2017 IEEE/ACM 39th IEEE International Conference on Software Engineering Companion

Blockchain-oriented Software Engineering: Challenges and New Directions

Simone Porru, Andrea Pinna
University of Cagliari
Department of Electrical and Electronic Engineering
[simone.porru, a.pinna]@diee.unica.it

Michele Marchesi, Roberto Tonelli
University of Cagliari
Department of Informatics and Mathematics
marchesi@unica.it, roberto.tonelli@dsf.unica.it

***Abstract*—In this work, we acknowledge the need for software engineers to devise specialized tools and techniques for blockchain-oriented software development. Ensuring effective testing activities, enhancing collaboration in large teams, and facilitating the development of smart contracts all appear as key factors in the future of blockchain-oriented software development.**

***Keywords*-blockchain; software engineering; smart contracts; cryptocurrencies;**

I. Introduction

In the past years, a lot of attention has been paid to the emerging concepts of blockchain and smart contract. Some observers are even talking of the dawn of a new era [1] and about the potential of reshaping the current financial services technical infrastructure [2] [3]. Ever since digital currencies started to represent a real monetary value, also hacks and attacks started. The biggest was the MtGox attack and another remarkable exploit was that sustained by the DAO organization in June 2016. Concerning software development, the scenario is that of a sort of competition *first-come-first-served* which does not assure neither software quality, nor that all the basic concepts of software engineering are taken into account.

This paper aims at revealing the current issues and new directions for blockchain-oriented software engineering, and investigates the need for novel specialized software engineering practices for the blockchain sector. To this purpose, we:

1) identified the most relevant challenges for the state-of-practice blockchain-oriented software engineering;
2) highlighted peculiarities of some of the most popular blockchain-oriented software projects going on in the world;
3) proposed new research directions for blockchain-oriented software engineering, based on the results obtained from the previous steps.

II. Blockchain-oriented Software Engineering: Challenges

We define as blockchain-oriented Software (BOS) all software working with an implementation of a blockchain. A blockchain is a data structure characterized by the following key elements:

- data redundancy (each node has a copy of the blockchain);
- check of transaction requirements before validation;
- recording of transactions in sequentially ordered blocks, whose creation is ruled by a consensus algorithm;
- transactions based on public-key cryptography;
- possibly, a transaction scripting language.

Considering the distinctive marks of a blockchain, software engineers could benefit from the application of BOS-specific software engineering practices. Such practices would constitute the base of a blockchain-oriented Software Engineering (BOSE). We identify the most relevant BOSE challenges, and the issues which originate from them. To this purpose, for most challenges, we also provide excerpts from the SWEBOK [4]. to properly frame the related issues.

New professional roles. *"A recognized profession entails specialized skill development and continuing professional education"* Due to the business-critical nature of the blockchain, finance and legal subjects have shown increasing interest toward BOS. At the same time, bootcamps for blockchain developers are flourishing. The blockchain sector will need professional figures with a well-defined skills portfolio comprising finance, law, and technology expertise. An example of a new role could be that of an intermediary between business-focused contractors with low technology expertise and IT professionals.

Security and reliability. *"Software Security Guidelines span every phase of the software development lifecycle"* and *"Software Reliability Engineered Testing is a testing method encompassing the whole development process"*. A blockchain must guarantee data integrity and uniqueness to ensure blockchain-based systems are trustworthy which, in the case of BOS, is that of security-critical systems. In particular, there is a need for testing suites for BOS. These suites should include:

- Smart Contract Testing (SCT), namely specific tests for checking that smart contracts i) satisfy the contractors' specifications, ii) comply with the laws of the legal systems involved, and iii) do not include unfair contract terms.
- Blockchain Transaction Testing (BTT), such as tests against double spending and to ensure status integrity (e.g. UTXO[1]).

[1]Unspent Transaction Output

978-1-5386-1589-8/17 $31.00 © 2017 IEEE
DOI 10.1109/ICSE-C.2017.142

Software architecture. Specific design notations, macroarchitecture patterns, or meta-models may be defined for BOS development. To this purpose, software engineers should define criteria for selecting the most appropriate blockchain implementation, evaluating the adoption of sidechain technology, or the implementation of an ad-hoc blockchain. For example, Ethereum[2] has adopted a key-value store, which is a very simplistic database. By adopting a higher level data representation such as an Object Graph, it would be possible to speed up many operations which would otherwise be expensive using a key-value store [5].

Modeling languages. Blockchain-oriented systems may require specialized graphic models for representation. More specifically, existing models might also be adapted to BOS. UML diagrams might be modified or even created anew to account for the BOS specificities. For example, diagrams such as the Use Case Diagram, Activity Diagram, and State Diagram could not effectively represent the BOS environment.

Metrics. BOSE may benefit from the introduction of specific metrics. To this purpose, it could be useful to refer to the Goal/Question/Metric (GQM) method, that was originally intended for establishing measurement activities, but that can also be used to guide analysis and improvement of software processes [4] [6] [7]. Due to the distributed nature of the blockchain, specific metrics are required to measure complexity, communication capability, resource consumption (e.g. the so-called gas in the Ethereum system), and overall performance of BOS systems.

III. Blockchain-oriented Software Repositories

In order to define new research directions for the BOSE on the basis of the state-of-practice of blockchain-oriented software, we conducted an exploratory study on a corpus comprising 1184 GitHub software repositories, which were identified with the use of the Moody's blockchain Report [8] and CoinMarketCap[3].

We define a BOS project as a software project which contributes to the realization of a blockchain project. This definition includes both blockchain platforms, such as Bitcoin and Ethereum, and general blockchain software [9]. To identify BOS repositories we start from the corresponding blockchain projects. Moody's Investor Services recently identified more than 120 publicly announced blockchain projects in an in-depth report of the blockchain sector [8]. And we focused on the first 17 most capitalized currencies and assets.

To focus on the most relevant repositories, we retained 193 repositories out of the initial 1184. We extracted information on popularity (*Stargazers*), programming languages, community involvement (*Contributors*, *Open Issues*, *Watchers*, *Forks*), and age (time elapsed since creation).

We found that JavaScript, Python, Go, C++, and Ruby are the top 5 languages, with BOS JavaScript repositories accounting for more than 30% of the total. The number of Java repositories don't reach the 4% of the total. The top 10 BOS repositories were created around 4 years ago on average, and most of them have a considerable number of open issues. The statistics about forks are staggering, topping at 4266 for the main bitcoin repository, followed by the Go repository from Ethereum with 695 forks.

IV. New Research Directions

We hereby suggest some new research directions for BOSE.

Testing. A recent study on over 50000 GitHub projects [10] has proved that a bigger team size leads to a higher number of test cases, whereas the number of test cases per developer decreases with an increase in the team size. It would be interesting to investigate whether the same can be said about BOS, considering that the most popular repositories have an unusually high number of contributors, even for open-source projects.

Collaboration. The high number of voluntary contributors testifies to the attractiveness of BOS in the open source landscape. A large base of voluntary contributing members has been shown to be a pivotal success factor in OSS evolution [11]. To achieve sustainable development and improve software quality, specific practices to enhance the synergy between the system and the community would be highly beneficial to BOSE [12].

Enhancement of testing and debugging A number of programming languages such as Go, Python, and Ruby are gaining increasing popularity among BOS projects. This arises the need for enhanced testing and debugging suites, since bugs are ubiquitous in software systems [13] [14], tailored upon the most popular BOS languages. Indeed, Java testing suites have undergone much more testing than Go. In addition, as BOS projects work with the blockchain, which is distributed by definition, testing in isolation would require properly mocking objects capable of effectively simulate the blockchain.

Creation of software tools for smart contract languages. The implementation of Smart Contract Development Environments (SCDEs)–the blockchain-oriented declination of IDEs–might be pivotal for the building and diffusion of BOS expertise. Such environments could streamline smart contract creation through specialized languages (e.g. Solidity, a language designed for writing contracts in Ethereum).

V. Conclusion

In the present work, we highlighted the most evident issues of state-of-art blockchain-oriented software development, by advocating the need for new professional roles, enhanced security and reliability, novel modeling languages, and specialized metrics. In addition, we used the 2016 Moody's blockchain Report and the market capitalization of cryptocurrencies to build a dataset of blockchain-oriented software repositories. On the basis of the results of the analysis, we proposed new directions for blockchain-oriented software engineering, focusing on collaboration among large teams, testing activities, and specialized tools for the creation of smart contracts.

[2]https://www.ethereum.org/

[3]https://coinmarketcap.com/

REFERENCES

[1] M. Swan, *Blockchain: Blueprint for a new economy*. O'Reilly Media, Inc., 2015.

[2] Unicredit, "Blockchain technology and applications from a financial perspective," 2016.

[3] F. Aymerich, G. Fenu, and S. Surcis, "A real time financial system based on grid and cloud computing," in *Proceedings of the ACM Symposium on Applied Computing*, 2009, pp. 1219–1220.

[4] A. Abran, J. W. Moore, P. Bourque, R. Dupuis, and L. Tripp, "Swebok: Guide to the software engineering body of knowledge 2004 version," *IEEE Computer Society, Los Alamitos, California*, 2004.

[5] D. Larimer, "Introducing bitshares object graph." [Online]. Available: https://goo.gl/TWWSif. Accessed on 2016-10-20.

[6] M. Concas, G.and Marchesi, G. Destefanis, and R. Tonelli, "An empirical study of software metrics for assessing the phases of an agile project," *International Journal of Software Engineering and Knowledge Engineering 22(4)*, pp. 525–548, 2012.

[7] G. Concas, M. Marchesi, A. Murgia, S. Pinna, and R. Tonelli, "Assessing traditional and new metrics for object-oriented systems," in *Proceedings of the 32th International Conference on Software Engineering 2010*, 2010, pp. 24–31.

[8] N. Caes, R. Williams, E. H. Duggar, and M. R. Porta, "Robust, cost-effective applications key to unlocking blockchain's potential credit benefits," 2016.

[9] G. Hileman, "State of blockchain q1 2016," 2016. [Online]. Available: http://www.coindesk.com/state-of-blockchain-q1-2016/

[10] P. S. Kochhar, T. F. Bissyandé, D. Lo, and L. Jiang, "Adoption of software testing in open source projects–a preliminary study on 50,000 projects," in *Software Maintenance and Reengineering (CSMR), 2013 17th European Conference on*. IEEE, 2013, pp. 353–356.

[11] K. Nakakoji, Y. Yamamoto, Y. Nishinaka, K. Kishida, and Y. Ye, "Evolution patterns of open-source software systems and communities," in *Proceedings of the international workshop on Principles of software evolution*. ACM, 2002, pp. 76–85.

[12] M. Aberdour, "Achieving quality in open-source software," *IEEE software*, vol. 24, no. 1, pp. 58–64, 2007.

[13] H. Zhang, "On the distribution of software faults," in *IEEE Transactions on Software Engineering 34 (2)*, 2008, pp. 301–302.

[14] M. Concas, G.and Marchesi, A. Murgia, R. Tonelli, and I. Turnu, "On the distribution of bugs in the eclipse system," in *IEEE Transactions on Software Engineering 37 (6)*, 2011, pp. 872–877.