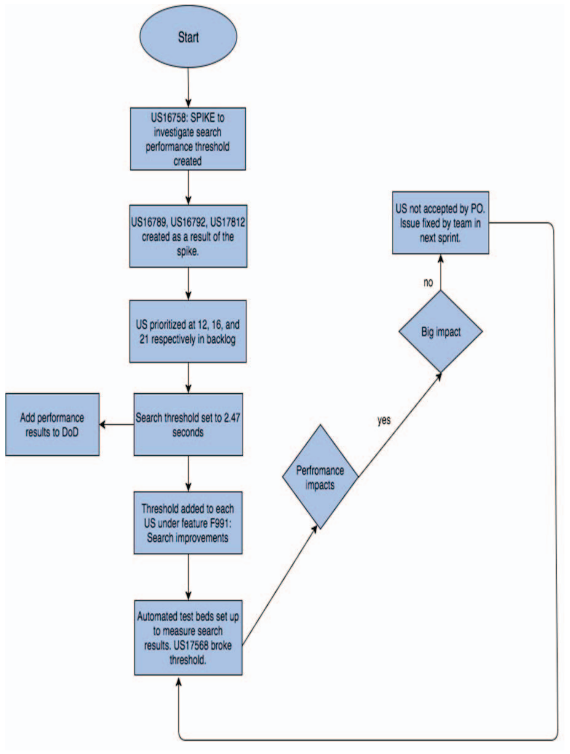
Start
US16758: SPIKE to investigate search performance threshold created
US16789, US16792, US17812 created as a result of the spike.
US prioritized at 12, 16, and 21 respectively in backlog
Search threshold set to 2.47 seconds
Add performance results to DoD
Threshold added to each US under feature F991: Search improvements
Automated test beds set up to measure search results. US17568 broke threshold.
Perfromance impacts
yes
Big impact
no
US not accepted by PO. Issue fixed by team in next sprint.

Fig. 6 Performance NFR process diagram in practice

VI. NON- FUNCTIONAL REQUIREMENTS CONFLICTS

As Barry Boehm mentioned, it is very important to find the right balance of quality attributes by identifying the conflicts between the attributes at an early stage and agree on a compromise in order to provide a win-win situation [19]. The two NFRs discussed in this paper conflict with each other and a balance needs to be maintained to ensure product success. For e.g. for the Next Gen Project, a security user story is 'US17964: As a user, I want the corporate directory search results to be visible only to logged in users, so that unauthorized users will not be able to see sensitive data.' Also lets consider a search user story 'US27415: As a user, I want to be able to search for people in corporate and local directories, so that I can contact them.'

TABLE I
PROJECT COMPARISONS

Criteria	Project 3rd Gen Phones	Project Next Gen Phones
Initial Scope	6 months	8 months
Feature	5 months	8.5 months
Fixing Issues	3 months	0.5 months
Final release	8 months	9 months
Revenue loss	6.5 million dollars	0.25 million dollars
Total Revenue	31.6 million dollars	47 million dollars

To meet these two requirements, a balance needs to be maintained between security constraints and performance threshold. The engineering team started off working on US27415 without knowing that US17964 existed. The data structures and algorithms designed were optimal for corporate and local directory search and the worst-case performance threshold identified was identified as 2.47 seconds. When US17964 was introduced, the teams had to change their algorithm designs to restrict corporate directory search to logged in users and local directory search was still global. This required effort, data structure and algorithm modifications, which in turn had performance impacts. The performance decreased to 2.89 seconds. To handle this conflict, the scrum teams at the end of the sprint created US43516: SPIKE Identify search threshold based on security concerns. The product owner prioritized this user story at the top of the backlog and was picked in the next sprint. The scrum team worked on this investigational user story and came up with a proposed threshold of 2.73 seconds as a compromise. The threshold was discussed with the other teams and the product owners and was agreed upon. The DoD was updated to reflect the threshold, and the automated test suites were updated as well. The future user stories would follow the new threshold and any impacts due to introduction of some new NFRs would be handled in a similar manner. These changes did require a lot of effort, time, and cost. However, if the conflicts were known/identified early, the compromise would have been achieved quicker and would have saved a lot of effort and time for the engineering team.

VII. Conclusion And Future Work

For projects involving cloud and big data, non-functional requirements (NFRs) have become way more crucial than ever before and therefore need to be handled in a well-defined manner. The case studies presented in the paper clearly illustrate the need to introduce NFRs, such as security and performance, early in the software lifecycle, whose lack of concerns can also severely damage other crucial NFRs, such as maintainability and scalability, later on. In scrum, security can be mapped to features and broken down into user stories with stakeholders' involvement. Clear acceptance criteria (AC) needs to be defined to allow measurement of security impacts. Performance needs to be quantified early and appropriate metrics and automated test beds need to be put into place to monitor the performance impacts with each user story. It is vital to call out security and performance in the definition of done (DoD) for the user stories as well as the release.

Future work will involve addressing other NFRs than security and performance. Application of our proposal to more industrial projects will help analyze the shortcomings towards improvement. Future work will also include handling process-related NFRs for cloud and big data projects in an agile environment.

Acknowledgment

The authors thank Riddhi Pai Raiker for her constant support and feedback in organizing the paper.

References

[1] Ashton, Kevin. "That 'internet of things' thing." *RFiD Journal* 22.7 (2009): 97-114.
[2] Aldrich, Frances K. "Smart homes: past, present and future." Inside the smart home. Springer London, 2003. 17-39.
[3] Speed, Chris, and Duncan Shingleton. "An internet of cars: connecting the flow of things to people, artefacts, environments and businesses." Proceedings of the 6th ACM workshop on Next generation mobile computing for dynamic personalised travel planning. ACM, 2012.
[4] Cuzzocrea, Alfredo, Domenico Saccà, and Jeffrey D. Ullman. "Big data: a research agenda." Proceedings of the 17th International Database Engineering & Applications Symposium. ACM, 2013.
[5] Beyer, Mark. "Gartner Says Solving'Big Data'Challenge Involves More Than Just Managing Volumes of Data." Gartner. Archived from the original on 10 (2011).
[6] Hadoop, Apache. "Hadoop." 2009-03-06]. http://hadoop. apache. org (2009).
[7] Highsmith, Jim, and Alistair Cockburn. "Agile software development: The business of innovation." *Computer* 34.9 (2001): 120-127.
[8] Abrahamsson, Pekka, et al. "New directions on agile methods: a comparative analysis." Software Engineering, 2003. Proceedings. 25th International Conference on. Ieee, 2003.
[9] Beck, Kent. "Embracing change with extreme programming." Computer 32.10 (1999): 70-77.
[10]Sutherland, Jeff, et al. "The scrum papers: Nuts, bolts, and origins of an agile process." (2007).
[11]K. Schwaber, "Agile Project Management with Scrum," Redmond, Washington, Microsoft Press, 2004.
[12]Cohn, Mike. User stories applied: For agile software development. Addison-Wesley Professional, 2004.
[13]Mead, Nancy R., Venkatesh Viswanathan, and Deepa Padmanabhan. "Incorporating security requirements engineering into the dynamic systems development method." 2008 32nd Annual IEEE International Computer Software and Applications Conference. IEEE, 2008.
[14]Paetsch, Frauke, Armin Eberlein, and Frank Maurer. "Requirements Engineering and Agile Software Development." WETICE. Vol. 3. 2003.
[15]Beck, Kent, et al. "Manifesto for agile software development." (2001).
[16]Mylopoulos, John, Lawrence Chung, and Brian Nixon. "Representing and using nonfunctional requirements: A process-oriented approach." IEEE Transactions on Software Eng. 18.6 (1992): 483-497.
[17]Farid, Weam M., and Frank J. Mitropoulos. "NORMATIC: A visual tool for modeling non-functional requirements in agile processes." Southeastcon, 2012 Proceedings of IEEE. IEEE, 2012.
[18]Bourimi, Mohamed, et al. "AFFINE for enforcing earlier consideration of NFRS and human factors when building socio-technical systems following agile methodologies." International Conference on Human-Centred Software Engineering. Springer Berlin Heidelberg, 2010.
[19]Boehm, Barry, and Hoh In. "Identifying quality-requirement conflicts." IEEE software 13.2 (1996): 25.