Blockchain applications for Agile methodologies

Valentina Lenarduzzi
Tampere University of Technology, Finland
valentina.lenarduzzi@tut.fi

Maria Ilaria Lunesu
University of Cagliari, Italy
ilaria.lunesu@diee.unica.it

Michele Marchesi
University of Cagliari, Italy
marchesi@unica.it

Roberto Tonelli
University of Cagliari, Italy
roberto.tonelli@dsf.unica.it

ABSTRACT

We present an application of Blockchain technology and Smart Contracts to the management of Agile projects, using Scrum or Lean-Kanban processes. In our application the duties of the Product Owner for certifying the correctness of the outcomes are delegated to one or more Smart Contracts deployed on the Ethereum Blockchain and written in Solidity. An agreement with the Customer can also allow the Smart Contracts to automatically enable payments, to introduce penalties or grants on the basis of the outcome. Product Owner duties and work can thus be relieved allowing to allocate resources on more profitable and productive tasks. Other possibilities are examined as well.

KEYWORDS

Scrum, Lean-Kanban, smart contract, Ethereum

ACM Reference Format:
Valentina Lenarduzzi, Maria Ilaria Lunesu, Michele Marchesi, and Roberto Tonelli. 2018. Blockchain applications for Agile methodologies. In *Proceedings of ACM Conference (XP'18).* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Agile methodologies are rooted in adaptive planning, early delivery and continuous improvement, all with an eye toward being able to respond to change quickly and easily. The processes based on these principles are multiples and organized in several steps. Some of them presume human actors key roles, some others, such as verification of acceptance criteria or test verification do not necessarily need any human decision or impression and seem suitable for being automated through other automatic tools or means. Our proposal is to use the new emerging blockchain technology in order to partially automate some specific tasks belonging to the Agile software development. We focus our work on the role of Product Owner, but our proposal can be extended to other roles or parts of the Agile Lean-Kanban or Scrum processes. In this work we use the technology of Blockchain to support the PO in managing the acceptance phase of an agile project, performed using Scrum or Lean-Kanban process.

We present an approach where Smart Contracts[1] are used to execute the acceptance tests and to verify the fulfillment of the conditions fixed at the beginning of the project, in order to avoid their modification or misconstruction.

Smart Contract (or a pull of Smart Contacts) can be used to verify the compliance to the initial given acceptance conditions and that all the test passed. We can also include the payment of the work done (computed on implemented user stories), into the Smart Contracts duties and automatically activate the payment to the developers once the work done has been validated and approved. Smart Contract render also any activity transparent and visible to all the responsible team since the blockchain is publicly accessible and immutable.

2 BACKGROUND

Agile methodologies allow developers to achieve a very good trade-off between effective coordination and process agility [1], allowing its adoption and switching from other processes without negatively affecting process quality or metrics [15][16][14], and enabling a good level of communication between teams [8] [6].

Agile methodologies, such as Scrum, are based on an incremental development model and follow three principles: any activity must be visible to all the responsible team, concurrent artifacts inspection, rapid process adaptation (in case of negative inspection results). The key roles of the process are the Product Owner and the developers. In Agile processes, the development can often be structured in iterative cycles (sprints) where developers present the work done to the PO and the PO discusses with them the requirements to be implemented. Requirements are generally structured as a list of "user stories". Different approaches can be applied in the different methodologies (XP, Scrum, and others) [2][5]. As an example, in Scrum, before each Sprint, in the Planning meeting a Sprint Backlog is created [3][4]. The Sprint Backlog is an ordered list of Product Backlog Items, preferably User Stories [10] that the team believes it can complete during the Sprint [13]. One of the problems of the user stories elicitation is the communication between PO and developers team [8] and then the decomposition of the high-level requirements into low-granularity user stories [9]. The PO, in order to evaluate the work done during the process iterations, verifies also if the acceptance criteria (specific requirements given by the customer [7]) are fulfilled [9] (acceptance/automatic tests might be

[1]Smart contracts are piece of software code deployed in a Blockchain, typically Ethereum, that can be called through Blockchain transactions for executing their code.

XP'18, May 2018, Porto, Portugal

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

used). Our proposal, in the presented use case, aims at relieving the duties of the PO for certifying the correctness of the process outcome delegating them to one or more Smart Contracts deployed on the Ethereum Blockchain and written in Solidity[2] and at automating some specific tasks. An agreement with the Customer can also allow Smart Contracts to automatically enable payments and possibly to introduce penalties or incentives, based on the outcome. PO duties and work can thus be relieved allowing to allocate resources on more profitable and productive tasks. Thanks to the characteristics of Blockchain, once the Smart Contracts are deployed the code cannot be changed and time stamps registered on the Blockchain blocks for each transaction can trace all operations performed by Smart Contracts. In particular the Blockchain natively supports hashing and cryptography, so that activation of Smart Contracts functionality can be controlled by the contract owner. All transactions in the Blockchain public ledger can easily be verified in a transparent way and are immediately available to all process actors.

3 MODEL IMPLEMENTATION

The project development is divided in User Stories (US) implementation (or features, or MMF). The implementation flow may be in Sprints (groupings of US to implement in a given time interval) or continuous (as when using a Kanban board). Each US is provided with one or more Acceptance Tests (AT), whose correct result can be coded in a Json file, whose hash digest is registered in the blockchain by the PO with his cryptographic credentials. When the corresponding US is correctly implemented by the developer who committed to it, the test should pass, generating a Json file identical to the correct one, and hence with the same hash digest. This, with a further possible OK by the PO, triggers the run of Smart Contract code on the blockchain to certify US validation comparing the two hash digests and eventually enabling payment in a cryptovalue or in "tokens". Below are the cases of continuous flow and Scrum, as depicted in Figure.

In case of Continuous flow.

1 The customer creates a SC for the project and holds a private key, allocates Ethers for payments and defines PO address and private key;
2 the PO registers on the SC the user stories (US). Each user story has the following data-structure: US-Id, value, state: open/closed, developer that work on it, test suite of acceptance test (with ID, US ID, hash of desired output), state (passed, not passed), sprint nr (if Scrum process);
3 PO registers on the SC the developers address and the user stories they work on using his cryptographic credentials;
4 developers, for each US, execute the test suite according to inputs and planned tests; if tests pass, register them sending to the Smart Contract the hash digest of the generate Json using their credentials;
5 The SC code runs: checks the hashes; if all user stories tests pass sets a green light, then can automatically send the associated ETH to the developers address. There could be, also, a further check (optional) in which the PO verifies if SC validated all tests (setting a flag to TRUE), then SC allows a transaction with "passed test" and transfers Ether to developers wallets.

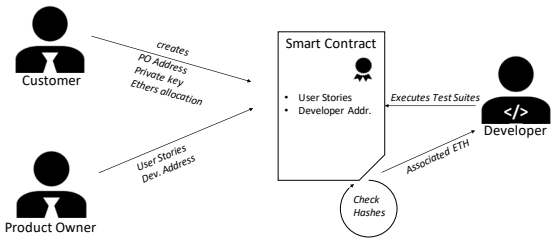

Figure 1: The Model Implementation - Continuous flow

In case of Scrum process. Passes 1-4: the same of continuous flow
4.a The PO, at the start of each Sprint: assigns proper USs to the Sprint and registers the current Sprint
4.b The developers, for each Sprint: subscribe to the US chosen and run the AT(s) for each US and, if OK, register the passed AT sending the hash digest of the generate Json
5 At the end of each Sprint the PO verifies the Sprint status and confirms the completed USs for the Sprint, then the SC sends the ETHs associated to completed USs to the developer's wallets.

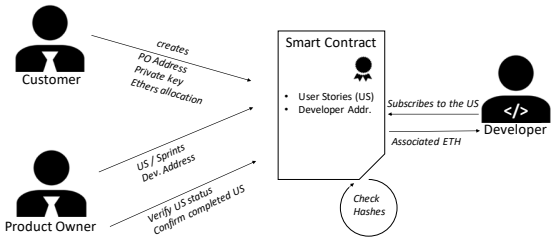

Figure 2: The Model Implementation - Scrum process

4 CONCLUSIONS

We present an approach where Smart Contracts and Blockchain are used to leverage part of the PO duties in a Lean-Kanban or in a Scrum process. This approach may seem to be opposite to the Agile mindset open to the continuous changes and based on mutual trust. On the contrary we believe that the native nature of the blockchain as a trustless technology perfectly suites the spirit of mutual trust of Agile Manifesto.

A recurring thought is that blockchain can "create trust", or allow different parties to make a transaction "without relying on trust". Trust is a crucial glue mostly for Agile community. The blockchain does not create or eliminate trust but it merely converts trust from one form to another. Likewise in our work we propose to use blockchain and smart contracts for the automatic verification

[2]Solidity is a high-level "contract-oriented" language for smart contracts implementation. It is inspired by C++, Python and JavaScript and is designed to be compiled into bytecode and run on EVM

that test passed without the direct involvment of the PO with the unique purpose of relieving some of his duties.

Smart Contracts render any activity transparent and visible to all the responsible team since the blockchain is publicly accessible and immutable.

Likewise Blockchain might be a good way to record the work flow and to track the enhancements of the product under work as well as the productivity of developers (maybe located in different places) and by using Smart Contracts as a payment support.

We use the blockchain technology as a support for the role of PO, helping him during the acceptance criteria phase and taking advantage of blockchain capabilities, not involving any aspect of trust among people. Blockchain technologies could also mitigate the difficulties in communication among people belonging to different sites and different behaviours.

In fact, the use of blockchain for tracking the work in flow and the issues closed by each developer or team member could allow the customer to automatically be informed on each activity performed during the process.

Our future plans are to investigate about how other steps related to several software development processes, usually conducted by human reasoning, can be implemented as a blockchain "transaction" managed by Smart Contracts. The main challenge in this work is to try to take advantage of the Blockchain strengths to complement the possible weaknesses of Agile-Lean approaches.

The continuous learning process acquired through this work flow recording could be highly beneficial to the whole team to continuously learn without requiring a lot of effort for collecting data, as also recommended by the Agile Experience Factory [11, 12].

Presently we are working on a practical implementation of the model analysing the set of Smart Contracts needed to perform all the operations required to manage the automated testing procedure, the allocation of Product Owner permits and capabilities for managing the related Smart Contracts. Finally we are planning to writing the solidity code and testing it into a private blockchain.

REFERENCES

[1] K. Schwaber, M. Beedle. Agile Software Development with SCRUM. Prentice Hall, Englewood Cliffs (2001)

[2] K. S. Rubin. Essential Scrum: A Practical Guide to the Most Popular Agile Process (1st ed.). Addison-Wesley Professional. (2012)

[3] E. F. Cruz, R. J. Machado and M. Y. Santos. On the Decomposition of Use Cases for the Refinement of Software Requirements. International Conference on Computation- al Science and Its Applications. (2014)

[4] J. Pauli and D. Xu. Integrating functional and security requirements with use case decomposition. International Conference on Engineering of Complex Computer Systems (ICECCS'06). (2006)

[5] J. Sutherland, A. Viktorov, J. Blount, and N. Puntikov. Distributed Scrum: agile project management with outsourced development teams. Hawaii International Conference on System Sciences (HICSS), IEEE Press, pp. 274a.(2007)

[6] V. Lenarduzzi, I. Lunesu, M. Matta and D. Taibi. Functional Size Measures and Effort Estimation in Agile Development: A Replicated Study. Agile Processes in Software Engineering and Extreme Programming (XP2015). pp. 105-116. (2015)

[7] D. Taibi and V. Lenarduzzi. MVP explained: A Systematic Mapping on the Definition of Minimum Viable Product. Euromicro Conference on Software Engineering and Advanced Applications (SEAA2016). pp. 112-119. (2016)

[8] D. Taibi, V. Lenarduzzi, M. O. Ahmad, and K. Liukkunen. Comparing Communication Effort within the Scrum, Scrum with Kanban, XP, and Banana Development Processes. International Conference on Evaluation and Assessment in Software Engineering (EASE'17). pp. 258-263. (2017)

[9] D. Taibi, V. Lenarduzzi, A. Janes, M.O. Ahmad and K. Liukkunen. Comparing Requirements Decomposition Within the Scrum, Scrum with Kanban, XP, and Banana Development Processes. Agile Processes in Software Engineering and Extreme Programming (XP2017). pp. 68-83. (2017)

[10] J. Patton and P. Economy. User Story Mapping: Discover the Whole Story, Build the Right Product. O'Reilly Media, Inc. (2014)

[11] D. Taibi, V. Lenarduzzi, P. Diebold, and I. Lunesu. 2017. Operationalizing the Experience Factory for Effort Estimation in Agile Processes. 21st International Conference on Evaluation and Assessment in Software Engineering (EASE'17). pp. 31-40. (2017)

[12] Destefanis, G., Counsell, S., Concas, G., Tonelli, R. (2014, May). Software metrics in agile software: An empirical study. In International Conference on Agile Software Development (pp. 157-170). Springer, Cham.

[13] N. C. Haugen. An empirical study of using planning poker for user story estimation. Proceedings of AGILE Conference, pp. 9-34. (2006)

[14] Turnu, I., Marchesi, M., Tonelli, R. (2012, June). Entropy of the degree distribution and object-oriented software quality. In Proceedings of the 3rd International Workshop on Emerging Trends in Software Metrics (pp. 77-82). IEEE Press.

[15] L. Lavazza, S. Morasca, D. Taibi, and D. Tosi. Applying SCRUM in an OSS Development Process: An Empirical Evaluation. Agile Processes in Software Engineering and Extreme Programming (XP2010). pp. 147-159. (2010)

[16] Concas G., Marchesi M.,Destefanis G., Tonelli R., An empirical study of software metrics for assessing the phases of an agile project. International Journal of Software Engineering and Knowledge Engineering, Vol. 22, n.4, p. 525-548 (2012).