


Chapter 8

Mechanism for the Systematic Generation of Functional Tests of Smart Contracts in Digital Publication Management Systems

Nicolas Sanchez-Gomez

 <https://orcid.org/0000-0001-9102-6836>
University of Seville, Spain

Javier Jesús Gutierrez

University of Seville, Spain

Enrique Parrilla

Lantia Publishing S.L., Spain


Julian Alberto García García

University of Seville, Spain

Maria Dolores de-Acuña

University of Seville, Spain

Maria Jose Escalona

 <https://orcid.org/0000-0002-6435-1497>
University of Seville, Spain

ABSTRACT

The application of state-of-the-art technologies in functional fields is complex and offers a significant challenge to user and expert teams as well as to technical teams. This chapter presents a mechanism that has been used in a project in the context of digital publications. Ensuring the traceability of digital publications (e-books and e-journals) is a critical aspect of the utmost importance for authors, publishers, and buyers. The SmartISBN project has used blockchain technology to define a protocol for the identification, tracking, and traceability of digital publications. As this was an innovative project that required communication between functional experts (authors, publishers, booksellers, etc.) and technical experts, it was necessary to identify protocols to facilitate communication. This chapter presents the protocol by which the functional tests have been defined and how this has favoured the validation of the project.

1. INTRODUCTION

Blockchain is a disruptive software technology that is advancing rapidly (Olea, 2019), being one of the fundamental technologies driving Digital Transformation today and, given its transversal nature applicable to a wide range of industrial and economic sectors, it is enabling disruption in the economy and in business beyond cryptocurrencies. This potential is largely based on its ability to offer individuals or organizations a communication channel that allows the transfer of rights, values, or real assets (tokenization), through the Internet, in a secure and reliable manner.

The publishing industry is one of the economic sectors in which blockchain technology has great applications because the publishing industry is a data and metadata intensive sector. This means that the quality of operations and their automation are linked to the quantity and quality of this data. From a global perspective, the distribution process and supply chain of digital publications (e-books and e-journals, among other formats) in Spain is a complex process (Martínez Alés, 2001). A wide variety of actors are engaged in this process, each with diverse needs and actions. The following is a summary of these actors to help understand the magnitude of the process.

A digital publication, once written, must enter a digital copy distribution process. This process can take several months or even years and requires a significant financial investment. While on demand publishing mechanisms exist with delivery times of days, they do not offer assimilable quality and are pushed to specific niches. Then, distributors take the publications from the publishers to the points of sale. These outlets may be physical bookshops, online platforms, or both (Magadán-Díaz et al., 2020).

The emergence of innovative technologies for data and metadata storage and management, such as the possibility of massively and automatically extracting information from web pages, as well as the development of new technologies, such as blockchain technology for information recording (Gramoli, 2022) (Alharby et al., 2018), open up the possibility of offering novel alternatives within the publishing industry.

Blockchain technology and, above all, smart contracts can make valuable contributions as discussed throughout this article. In short, this recent technology offers more transparency, security, and efficiency in the tracking of publications (books, journals, etc.) at each stage of the process. For example, in this project, it has been possible to track and trace digital publications from their production to their final sale, which has made it possible to know the status of the publication at all times.

However, before deploying a smart contract in a business environment, it is necessary that any smart contract is verified using rigorous mechanisms that allow

validating its correct operation, since an error or defect in the code that forms it could cause an unreparable effect (Legerén-Molina, 2018). From an engineering perspective, the serious and competitive progress in the implementation of recent technologies such as blockchain and smart contracts requires new processes, methods, tools, and techniques to manage quality in software development and, above all, to ensure the quality of the final product.

Currently, there are several blockchain platforms that support the implementation, deployment, and execution of smart contracts without many restrictions. For example, Ethereum, Hyperledger or EOS platforms, among others (Zheng Z., 2020), allow deploying smart contracts without going through any verification and validation process. In this sense, verification of smart contracts remains an unexplored line of research to date. Any blockchain network may be running smart contracts with unexpected behavior, with serious deficiencies, errors and even security vulnerabilities (Luu, 2016). Unlike classical applications, which can be patched when errors are detected, smart contracts are irreversible and immutable, given the characteristics of the underlying technology.

In this context, the objective of this article is to present a proposal to generate functional test plans based on smart contract specifications. For this purpose, the proposal will be based on early testing principles, which will allow validating the functional quality of smart contracts independently of the blockchain technology used and from the requirements specification stage. In addition, this article describes the validation of our proposal in the SmartISBN project, which was carried out between 2019 and 2022.

The SmartISBN project aimed to develop mechanisms to semi-automatically extract a set of data and metadata to facilitate the management of publications, together with the development of a practical case of application of blockchain technology for the registration of transactions throughout the life cycle of a digital publication until it reaches its final purchaser.

This article aims to present the results achieved with this project, focusing the main part of the article on the practical case developed with blockchain, since it is not only a novel technology but there are few references to practical cases of application outside its original scope.

This technology also imposes new challenges. In particular, the main challenge in this project was the testing of blockchain technology and the use of smart contracts. As the distribution and supply chain is a complex process, it was necessary to cover many tests. As part of the SmartISBN project, the generation of a complete set of tests was systematized to verify that the system worked properly in all steps of the process and satisfied all its participants.

The organization of this work is described below. Section 2 presents the objectives of the SmartISBN project and the fundamentals of blockchain technology. Then,

Section 3 presents a comprehensive literature review to identify gaps in the existing models. Next, Section 4 presents the proposed solution for systematic functional test generation in blockchain environments and how it has been validated in the SmartISBN project. Finally, Section 5 presents conclusions and future work.

2. BACKGROUND

This section describes the background to the proposal presented in this article. To do so, on the one hand, it describes the context of the SmartISBN project, delving into its objectives, the problems it aims to solve and the technical and business challenges it faces. On the other hand, the fundamentals of blockchain technology are presented in general terms.

2.1. SmartISBN Project: Context and Approach

When the SmartISBN project started, there was no uniformity in the metadata (data describing other data, e.g., data describing the information to be managed for each specific digital publication) managed in the publishing sector. This is because of the different approaches and because different systems offer different data sets. This results in publishing management systems having to work with the minimum set of common data, which decreases the power of the management that can be applied.

The mission of the SmartISBN project was to address the problem indicated by researching and developing a metadata model applicable to the publishing sector that would enable the processing of data associated with a digital publication in a unified manner. The project also included the development of tools that allow the appropriate management of the information in the publications and the operations that could be carried out with them. To fulfil this mission, the SmartISBN project had to meet the three objectives briefly described below.

The first objective was to store a publisher's complete catalogue information in an automated way. This automation consisted of incorporating the data using tools that detect this data on web pages and then storing it in a system based on the ONIX Standard (Needleman, 2001). ONIX is an open, international standard for the encoding and electronic exchange of bibliographic and commercial information in the publishing industry, with the participation of representatives of the commercial publishing chain from more than twenty countries (including Spain).

The second objective was the processing of the publications data considering the needs of different actors in the sector such as publishers, distributors, etc. In addition, this catalog will be self-verified in the sense that it will report incidences in the information stored in the catalog itself.

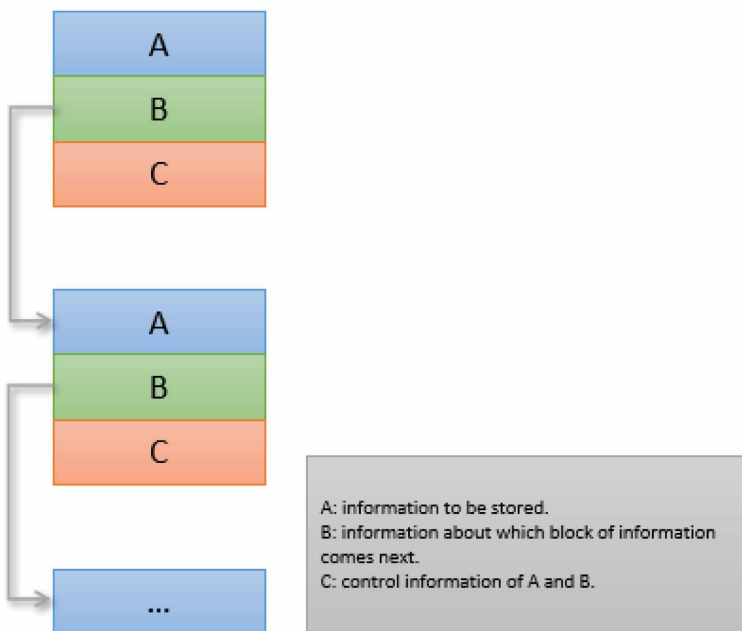
The third and final objective was to provide a record of the different operations carried out with the publications in a blockchain registry. This objective allows all transactions to be recorded without the possibility of changes or modifications, which ensures the veracity of the information and makes it possible, for example, to detect fraud or illicit transactions more easily. This third objective is the one most closely related to blockchain technology, whose fundamentals and application to this project are explored in more detail in the following section.

2.2. Blockchain Fundamentals

The origin of digital assets in 2008 with the appearance of Bitcoin also implied the appearance of recent technologies necessary to support these digital currencies and the operations that can be carried out with them. One of these technologies is well known by its English name: blockchain (Gramoli, 2022).

In a simplified way, blockchain technology consists of information that is completed with metainformation designed to guarantee the integrity of the information, so that it cannot be modified, and designed to maintain the time reference so that the temporal order of information generation can be precisely known. Figure 1 shows

Figure 1. How the blockchain works



Mechanism for the Systematic Generation of Functional Tests

an example of how blockchain technology works. The A blocks are containers of information (e.g., transactions made with publications), and the B blocks link the information so that it is all located and ordered temporally. The C blocks are calculated from the A and B blocks, so a change in the information (an A block), or in the sequence (a B block), would make the C block incorrect and the change would be immediately discovered.

In its first implementation, blockchain technology was used to store all Bitcoin transactions, i.e., who owns which coins. However, this technology quickly became independent of digital currencies and was applied to any area where it is necessary to store an immutable record of transactions, for example, biological samples, domain name registrations, public tenders, etc. Another key aspect of blockchain is server management. The blockchain chain, as seen in the example in Figure 1, must be stored on a computer with external communication.

On the other hand, blockchain technology works through smart contracts. This is a piece of software whose mission is to fulfill and enforce agreements usually registered between two or more parties, for example, to validate the change of ownership of a digital asset. Typically, smart contracts (Figure 2) are used to automate a blockchain system, i.e., the storage of information in a blockchain system is controlled by compliance with the rules and decisions indicated in a smart contract. In the same way, a blockchain system serves as a record of all deployed smart contracts.

Figure 2. Smart contract in blockchain



Although the blockchain itself guarantees that the information is reliable (as we have seen), if the server is not well managed, or suffers physical problems, it can compromise the stored information. To avoid this problem, the non-profit association Alastria exists in Spain to set up blockchain servers.

3. LITERATURE REVIEW

This section presents the state-of-the-art survey of research papers in the context of the development lifecycle of smart contracts in the blockchain. This review focused on the analysis of primary studies addressing some of the phases of the development lifecycle and/or model-driven engineering or other best practices for designing, developing, and testing smart contracts. For this purpose, the SLR (Systematic Literature Review) method proposed by Kitchenham (Kitchenham, 2013), which is one of the most widely applied methods in the field of software engineering, was used. This method proposes three main phases to execute a systematic review: planning the systematic review (planning), which defines aspects such as the need for the research, review protocol and research questions; execution of the review protocol (conducting), where the established protocol is carried out; and presentation of the results obtained (reporting), which presents the final analysis to answer each research question. These phases are described in detail below.

3.1. Planning Review

During this stage of the process, the need to conduct this literature review, the identification of research questions and the definition of the review protocol are established. On the one hand, regarding the need to conduct the review, in recent years, many studies have been published to evaluate and identify current challenges in the application of blockchain technology and smart contracts. Some of these research activities aimed to evaluate the use of blockchain in multiple sectors such as, supply chain (Pranto, 2019), (Hidayanto et al, 2019), education sector (Steiu, 2020), agriculture sector (Yadav, 2019) or healthcare sector (Yaqoob, 2021). Other authors have even published studies partially related to our SLR proposal. For example, Alharby et al. (Alharby, 2018) presented a systematic mapping of smart contract technology, selecting and classifying 188 relevant articles. In this classification, the lack of validation mechanisms for smart contracts is evident. Macrinici et al. (Macrinici, 2018) also conducted a systematic mapping, but, in this case, to identify the application of smart contracts and offer a perspective on current issues. Specifically, the authors presented research trends within this context and gathered sixty-four articles. The work of these authors concluded by indicating that, since 2016, there has been an increasing trend towards the publication of articles related to smart contracts and that the most discussed problems and solutions in the literature were related to security, privacy, and scalability of the blockchain and quality of smart contracts. Dhaiouir et al. (Dhailouir, 2020) also presented a systematic review of smart contracts, focusing on platforms, languages or applications and selection criteria. Specifically, this study indicates that smart contracts are being adopted

Mechanism for the Systematic Generation of Functional Tests

in several types of projects, but that they still face many challenges and technical problems, but these authors do not study validation and verification aspects.

In this context, the need to study current methods and techniques that allow quality assurance in the development of smart contracts is identified. Specifically, to analyze techniques for formal modeling of smart contracts, automatic generation of functional tests and/or code from such modeling, in order to characterize and present the state of the art in this field and to identify possible gaps and opportunities for further research. For this purpose, the following research questions (RQ) were proposed:

RQ1: Are there approaches in the literature that promote the application of a Software Development Life Cycle (SDLC)? What phases of the life cycle do the different studies promote? The motivation of this RQ is to find proposals that have been published and to identify their general contexts and the objectives they achieved using SDLC, all in the context of blockchain smart contracts.

RQ2: Do they promote model-based software engineering, early starting of the testing phase or automatic source code generation? The purpose of this RQ is to identify the techniques and guidelines applied in the different proposals, all in the context of blockchain smart contract.

On the other hand, once our research questions were established, inclusion/exclusion criteria were established to filter the primary studies found in some of the main digital libraries, as recommended by authors such as Ngai (Ngai, 2011). In this sense, the libraries selected were ACM Digital Library, IEEE Xplore Digital Library, ScienceDirect, Elsevier's Scopus and Springer Link. In our case, this strategy focused on locating articles published in peer-reviewed journals, presented at relevant conferences, and was done in two steps: (1) the keywords to be used in the search protocol were defined; and (2) preliminary searches were performed to refine the set of keywords and select the most appropriate ones in order to improve the quality of the results. Finally, the keywords systematically applied in each digital library were the following: (Engineering OR Semantic OR Model-based) AND (Requirement OR Analysis OR Validation OR Verification OR Check OR Testing) AND (Blockchain OR Smart Contract).

Regarding the exclusion/inclusion criteria, these were rigorously applied considering five phases as shown in Table 1. Moreover, only articles written in English and published in journals indexed in Journal Citation Reports (JCR) or prestigious conferences (i.e., conference level A*, A, B and C categorized in CORE Conference Rank) were considered. In addition, it was decided to exclude surveys, discussions, reviews, or opinion studies related to the subject matter sought. Finally, following the recommendations given in Kitchenham's method, the SLR protocol was reviewed by an external researcher to obtain a comprehensive review process.

Table 1. Exclusion/inclusion criteria by phase

Phase	Relevance analysis phase description
Ph1	Automatic search was conducted in each scientific database.
Ph2	English only; year of publication greater than or equal to 2016, because after analyzing numerous papers from other years, only from 2016 onwards did we start to identify articles that enhanced the predefined search criteria; full text obtained. Papers not related to the subject were excluded. This exclusion phase included the elimination of duplicate papers and the reading of the title and abstract of the work. In case of any doubt about any document, that document would be preliminarily included. The final decision would be considered and evaluated in the next phase.
Ph3	No new exclusion / inclusion criteria were applied (first meeting), but relevant papers were included. In this phase the researchers also analyzed all “doubtful” papers in detail, considering all their content.
Ph4	In this phase the «snowball» technique was applied, and it was therefore necessary to re-apply the P2 criteria.
Ph5	In this phase (second meeting) no new exclusion / inclusion criteria were applied, but the researchers analyzed all the “doubtful” papers in detail, considering all their content.

In this sense, a Professor of Software Engineering from the University of Seville (Spain) participated as an external expert to validate our review protocol.

3.2. Conducting and Report Review

The aim of this phase is to present the primary papers obtained after applying the search described in the previous section. Table 2 shows the primary papers obtained after applying the inclusion/exclusion criteria set out in the previous section.

Table 2. Primary studies

Data base	Ph1	Ph2	Ph3	Ph4	Ph5
ACM Digital Library	27	6	2	-	-
IEEE Xplore	39	7	3	-	-
ScienceDirect	372	31	7	-	-
Elsevier’s Scopus	352	42	6	-	-
SpringerLink	243	24	4	-	-
Snowball technique	-	-	-	10	3
Subtotals	1.033	110	22	10	3
Total	25				

Mechanism for the Systematic Generation of Functional Tests

After applying the search protocol and review phases (Table 1), twenty-five primary studies have been identified as the sum of the results of the third and fifth phase of the review protocol. Finally, Table 3 summarizes all the primary papers identified and analyzed, following all the criteria set out in the previous two sections.

Table 3. Summary of studies that have been analyzed

PS	Authors	Title	Year
PS01	Marchesi et al.	An Agile Software Engineering Method to Design Blockchain Applications (Marchesi et al., 2018)	2018
PS02	Liu, et al.	Applying Design Patterns in Smart Contracts (Liu et al., 2018)	2018
PS03	Choudhuret al.	Auto-Generation of Smart Contracts from Domain-Specific Ontologies and Semantic Rules (Choudhuret et al., 2018)	2018
PS04	Tateishi, et al.	Automatic smart contract generation using controlled natural language and template (Tateishi et al., 2019)	2019
PS05	Tsai et al.	Beagle: A New Framework for Smart Contracts Taking Account of Law (Tsai et al. 2019)	2019
PS06	Koul, R.	Blockchain Oriented Software Testing - Challenges and Approaches (Koul, R., 2018)	2018
PS07	Dolgui et al.	Blockchain-oriented dynamic modelling of smart contract design and execution in the supply chain (Dolgui et al., 2019)	2019
PS08	Porru et al.	Blockchain-Oriented Software Engineering: Challenges and New Directions (Porru et al., 2017)	2017
PS09	Shishkin, E.	Debugging Smart Contract's Business Logic Using Symbolic Model-Checking (Shishkin, 2018)	2018
PS10	Mavridou, A. et al.	Designing Secure Ethereum Smart Contracts: A Finite State Machine Based Approach (Mavridou et al. 2018)	2018
PS11	Parizi, et al.	Empirical vulnerability analysis of automated smart contracts security testing on blockchains (Parizi et al., 2018)	2018
PS12	Lee et al.	Formal Specification Technique in Smart Contract Verification (Lee et al., 2019)	2019
PS13	Mavridou, et al.	FSolidM for Designing Secure Ethereum Smart Contracts: Tool Demonstration (Mavridou, et al., 2018)	2018
PS14	Sillaber et al.	Life Cycle of Smart Contracts in Blockchain Ecosystems (Sillaber et al., 2017)	2017
PS15	Grigg, I.	On the intersection of Ricardian and Smart Contracts (Grigg 2015)	2015
PS16	Kruijff et al.	Ontologies for Commitment-Based Smart Contracts (Kruijff et al., 2017)	2017
PS17	Clack	Smart Contract Templates: Legal semantics and code validation (Clack, 2018)	2018
PS18	Clack et al.	Smart Contract Templates: Foundations, design landscape and research directions (Clack et al., 2016)	2016
PS19	Syahputra et al.	The Development of Smart Contracts for Heterogeneous Blockchains (Syahputra et al., 2019)	2019
PS20	Liao et al.	Toward A Service Platform for Developing Smart Contracts on Blockchain in BDD and TDD Styles (Liao et al., 2017)	2017
PS21	Al Khalil, et al.	Trust in Smart Contracts is a Process, As Well (Al Khalil, et al., 2017)	2017
PS22	Mavridou et al	VeriSolid: Correct-by-Design Smart Contracts for Ethereum (Mavridou et al, 2019)	2019
PS23	Permenev et al.	VerX: Safety Verification of Smart Contracts (Permenev et al., 2019)	2019
PS24	Mao et al.	Visual and User-Defined Smart Contract Designing System Based on Automatic Coding (Mao et al., 2019)	2019
PS25	Clack et al	Smart Contract Templates: essential requirements and design options (Clack et al, 2016)	2016

RQ1: Are there approaches in the literature that promote the application of a Software Development Life Cycle (SDLC)? What phases of the life cycle do the different studies promote?

After analyzing the primary studies, the phases of the software development life cycle that have been most addressed by the authors were: (A1) Requirements, analysis, or design phase (68%), (A2) Coding phase (40%), (A3) Testing phase (28%) and (A4) Other phases (12%).

The work of Marchesi et al (PS01) and Tsai et al (PS05) stands out. Study PS01 proposes a software development process to elicit requirements, analyze, design, develop, test and implement blockchain applications and study PS05 proposes a framework with five stages: development of smart contract templates, from domain analysis, formal model of smart contracts, code development from templates, verification and validation.

It seems, therefore, that some efforts of the scientific community are currently directed towards implementing some kind of development lifecycle. However, in the context of the blockchain, the analyzed processes consist only of a certain number of unlinked phases, as they are not arranged in a clear order of precedence and the inputs/outputs of each stage are also not clearly defined.

It is important to highlight, due to the relevance it has in the blockchain methodology, the fact that the phase with the least impact in the identified literature is the software testing phase. From our point of view, blockchain applications differ quite a bit from other traditional applications, since once a smart contract is implemented, its execution cannot be reversed. Therefore, robust testing is essential, with an emphasis on requirements elicitation, verification and validation, and code debugging. Moreover, testing should involve the simulation of all possible expected and unexpected variables for each smart contract and for the triggers that execute the transactions.

RQ2: Do they promote model-based software engineering, early starting of the testing phase or automatic source code generation?

In recent years, the use of modeling tools or CASE tools, as well as the use of the UML standard have helped to document the functionality of business processes and to use transformations between models. This has made it possible to automate code generation in many cases. For example, among the selected studies, Marchesi et al. (PS01) and Syahputra et al. (PS19) propose the use of UML diagrams to describe application requirements, which makes it possible to start testing at an early stage of system development (early testing). In this sense, performing model-based software engineering is important, as it provides the following advantages (Pohl, 2012): it is

Mechanism for the Systematic Generation of Functional Tests

possible to implement best practices and generate well-tested code, which reduces the occurrence of vulnerable code; software code is more difficult to understand than models, which makes it easier to test the correctness of a model; and it is possible to apply model-based engineering on multiple platforms.

In this context, the proposals addressed in the primary studies are made by applying different approaches: (B1) Application of model-based software engineering; (B2) Promotion of early testing; and (B3) Proposal of automatic code generation. However, it is possible to observe that although early testing helps to reduce the number of defects, it seems that the efforts of the scientific community are not directed towards this approach. Nevertheless, some authors such as Koul et al. (PS06) highlight the need to ensure software quality from early stages, indicating the challenges currently faced by the testing of this type of applications. These authors also recognize the need to design specific tools and techniques for testing this type of software, in order to ensure high quality standards in the development of smart contracts, achieving greater reliability and lower development costs.

Regarding the automatic generation of smart contracts, an important aspect to consider is the technique that the primary studies have used. The automatic generation of the smart contract code using a model-based software engineering process would eliminate the manual effort required in coding from design and, therefore, speed up the process, while decreasing the possibility of errors compared to the manual coding of the requirements or models. In this sense, the techniques most commonly used or proposed by the authors are: (C1) Generation using ontologies and/or domain-specific semantic rules; (C2) Generation using model-based engineering; and (C3) Generation through templates or other utilities. Interestingly, the study by Syahputra et al. (PS19) proposes the use of a smart contract platform to generate smart contracts for heterogeneous blockchain technologies using UML and OCL (Object Constraint Language).

In summary, after analyzing the primary studies found, it is possible to observe that the phases of requirements specification and software testing are among the aspects least addressed by the research community. However, Marchesi et al. (PS01) proposes a software development process considering the typical phases of the software development life cycle, but they focus on the application of Agile methodologies. In their study they propose the use of UML diagrams to describe the design of the applications and even provide a modeling of the interactions between the traditional software and the blockchain environment. Other authors such as Syahputra et al. (PS05) discuss the development process from a smart contract platform. This platform aims to create a smart contract for heterogeneous blockchain technologies, and they propose the use of UML, in addition to OCL, for the design.

All primary studies, in one way or another, indicate the need to obtain well-functioning software. However, more emphasis needs to be placed on functional, security and performance testing in the case of smart contracts due to its critical factor in ensuring the reliability of blockchain networks. In this sense, some authors such as Koul et al. (PS06) highlight the need to ensure software quality from early stages. Therefore, this study partially coincides with our approach of obtaining test cases in early stages of the smart contract development lifecycle. Furthermore, these authors recognize the need to design specific tools and techniques for testing this type of software, to ensure high quality standards.

Finally, several papers stand out especially due to their proposed verification and testing of smart contracts and blockchain applications:

- Marchesi et al. (PS01) proposes a software development process that allows gathering requirements, analyzing, designing, developing, testing, and implementing blockchain applications. The process is based on Agile practices, using user stories and iterative and incremental development based on them.
- Choudhury et al. (PS03) provides a framework for the automatic generation of smart contracts. This framework uses ontologies and semantic rules to encode domain-specific knowledge and then leverages the structure of abstract syntax trees to incorporate the required constraints.
- Tateishi et al. (PS04) proposes a technique to automatically generate a smart contract from a human-understandable contract document. Specifically, this is created using a template and a controlled natural language. The automation is based on a mapping of the template and that natural language to a formal model that can define the terms and conditions of a contract, including temporal constraints and procedures.
- Mavridou et al. (PS13) argue that, in practice, smart contracts are plagued with vulnerabilities. To facilitate the development of secure smart contracts, these researchers have created a framework that allows contracts to be defined as Finite State Machines (FSM) with rigorous and clear semantics.
- Syahputra et al. (PS19) address a discussion on how the development process of a smart contract platform that aims to generate smart contracts for heterogeneous blockchain technologies should look like.
- Mavrodou et al. (PS22) present a framework for the formal verification of smart contracts using a model based on a transition system with operational semantics and allows the generation of Solidity code from the verified models, which would enable the development from the design of smart contracts.

4. PROPOSED SOLUTION

The objective of this section is to present a model-driven approach to generate functional test plans from smart contract specifications (Section 4.1). After describing our proposal, Section 4.2 describes a validation case on a real business project, the SmartISBN project to solve the challenges described in Section 2.1. To this end, Section 4.2.1 describes the proposed life cycle for managing the production and distribution chain process of a publication. Once this life cycle has been defined, Section 4.2.2 describes, in general terms, the architecture of the SmartISBN platform that supports the proposed life cycle. Finally, Section 4.2.3 explains how the functional tests necessary to validate the smart contracts associated with the proposed publication distribution process were systematically generated.

4.1. Proposal For Systematic Functional Test Generation in Blockchain Environments

The proposal presented in this article for the systematic generation of functional tests of smart contracts in blockchain is based on the principles of early testing, in such a way that it is possible to generate functional test plans based on the specifications of the smart contracts, independently of the blockchain platform used.

To achieve this purpose, our proposal is based on the model-driven engineering paradigm (Bézivin, 2004). Specifically, it is based on: (1) the design of a metamodel containing the definition of all the concepts needed to model smart contracts from functional specifications; and (2) the design of systematic mechanisms to generate functional test plans from the smart contract models designed according to the aforementioned metamodel. Both aspects of the proposal are described below.

On the one hand, Figure 3 shows our proposed smart contract metamodel. This metamodel is based on the following pillars (see Figure 4): (a) a set of legal relations (Legal relation) between stakeholders; (b) Stakeholders (interested parties) that could be considered as a person, an organization or any other entity capable of entering into a legal agreement; (c) a set of internal and external data sources, from which the smart contract is nourished; (d) a set of actions (or behaviors), which are composed of activities and operations on the input data and which are applied on the different business rules of the smart contract; and (e) a set of constraints, which allow controlling the consistency of the smart contract automatically and autonomously during its execution. It is also important to mention that the constraints model the terms and conditions of the smart contract, imposing restrictions as to when an action can be performed, whether the circumstances allow the action to be performed, and so on. Thus, in a smart contract model, a constraint links the execution of an action to the

Figure 3. Smart contract metamodel proposal

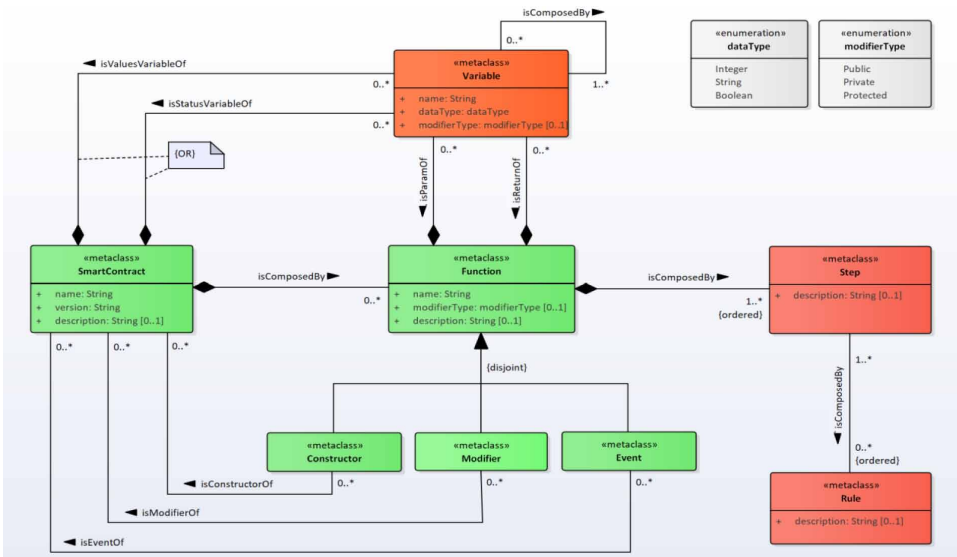
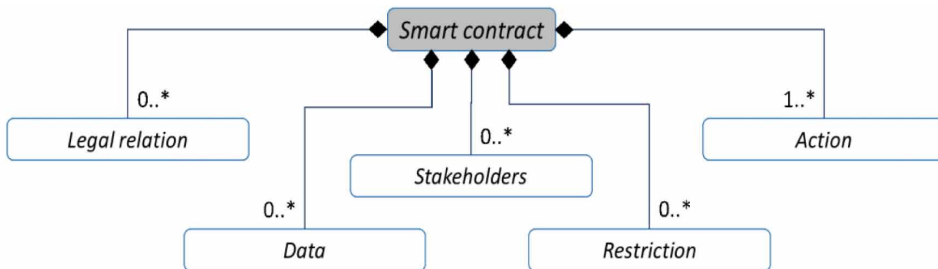


Figure 4. Pillars of the smart contract metamodel



fulfillment of additional conditions and rules. A constraint can affect one or more actions and, in addition, they can read from possible data sources.

On the other hand, once the smart contract metamodel and all its entities have been instantiated, systematic mechanisms are proposed to generate scenarios and functional test cases.

In this sense, the systematic generation of functional tests is done in two stages. First, the skeleton of all test scenarios is generated from each Smart Contract in the model and, then the test case casuistry is generated by combining the test scenario data. Specifically: (1) for each Function of a smart contract, in conjunction with the input data, a test case is created; (2) for each Function Step of a smart contract,

a test case step is created; and (3) for each Function Step Restriction of a smart contract, test restrictions are created.

4.2. Validation Case: SmartISBN Project

This section describes the validation context provided by the SmartISBN project. To do so, it first introduces the life cycle associated with the process of the production and distribution chain of a digital publication proposed in the framework of the project. Next, the technological and functional architecture of the SmartISBN platform, which supports the proposed distribution process, is described. Finally, the section presents how the theoretical proposal described in Section 4.1 has been applied to systematically generate the functional tests from the specification of the smart contract that governs SmartISBN.

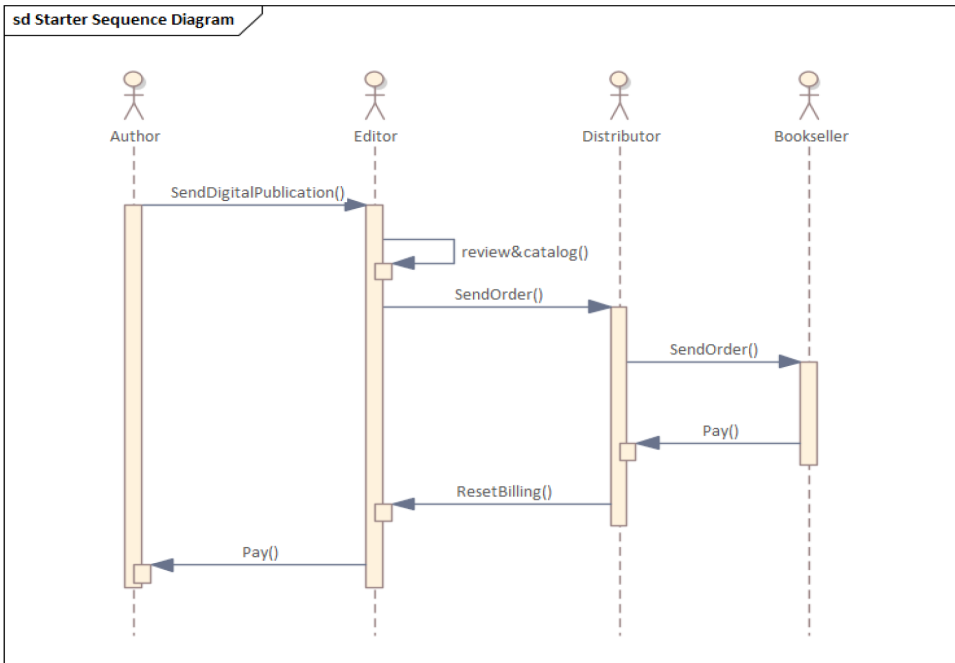
4.2.1. Proposed Life Cycle of the Production and Distribution Chain of a Digital Publication

As a preliminary step to the design of the SmartISBN technological solution, within the framework of the project, the general process of the life cycle of a publication from the point of view of the production and distribution chain was conceptually proposed. In this sense, Figure 5 represents the distinct stages of this life cycle, as well as the different actors involved in each stage. For this purpose, the UML (Unified Modelling Language) sequence diagram notation (Fontela, 2012) is used to represent the communication flow between the stages described above.

Initially, the distribution process could be considered to begin with the first stage of “E1. Conception and drafting of the publication”, in which the Author gives shape, consistency, and meaning to its content until the final manuscript is obtained. Then, the Author would initiate the second stage of the life cycle: “E2. Editorial processing of the publication”. In this stage, the Editor receives the manuscript and carries out its review process, cataloging the publication within its editorial line and identifying metadata. Once this processing is completed, the Publisher would initiate the stage “E3. Printing and distribution”, establishing different contracts or orders with the Distribution company so that the latter can begin the physical printing and/or digital dissemination of the different editions of the publication. Finally, the life cycle would end with the “E4. Acquisition of copies of the publication” stage, in which Bookshops (or other points of sale) would establish contracts and orders for the publications under distribution.

Considering the above process, it is worth noting that during the transitions between the distinct stages, payments, purchase orders, sales orders, etc., take place between

Figure 5. Life cycle of a publication's overall production and distribution process



the different actors involved in the process. In this sense, it is crucial to maintain the traceability of all these transactions throughout the entire supply chain process.

4.2.2. SmartISBN Platform Architecture

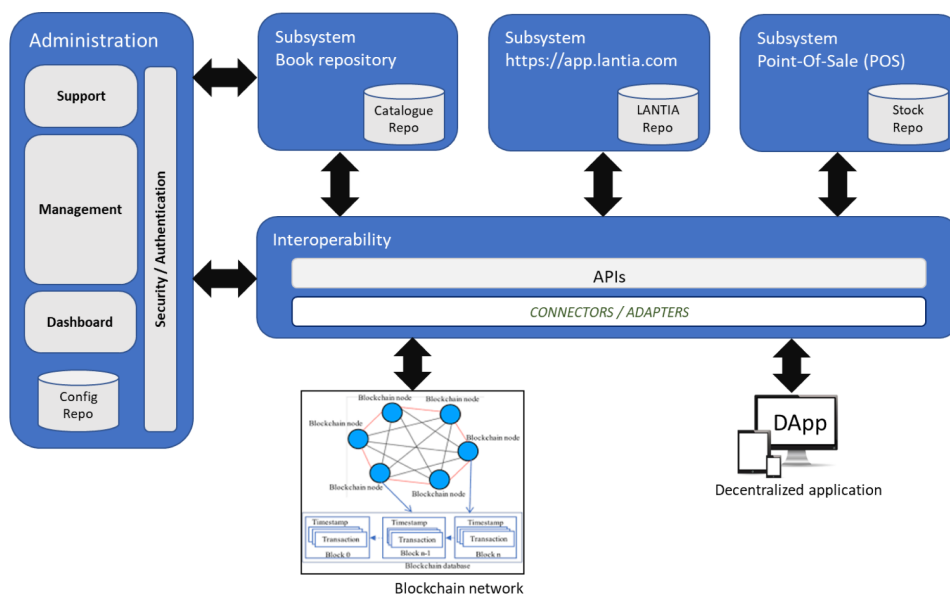
To meet the objectives of the SmartISBN project and to support the life cycle of the publication's distribution process, a technological architecture is proposed with the subsystems shown in Figure 6.

On one hand, the platform incorporates an administration subsystem so that users with this role can manage users, roles, and access permissions to the platform, as well as control the status of the platform through dashboard utilities.

One of the main objectives of the SmartISBN project was to allow publishers to catalog works correctly within the platform so that users could carry out advanced searches and even receive recommendations based on their previous purchases. The cataloging subsystem is responsible for automating this cataloging process by analyzing the metadata of the digital application, based on the international standard ONIX (XML). However, as a prior step to this automatic cataloging process, the user with the role of Editor must incorporate in the platform, at least, the ISBN

Mechanism for the Systematic Generation of Functional Tests

Figure 6. SmartISBN platform architecture



(International Standard Book Number) metadata. Based on this information, the SmartISBN platform includes automatic functionalities to consult the rest of the metadata of the digital publication by consulting public bibliographic sources. SmartISBN is currently integrated with Amazon, Google Book, La Casa del Libro, Todos tus libros and Editorial Lantia, among others.

On the other hand, the SmartISBN platform includes a frontend subsystem and a point-of-sale terminal subsystem, which manages, respectively, the repository of digital publications and their inventory and stock, together with payments and the different order and sales orders. These subsystems will be directly accessible by users with the role of Distributor and Bookshop.

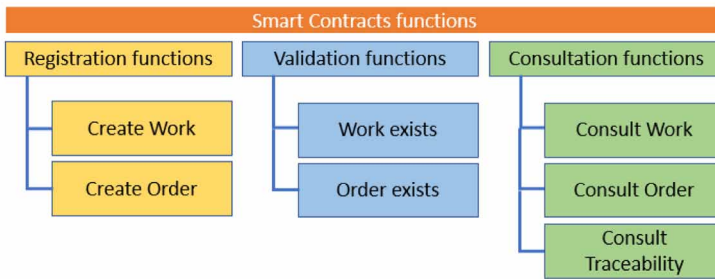
To control the traceability of all order, sales, and distribution orders, the SmartISBN platform includes integration with the Ethereum platform and the use of the Solidity programming language (for the implementation of smart contracts). As part of the SmartISBN project, an Ethereum virtual machine was deployed, and its platform was used to manage the traceability of order and sales transactions in the distribution process of a publication.

Finally, the SmartISBN platform includes an integration subsystem that provides the different communication APIs (Application Programming Interface) to allow the flow of information and data between the different subsystems described above.

4.2.3. Applying the Functional Test Generation Approach in SmartISBN

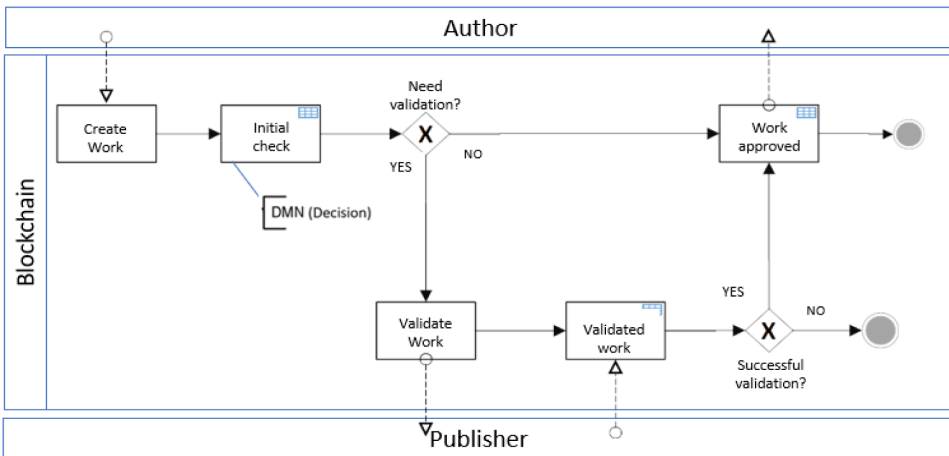
To control the consistency and integrity of transactions in the process of managing the production and distribution chain of a digital publication, it was necessary to implement smart contracts with various functions (see Figure 7), business rules and restrictions.

Figure 7. Smart Contract Functions



Due to space limitations, it is not possible to describe the complete functional test generation casuistry of this functionality but, as an example of application, we will focus on the following activity diagram. The diagram in Figure 8 shows the expected behavior of the smart contract and specifically the “Create Work” functionality, as well as the rules and constraints to be considered at each step.

Figure 8. Functionality of Smart Contract



assets. The paper presents how blockchain technology is suitable for this purpose and analyzes the challenges it poses. Specifically, it presents the mechanisms that have been used to generate the functional tests that have facilitated the communication between the experts and the technical team to validate the results of the project. Other mechanisms have been developed in SmartISBN to facilitate this communication in other phases, such as in the requirements identification phase. The results in Section 3 indicate that we have not found any proposal that contemplates formal modelling of contracts and automated generation of artefacts from these models. Marchesi et al. presents a complete process but does not include support for generating artefacts automatically. Choudhury et al., Tateishi et al., Mavridou et al. and Mavrodou et al. describe automations for generating or verifying smart contracts, but none of them include requirements artefact management or test artefact generation.

In future work, we plan to improve our communication protocols to generalize them, as well as to enable mechanisms that allow us to automatically generate smart contract code. In fact, we are currently working on another international project that will allow us to make progress on this. In the context of software testing, our idea is to improve test prioritisation mechanisms, not just generation. The idea would be that the technical team could not only generate the functional tests from the requirements, guaranteeing their correspondence with them, but also prioritize them so that, in the event of a lack of resources, the tests could be generated according to the established prioritization.

ACKNOWLEDGMENT

This research article has been elaborated within the following projects: EQUAVEL Project (PID2022-137646OB-C31), which was funded by the Ministry of Economy and Competitiveness of the Government of Spain; and SmartISBN, a technology transfer project, which was funded by the company Lantia Publishing S.L.

REFERENCES

- Al Khalil, F., Butler, T., O'Brien, L., & Ceci, M. (2017). Trust in smart contracts is a process, as well. In *Financial Cryptography and Data Security: FC 2017 International Workshops, WAHC, BITCOIN, VOTING, WTSC, and TA*, (pp. 510-519). Springer International Publishing. doi:10.1007/978-3-319-70278-0_32
- Alharby, M., Aldweesh, A., & van Moorsel, A. (2018). blockchain-based smart contracts: A systematic mapping study of academic research (2018). In *2018 International Conference on Cloud Computing, Big Data and blockchain (ICCB)* (pp. 16). IEEE. 10.1109/ICCB.2018.8756390

Mechanism for the Systematic Generation of Functional Tests

Bézivin, J. (2004). In search of a basic principle for model driven engineering. *Novatica Journal, Special Issue*, 5(2), 2124.

Choudhury, O., Rudolph, N., Sylla, I., Fairoza, N., & Das, A. (2018). Auto-Generation of Smart Contracts from Domain-Specific Ontologies and Semantic Rules. *Conference: IEEE Conferences on Internet of Things, Green Computing and Communications, Cyber, Physical and Social Computing, Smart Data, Blockchain, Computer and Information Technology*. IEEE. 10.1109/Cybermatics_2018.2018.00183

Clack, C.D. (2018). Smart Contract Templates: Legal semantics and code validation. *Journal of Digital Banking*, 2(4), 338-352.

Clack, C. D., Bakshi, V. A., & Braine, L. (2016). *Smart Contract Templates: foundations, design landscape and research directions*. arXiv. <https://arxiv.org/pdf/1608.00771.pdf>

Clack, C. D., Bakshi, V. A., & Braine, L. (2016). “Smart Contract Templates: essential requirements and design options”. <https://arxiv.org/pdf/1612.04496.pdf>

Dhaiouir, S., & Assar, S. (2020). A systematic literature review of blockchain-enabled smart contracts: platforms, languages, consensus, applications and choice criteria. In *International Conference on Research Challenges in Information Science* (pp. 249-266). Springer, Cham. 10.1007/978-3-030-50316-1_15

Dolgui, A., Ivanov, D., Potryasaev, S., Sokolov, B., Ivanova, M., & Werner, F. (2020). Blockchain-oriented dynamic modelling of smart contract design and execution in the supply chain. *International Journal of Production Research*, 58(7), 2184–2199. doi:10.1080/00207543.2019.1627439

Fontela, C. (2012). *UML: modelado de software para profesionales*. Alpha Editorial.

Gramoli, V. (2022). Blockchain Fundamentals. In *Blockchain Scalability and its Foundations in Distributed Systems* (p. 1739). Springer International Publishing. doi:10.1007/978-3-031-12578-2_3

Grigg, I. (2015). *On the intersection of Ricardian and Smart Contracts*. IANG. https://iang.org/papers/intersection_ricardian_smart.html.

Hidayanto, A. N., & Prabowo, H. (2019). The latest adoption blockchain technology in supply chain management: A systematic literature review. *ICIC Express Letters*, 13(10), 913–920.

Janssens, L., Bazhenova, E., De Smedt, J., Vanthienen, J., & Denecker, M. (2016, June). Consistent Integration of Decision (DMN) and Process (BPMN) Models. In CAiSE forum (Vol. 1612, pp. 121128).

- Kitchenham, B., & Brereton, P. (2013). A systematic review of systematic review process research in software engineering. *Information and Software Technology*, 55(12), 2049–2075. doi:10.1016/j.infsof.2013.07.010
- Koul, R. (2018). Blockchain Oriented Software Testing - Challenges and Approaches. *3rd International Conference for Convergence in Technology (I2CT)*, Pune, India. 10.1109/I2CT.2018.8529728
- Kruijff, J., & Weigand, H. (2017). *Ontologies for Commitment-Based Smart Contracts*. OTM 2017 Conferences: Confederated International Conferences: CoopIS, C&TC, and ODBASE 2017, Rhodes, Greece.
- Lee, S., Park, S., & Park, Y. B. (2019). Formal Specification Technique in Smart Contract Verification. *6th International Conference on Platform Technology and Service (PlatCon)*. IEEE. 10.1109/PlatCon.2019.8669419
- Legerén-Molina, A. (2018). Los contratos inteligentes en España (La disciplina de los smart contracts) / Smart contracts in Spain; the regulation of smart contracts. *Revista de Derecho civil*, 5(2), 193-241.
- Liao, C., Cheng, C., Chen, K., Lai, C., Chiu, T., & Wu-Lee, C. (2017). Toward A Service Platform for Developing Smart Contracts on Blockchain in BDD and TDD Styles. *2017 IEEE 10th International Conference on Service-Oriented Computing and Applications*. IEEE. 10.1109/SOCA.2017.26
- Liu, Y., Lu, Q., Xu, X., Zhu, L., & Yao, H. (2018). *Applying Design Patterns in Smart Contracts*. Springer International.
- Luu, L., Chu, D. H., Olickel, H., Saxena, P., & Hobor, A. (2016). Making smart contracts smarter. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security* (pp. 254-269). ACM. 10.1145/2976749.2978309
- Macrinici, D., Cartofeanu, C., & Gao, S. (2018). Smart contract applications within blockchain technology: A systematic mapping study. *Telematics and Informatics*, 35(8), 2337–2354. doi:10.1016/j.tele.2018.10.004
- Magadán-Díaz, M. y Rivas-García, J.I. (2020). *La industria editorial española: dos décadas clave de transformación y cambio (19962016)*. Investigaciones de Historia Económica Economic History Research doi:10.33231/j.ihe.2020.04.003
- Mao, D., Wang, F., Wang, Y., & Hao, Z. (2019). *Visual and User-Defined Smart Contract Designing System Based on Automatic Coding*. IEEE Access. Digital Object Identifier., doi:10.1109/ACCESS.2019.2920776

Mechanism for the Systematic Generation of Functional Tests

- Marchesi, M., Marchesi, L., & Tonelli, R. (2018). An Agile Software Engineering Method to Design Blockchain Applications. *Software Engineering Conference Russia (SECR 2018)*, Moscow, Russia. 10.1145/3290621.3290627
- Martínez Alés, R. (2001). *Información Comercial Española, ICE: Revista de Economía*. Dialnet.
- Mavridou, A., & Laszka, A. (2018). Designing Secure Ethereum Smart Contracts: A Finite State Machine Based Approach. *International Conference on Financial Cryptography and Data Security*. FC 2018: Financial Cryptography and Data Security. Springer. 10.1007/978-3-662-58387-6_28
- Mavridou, A., & Laszka, A. (2018). FSolidM for Designing Secure Ethereum Smart Contracts: Tool Demonstration. *7th International Conference on Principles of Security and Trust (POST) Held as Part of the 21st European Joint Conferences on Theory and Practice of Software (ETAPS)*. Springer. 10.1007/978-3-319-89722-6_11
- Mavridou, A., Laszka, A., Stachtiari, E., & Dubey, A. (2019). VeriSolid: Correct-by-Design Smart Contracts for Ethereum. *Cryptography and Security; Software Engineering*. arXiv.org. arXiv:1901.01292
- Needleman, M. H. (2001). ONIX (online information exchange). *Serials Review*, 27(34), 102104.
- Ngai, E. W., Hu, Y., Wong, Y. H., Chen, Y., & Sun, X. (2011). The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature. *Decision Support Systems*, 50(3), 559–569. doi:10.1016/j.dss.2010.08.006
- Parizi, R. M., Dehghantanha, A., Choo, K. K. R., & Singh, A. (2018). Empirical vulnerability analysis of automated smart contracts security testing on blockchains. *CASCON. Proceedings of the 28th Annual International Conference on Computer Science and Software Engineering*. Springer.
- Permenev, A., Dimitrov, D., Tsankov, P., Drachsler-Cohen, D., & Vechev, M. (2019). “VerX: Safety Verification of Smart Contracts”. Safety verification of Smart Contracts. *Security and Privacy*, 2020.
- Pohl, K., Hönniger, H., Achatz, R., & Broy, M. (Eds.). (2012). *Model-based engineering of embedded systems: The SPES 2020 methodology*. Heidelberg: Springer.
- Porru, S., Pinn, A., Marchesi, M., & Tonelli, R. (2017). Blockchain-Oriented Software Engineering: Challenges and New Directions. *2017 IEEE/ACM 39th IEEE International Conference on Software Engineering Companion*. ACM. 10.1109/ICSE-C.2017.142

Pranto, S., Jardim, L., Oliveira, T., & Ruivo, P. (2019, October). Literature review on blockchain with focus on supply chain. In *Atas da Conferencia da Associacao Portuguesa de Sistemas de Informacao 2019*. Associação Portuguesa de Sistemas de Informação.

Shishkin, E. (2018). *Debugging Smart Contract's Business Logic Using Symbolic Model-Checking*. arXiv.org > cs > arXiv:1812.00619v1

Sillaber, C., & Walth, B. (2017). Life Cycle of Smart Contracts in Blockchain Ecosystems. *Datenschutz und Datensicherheit – DuD*, 41(8), 497–500.

Steu, M. F. (2020). blockchain in education: Opportunities, applications, and challenges. *First Monday*. doi:10.5210/fm.v25i9.10654

Syahputra, H., & Weigand, H. (2019). The Development of Smart Contracts for Heterogeneous Blockchains. *Enterprise Interoperability, VIII*, 229–238. doi:10.1007/978-3-030-13693-2_19

Tateishi, T., Yoshihama, S., Sato, N., Saito, S. (2019). Automatic smart contract generation using controlled natural language and template. *IBM Journal of Research and Development*, 63.

Tsai, W., Ge, N., Jiang, J., Feng, K., & He, J. (2019). Beagle: A New Framework for Smart Contracts Taking Account of Law. *IEEE International Conference on Service-Oriented System Engineering (SOSE)*. IEEE 10.1109/SOSE.2019.00028

Yadav, V. S., & Singh, A. R. (2019). A systematic literature review of blockchain technology in agriculture. In *Proceedings of the International Conference on Industrial Engineering and Operations Management* (pp. 973-981). Springer.

Yaqoob, I., Salah, K., Jayaraman, R., & Al-Hammadi, Y. (2021). blockchain for healthcare data management: Opportunities, challenges, and future recommendations. *Neural Computing & Applications*, 1–16.

KEY TERMS AND DEFINITIONS

Blockchain: It is a shared, immutable ledger that facilitates the process of recording transactions and tracking assets in a business network. An asset can be tangible (a house, car, cash, land) or intangible (intellectual property, patents, copyrights). Virtually anything of value can be tracked and traded on a blockchain network, reducing risk and cutting costs for all involved.

Mechanism for the Systematic Generation of Functional Tests

Digital Publishing Systems: This concept, also called digital publishing platform, allows creators to share, discover, and monetize digital magazines, catalogs and other publications with a global audience.

Digital Publishing: This concept, also called electronic or online publishing, is the distribution of a variety of online content, such as journals, magazines, newspapers, and eBooks. Through this process, any company or publisher can digitize documents and information that people can view online, download, sometimes manipulate, and even print out or share otherwise, if they choose.

Functional Tests: It is a type of software testing that validates the software system against the functional requirements/specifications. The purpose of Functional tests is to test each function of the software application, by providing appropriate input, verifying the output against the Functional requirements.

Smart Contracts: It is programs stored on a blockchain that run when predetermined conditions are met. They typically are used to automate the execution of an agreement so that all participants can be immediately certain of the outcome, without any intermediary's involvement or time loss.