"Scrumbear" framework for solving traditional scrum model problems

Leanah Alsaber, Ebtesam Al Elsheikh, Sarah Aljumah, Nor Shahida Mohd Jamail
College of Computer and Information Sciences, Prince Sultan University, Saudi Arabia

Article Info

Article history:

Received Apr 8, 2020
Revised Jun 16, 2020
Accepted Jul 25, 2020

Keywords:

Agile
Framework
Scrum
Software development
Software engineering

ABSTRACT

Software engineering is a discipline that is little understood by people. It defines how software is developed and maintained to meet the clients' requirements. Software engineers follow certain systems and standards in their work to meet the clients' desires. It is on this background that engineers follow specific models in coming up with the final product. One of the models highly used is scrum, which is one of the agile methodologies. However, despite being highly used, it has inherent flaws that need to be corrected. Those flaws are product owner continues changing; do not accept changes in working scrum, sprint's release time limitation, finally wasting team time within each sprint. This paper presents a new framework, which is an updated version of the current Scrum, to overcome the scum models mentioned issues. In this study, a new framework is presented in a way that is understandable and needed by software developer's team upon the collected qualitative and quantitative data. The implementation was by making some changes to the current scrum model leading to the "Scrumbear", certain flaws can be corrected. One of the presented changes involve adding the control master rule to ensure controlling the requirements changing.

Corresponding Author:

Leanah Alsaber,
College of Computer and Information Sciences,
Prince Sultan University,
66833 Rafha Street, Riyadh 11586 Saudi Arabia.
Email: Leanah.Alsaber@gmail.com

1. INTRODUCTION

Technology rapidly changing and the customers start to have the instant needs of the products with no patient for waiting for a long process, which may take months and could end up with different requirements. According to [1], the customers testing on the provided version is to provide their inputs for enhancement and requirements accuracy. From here, the customer is permitted to demonstrate and reveal their actual needs. Therefore, it could be said that users and customers' needs become more craving to change than before. SE is an approach of engineering the process of system development life cycle (SDLC) [2]. SD has continued to improve the process for years and presented different methods to decrease the difficulties and problems [3]. As a result, to cover the needs of the customers continuous changes, SE start to adapt the concept of agility by having shorter development period and continuous new versions to fulfill the customer satisfactions [4]. There are many methods have been used by SE to carry out the products to customers. One of these methods is Agile, which is consisting of three main models Kanban, Extreme Programing, and Scrum. Scrum is an agile model that maintain rapid and continues implementation of requirement [5]. Moreover, upon practices and discussed statistics, scrum is more suitable for medium to small products business. Scrum team have some challenges to face as the continues requirements change by the product owner, no changes are accepted while the sprints is under development, sprints and release time limitations, and finally the unproductive time period spent by the team. In the interest of these issues, as software engineers in this paper we introduce a new proposed framework to overcome these issues by covering them by adding a new role as a control master, besides having number of sprints working at the same time to have on release for them at each time. This framework was implemented after analyzing the data from the qualitative and quantitative conducted techniques.

2. LITERATURE REVIEW

Software engineering SE is a methodical discipline/approach that is applied in the management of software lifecycle processes in addressing the issues of the low-quality software projects to produce high-quality software and software solutions. It is more about engineering in terms of analysis, design, development, and maintenance processes. Besides, SE helps in ensuring that the projects meet all prerequisite standards within the projected budgets and time plans. According to IEEE, SE defined as the process of applying a methodical, disciplined, and quantifiable approach in developing, operating, and maintaining software-the interplay between engineering and software. Computer systems need to be reliable and efficient; which is met through systematic software processes. SE has played a role in the creation of large, extensive and dynamic software systems that operate in a vast range of fields [6]. Because of SE, the software that is developed has grown tremendously since the time of introducing the process.

2.1. SE layers

To be able to understand SE, one needs to view and comprehend that it is a layered technology where each layer seeks the fulfillment of its predecessor show in Figure 1.

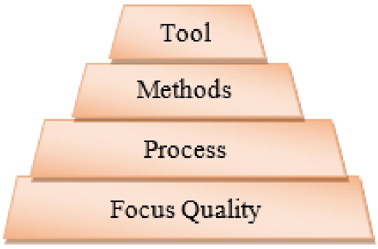

Figure 1. Software engineering layers

- Quality focus

The last layer is the quality layer; which is used as the stone base of all layers in SE. It is about the correctness of the requirement and how it is connected to the functionality, and how the software should be performed eventually. In addition, it is about the maintainability of the software and its integrity from the perspective of security amongst other features. Finally, it is about the usability and what it the necessitated effort to run the software.

- Process

After the quality layer, the process layer comes next, and it is considered as the foundation of SE. This layer defines the key processes that one has to focus critically on to ensure the effective delivery of the software, where it works as a glue to keep all the layers together. This layer enables the development of software in the designated time; it also has the definition of the SE framework as explained in the following sections.

- Methods

The Method layer mainly answers the questions of “How-to” do the processes, and it includes the SDLC that covers such tasks in the SDLC as in requirements, designing, implementation, testing and support. SDLC offers technical knowledge behind the development of software. It is considered crucial as it covers broad tasks such as in the SDLC [7].

- Tools

Finally, the last layer is a tool, which provides fully-/semi-automated support for its predecessors - the methods and the process layers. In this layer, the tools integrated in the need of passing the information, like when information is created by one, it could be used others. In the design position, the “Microsoft publisher” is counted as a tool in SE for the web designing.

2.2. Software process

This forms a sequence of functions that should be followed to develop software. There are numerous models of software, each of which consists of these following main activities:

- Requirements engineering

This is where all the business and system requirements are elicited, analyzed, evaluated and validated.

- Design

This is where the system components are designed along with the design of the system’s logical structure.

- Implementation

It is the process where system programming and construction are performed to present the software.

- Testing

As any software should be reliable and works without failure in the intended requirements to consider it a successful software [8]. The testing where it is a verification & validation process to test and ensure that the implementation is done correctly and that the system is performing in the right manner.

- Maintenance

It is about enhancing and modifying the system to meet requirements changes. Furthermore, the software process has a collection of actions and tasks to be performed. In fact, the software process is adaptable to allow software engineers to select the appropriate set of tasks as well as the work actions. Some of the software process models support this adaptation [9].

2.3. Process framework

This constructs the foundation of the software engineering processes by fronting several framework operations. These framework activities are suitable for any software project regardless of complexity or size. The generic process framework for software engineering includes five activities, namely:

- Communication

It is the first activity and involves requirements gathering, where the software engineers communicate with customers and other relevant stakeholders. The need of the communication between the stakeholders is a fact, which needs management through the software process [10].

- Planning

It involves describing technical tasks to be conducted, the required resources, the required work products to be produced, the possible risks, and the work schedule [9].

- Modeling

It involves creating models that assist the client/developer to understand the requirements needed for the software [11].

- Construction

This is where the actual software is built, and it involves generating codes and eventually testing the software to determine the presence of any errors and bugs.

- Deployment

Which is the final activity of the software process. It involves delivering the product to the client who will evaluate it and give their feedback [9].

Generally, the software process details differ from one software project to another, whereas the framework activities remain the same for all software projects. Moreover, the framework activities are applied in an iterative manner in some software projects. Then again, other software projects have applied the activities in a sequence manner [12].

2.4. Agile

Agile means the ability to move quickly, and easily, and in SE it is a SD method where the software is implemented in an iterative software process. Agile has been the lead method of all process in software project due to its ability, flexibility, and tolerance in accepting the requirement changes. The continuous communication with customers helps to get the actual need from them in addition with getting the requirements based on previous cases [13]. Here where agile considering this and where the product could be implemented in iterations, where the customer provides their recommendations and the quick implementation will be processed in the Agile. Here in the agility process, the team can work in divide the software delivery process in N iterations depending on the business needs and plan [14]. Agile is about teamwork and personal interaction over tools, in order to deliver a running and usable software product over documentations as in linear software processes, and about negotiation over contracts. One of the agile models is the Scrum model, which is the most process model being used in delivering software products.

2.5. Scrum

Scrum is a framework that could be described as a lightweight framework that defines a set of steps to manage and control the SD process. In SE scrum is a mash-up of iterative and incremental SD agile model and it mainly focuses on team collaboration and self-organization. The objective of designing Scrum was to increase development speed, define a culture to focus on performance, support stakeholder value creation and improve individual development and quality of life [15-17].

2.6. Scrum roles

Scrum has three core roles, namely, the product owner, scrum master, plus scrum team, each of which has a set of tasks and responsibilities [18].

- Product owner

It is supposed to understand the business and customer needs. In addition, the product owner accounts for the requirements prioritization, and ensures that the scrum team delivers the best value to the enterprise.

- Scrum master

It is an accounting for elimination and removal of impediments from the scrum team to enhance the deliverables. In fact, they coach the scrum team and motivates them to focus on the tasks, ensures following and applying the process as supposed.

- Team

This team is responsible for delivering the increments, and it consists of three to nine members; each member has cross-functional skills. The team members are accountable for developing the software and their main tasks include analyzing the requirements, designing the software, developing and testing the software [19].

2.7. Scrum flow

First, Scrum has a basic unit of development, which called sprint. Sprint's duration ranges from one-four weeks where each sprint increment is ready for release to the customer. As can be seen in Figure 2 Scrum consists of three artifacts; product backlog, spring backlog and in increment. The product backlog is a selected list of requests/topographies which created by the product owner. Once a feature is added to the backlog, it is called a user story; thus, a sprint buildup is a list of user stories that will be developed in the current sprint. The last artifact is the increment, which is a set of items of product backlog completed during a sprint. Releasing the increment is a decision by the product owner. In addition, Scrum has a set of key meetings; planning for sprints, daily scrum, review of sprint plus sprint retrospective. The scrum master enables all of these consultations and those in attendance include the owner and the scrum team. In the planning summit, the client clarifies the utmost priority user stories and the sprint backlog is initiated after the agreement with the scrum team. The second meeting is the daily scrum to follow up with the scrum team and ensure that there are no blockers or issues that could affect the progress. The third consultation is a sprint review for the progress review. And, the fourth meeting is sprint retrospective where the lessons learned from the current sprint are identified to be applied next sprint to improve the process [18-20].

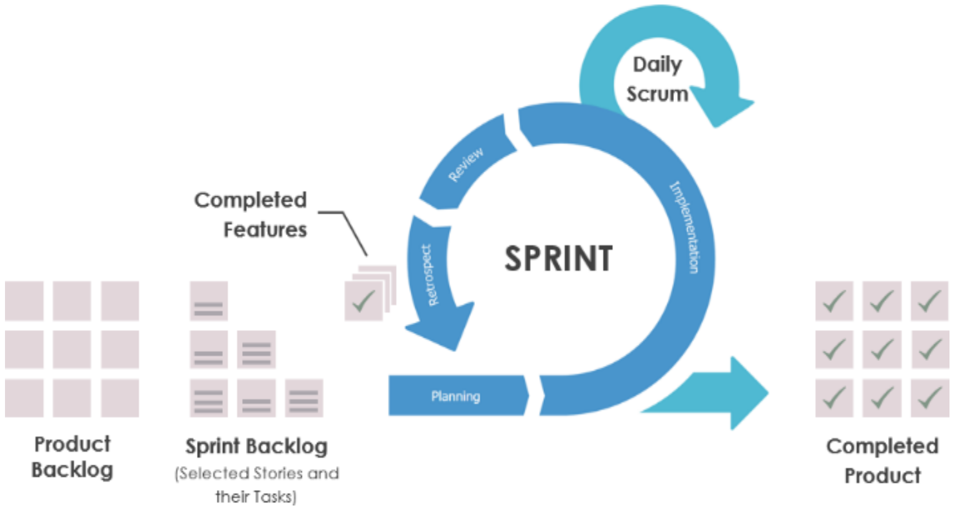

Figure 2. Scrum model process

2.8. Scrum issues

Though Scrum is considered as one of the great models of agile, yet it has some issues [21, 22]. As the continuous incrementing changes by the client in addition of a critical changes must be done to the production which could be avoided if was changes are accepted directly before going to the production [23].

3. RESEARCH METHOD

A questionnaire distributed online among employees who worked in the development of different solutions for different sectors; such as Educational sector, Health sector, and governmental sector. This online questionnaire implemented to evaluate and see the existence of the issues, and to be able to provide a solution to overcome them. Moreover, the companies' size where the questionnaire took a place goes between 5000 employees to +20,000 employees.

The questionnaire results show that 56.66% of the scrum team works in 2 and more sprints at the same time. Yet the team who works on these sprints, 70% of them do accept changes and 50% do apply it within the current sprint working sprints and 50% do not have a change control process to control the sprints. Moreover, 83.33% of the teams said that the product owner does make changes to the working sprints. In addition, because of these issues and more of them 62.07% has said that this adds extra pressure to the team work, which is against the concept of scrum that clarifies scrum team are a team who works in harmony and having a happy member. The SE's has analyzed the questionnaire results and summarized the main critical issues that affect the scrum process.

A list of issues resulted from the questionnaire and from the perspective and experience of the scrum team, which have been working with scrum for years, and with different business domains. The collected data represent flaws, which this paper has checked to for discussion to have a solution for them. The flaws summarized in points:

- Product owner continuous changing

One of the flaws is when the product owner keeps asking for changes for the sake of business urges and customer needs. Moreover, this is affecting the sprint planning, team focus, and team harmony at work.

- No changing accepted in working sprints

Not accepting the changes is more suitable for medium to small businesses. However, for huge and critical business domains as health and governmental solutions, continues requirement changes are predictable. In addition, for those changes, scrum team will be always under pressure because of the added work and scrum will suffer from the uncontrolled changes that the product owner keeps requesting as explained in the previous point. Which these changes will affect the actual plan and schedule for a project estimation, leading to providing inaccurate plan which will cause a delay on the actual delivery date. Having such this negative impact can cause critical risks [24].

- Sprints and release time limitations

Here, the concept that the sprint has a duration of two-to-four weeks, which is not enough time to have a release with much crucial functionality; especially in the big domain business. Which lead to the dissatisfaction of the customer in the way of not satisfying the business with these small releases.

- Wasting time

In the current scrum, each team will go in the phase of "not-doing-anything" and waiting for others. As the QA team of the sprint waiting for the developers to finish their work to so they can start do testing, and this is done in the beginning of each sprint. The same as the developer who either waits for the QA feedback so they can be released from the sprint or wait to receive the QA finding bugs in which to be solved. Imagining how this is happening into many sprints for big projects, how many hours will be the left resulting of this time, and where the team could use it doing some actual work.

3.1. Proposed model

In the proposed model show in Figure 3, we have tried to put more attention in the gaps caused by the previously summarized problems, which affect the current business to control them by adding one more rule and regression test for an N+1 sprint before each release. The basic concept is to add a "Control Master" rule, to control the slipping and the uncontrolled requirements that occur in the sprint while it is under process. Neither the developers nor the quality assurance QA team can control those added changes by the product owner because of the urgent cases. The added changes are explained as:

- Control master

It is a new rule that is responsible for controlling the requirement changes in the scrum lifespan; by studying those changes before adding them to the sprint and disturbing unnecessarily. Instead, let the team focus on the implementation of current sprint backlog for the control master to attend the daily meeting and evaluate changes and their effectiveness.

- N-sprint

Here is the actual concept of the sprint; however, instead of having one release for one sprint, it will involve one release for several sprints at once. This process suite the big business that needs the production releases to have as much functionality as it could for the urge of the business needs. Explaining the concept involves having the resources working in the first sprint each in their rule, and when the developer finishes their first sprints and hand the work over to the QA to start, the developer here can directly start working on the backlog of the second sprint. When they finish from the second sprint, the developer might go back to the first sprint to solve any finding bugs, or they can move to N number of sprints. The same applies for the QA; but for clarifications, each sprint will still have its own planned backlog. Eventually, after finishing the N sprints, the QA will do a regression test before the release.

- Regression test

Here the QA team will test the N-Sprints that will be combined into one increment in order to ensure the stability of the increment-the changes or bug fixes do not affect the previous functionalities, and they are not introducing any new errors [25]. Moreover, regression tests ensure that all the requirements have been implemented and the increment is ready to be released to the customer. In the proposed model, the regression test will be applied at the end of N-Sprints to ensure that all sprints have been implemented, no defects exist, and that the N-Sprints are ready to be released. Eventually, the naming is based on the model that has the concept of the scrum model and the added changes shape from the first eye site it looks like a bear, which leads to name this model to be "Scrumbear".

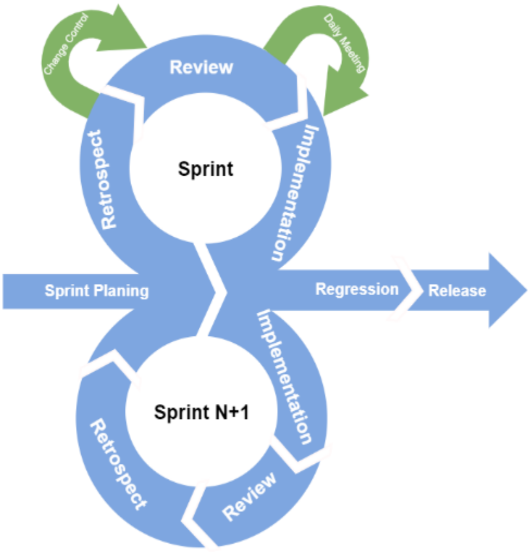

Figure 3. Scrumbear proposed model

4. RESULTS AND DISCUSSION

The findings of the questionnaire indicate that 56.66% of the scrum team operates concurrently in 2 and more sprints. However, the team that works on these sprints, 70% support improvements and 50% implement it to the existing sprint work sprints and 50% have no change management mechanism to management the sprints. In addition, 83.33% of the teams said the owner of the company is making improvements to the working sprints. In addition, 62.07% said this brings additional strain to the team because of these problems and more of them, which are against the scrum idea that clarifies scrum team, are a team that operates in a good environment and has a satisfied member. The SE's evaluated the results of the questionnaire and outlined the principal critical issues affecting the scrum process.

As in the comparison, in Table 1 the new model will help in resolving some of the flaws that already exist and discussed in the current scrum. Through the adoption of Scrumbear, the scrum team will be able to work in the scrum environment, which has a requirement of changing the management process to control and manage the entire process. Due to this change, the team will start working in the new changes and by having the control master with the collaboration of the scrum master it ensures that there is no pressure in completing the sprints while other sprints are working, at the same time, the team time will be fully used without affecting each sprint time.

Table 1. Comparison between traditional Scrum and Scrumbear

Discussion	Description	Traditional Scrum	Scrumbear
Control Changes	Where the rule played by the control master to control the changes	NO	YES
Accept Changes	To allow changes within the current working sprint	NO	YES
Team time wasting	The time where a team member does not do anything and waiting the result from another team member.	YES	NO

5. CONCLUSION

In conclusion, software engineering is an important subject especially now that the world of technology is growing at a very fast pace. It is on this background that the paper "Scrumbear" framework to overcome the traditional Scrum model problems, has deliberated on how teams can improve their output. After discussing the summarized collected problems, the product owner continuous changing, no accepted changes in the active sprints, the time limitation of sprints and release, and wasting time that could be used in other active sprints to fast-moving the production releases delivery. By using the new proposed Scrumbear framework, those problems would be tackled and solved. It shown that traditional Scrum model is not support the control of changes and not even accept the changes since the full authorized changes are in business owner/software owner decisions. Furthermore, based on evaluation results of the Traditional Scrum, it also proved that the wasting of the team member's time where a team member does not do anything and wait for the result from other team member in order to proceed for the next Sprint. All the problems arise from traditional Scrum are solved by implementation of the Scrumbear framework.

ACKNOWLEDGEMENTS

This work was partially supported by the Artificial Intelligence Data Analytics Lab (AIDA), Prince Sultan University, Riyadh, Saudi Arabia.

REFERENCES

[1] Fachrurrazi, "The end user requirement for project management software accuracy," *International Journal of Electrical and Computer Engineering*, vol. 8, no. 2, pp. 1112-1121, 2018.

[2] A. Verma and S. Kaur, "Design and development of an agorithm for prioritizing the test cases using neural network as classifier," *IAES International Journal of Artificial Intelligence*, vol. 4, no. 1, pp. 14-19, 2015.

[3] M. Ahmadi, B. B. Rad, and M. O. Thomas, "Tailoring software development methodologies for reliability," *Journal of Telecommunication Electronic and Computer Engineering*, vol. 10, no. 3, pp. 117-121, 2018.

[4] A. M. Dima and M. A. Maassen, "From waterfall to agile software: Development models in the IT sector, 2006 to 2018. Impacts on company management," *Journal of International Studies*, vol. 11, no. 2, pp. 315-326, 2018.

[5] T. Linz, "Testing in scrum: A guide for software quality assurance in the agile world," Rocky Nook, Inc., p. 224, 2014.

[6] R. Mall, "Fundamentals of software engineering-5th ed," PHI Learning, pp. 612, 2018.

[7] J. Xiong, "Practical training of software engineering talent in one integration and four layers mode," *World Transactions on Engineering and Technology* Education, vol. 14, no. 1, pp. 146-150, 2016.

[8] D. Hema Latha and P. Premchand, "Estimating software reliability using ant colony optimization technique with salesman problem for software process," *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 7, no. 2, pp. 20-29, 2018.

[9] P. Clarke, R. V. O'Connor, B. Leavy, and M. Yilmaz, "Exploring the relationship between software process adaptive capability and organisational performance," *IEEE Transactions on Software Engineering*, vol. 41, no. 12, pp. 1169-1183, Dec. 2015.

[10] P. Thakur and R. Shrivastava, "A review on text based emotion recognition system," *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 7, no. 5, pp. 67-71, 2018.

[11] S. Aleem, L. F. Capretz, and F. Ahmed, "Critical success factors to improve the game development process from a developer's perspective," *Journal of Computer Science and Technology*, vol. 31, no. 5, pp. 925-950, 2016.

[12] R. S. Pressman and B. Maxim, "Software engineering: A practitioner's approach-8th ed.," McGraw-Hill Education, p. 18, 2014.

[13] O. Oueslati, A. I. S. Khalil, and H. Ounelli, "Sentiment analysis for helpful reviews prediction," *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 7, no. 3, pp. 34-40, 2018.

[14] P. Abrahamson, O. Salo, J. Ronkainen, and J. Warsta, "Agile software development methods: Review and analysis," VIT Publication, Finland, pp.1-107, 2002.

[15] F. ur Rehman, B. Maqbool, M. Q. Riaz, U. Qamar, and M. Abbas, "Scrum software maintenance model: Efficient software maintenance in agile methodology," *2018 21st Saudi Computer Society National Computer Conference (NCC)*, pp. 1-5, 2018.

[16] R. Kumar, P. Maheshwary, and T. Malche, "Inside agile family software development methodologies," *International Journal of Computer Sciences and Engineering*, vol. 7, no. 6, pp. 650-660, 2019.

[17] A. Srivastava, S. Bhardwaj, and S. Saraswat, "SCRUM model for agile methodology," *2017 International Conference on Computing, Communication and Automation (ICCCA)*, pp. 864-869, 2017.

[18] S. S. Samarawickrama and I. Perera, “Continuous scrum: A framework to enhance scrum with DevOps,” *2017 Seventeenth International Conference on Advances in ICT for Emerging Regions (ICTer)*, pp. 1-7, 2017.

[19] S. Hariharan, A. Rengarajan, and R. Prem Kumar, “Scrum based scaling using agile method to test software projects and its future solutions using in artificial neural networks,” *International Journal of Innovative Technology and Exploring Engineering*, vol. 8, no. 9, pp. 223-230, 2019.

[20] J. Sutherland, “Agile development: Lesson learned from the first scrum,” *Cutter Agile Project Management Advisory Service: Executive Update*, vol. 5, no. 20, pp. 1-4, 2004.

[21] J. Cho, “Issues and challenges of agile software development with SCRUM”, *Issues in Inoformation System*, vol. 9, no. 2, pp. 188-195, 2008.

[22] D. Maximin, “The scrum culture: Introducing agile methods in organizations,” Springer, 2018.

[23] J. Miler and K. Kaidy, “Problems and solutions of software design in scrum projects,” *2018 Federated Conference on Computer Science and Information Systems (FedCSIS)*, pp. 975-978, 2018.

[24] A. Zakrani, M. Hain, and A. Idri, “Improving software development effort estimating using support vector regression and feature selection,” *International Journal of Artificial Intelligence*, vol. 8, no. 4, pp. 399-410, 2019.

[25] B. Srisura and A. Lawanna, “False test case selection: Improvement of regression testing approach,” *2016 13th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, pp. 1-6, 2016.