TOWARDS THE APPROACH TO BUILDING
SMART CONTRACTS BASED ON
BUSINESS RULES USING NATURAL
LANGUAGE PROCESSING

Dmytro Shynkarenko
Master’s Student of the Educational and Scientific Institute of Computer Science and Information Technology
National Technical University “Kharkiv Polytechnic Institute”, Ukraine

Andrii Kopp
Ph.D., Associate Professor of the Department of Software Engineering and Management Intelligent Technology
National Technical University “Kharkiv Polytechnic Institute”, Ukraine

Summary. This paper considers the smart contracts development process based on business rules using natural language processing as the research object. The research subject includes software components for creating smart contracts based on business rules using natural language processing. The research aims to simplify the software component development for decentralized systems by using smart contracts generation from business rules written in natural language. This study considers smart contract development approaches and technologies, intelligent text processing methods, as well as software development techniques using the Python programming language for the experimental implementation of the proposed solution. This study outlines the relevance of this research, provides a state-of-the-art analysis, proposes the improved procedure of smart contracts’ development and deployment, and suggests an algorithm for smart contract generation based on business rules.

Keywords: business rule, blockchain, smart contract, natural language processing.

Introduction. Today many intelligent technologies have emerged as the most important sector of information technology. Artificial intelligence-based systems have found widespread application in a wide range of systems meant to accomplish a variety of functions. Intelligent information systems are one type of such intelligent system intended to facilitate human activity and information search.

However, the application of intelligent information systems was hampered by the need to analyze vast amounts of unstructured data that were inappropriate for computer comprehension. The transformation of data through smart contracts is the appropriate answer to this challenge.

A basic definition of knowledge is that it is a collection of real-world entities, their properties, and the relationships between them to reflect the real world better. On the other hand, the knowledge representation field is concerned with creating
models using which knowledge is presented in a way that is both acceptable for computer processing and understandable to humans.

In the functioning of intelligent information systems, knowledge representation is related to the use of different models for representing a set of statements based on which the system can make judgments, issue suggestions, etc. Therefore, knowledge representation is a branch of Artificial Intelligence (AI) concerned with how a system decides what to do based on what it “knows” [1].

Because an enormous volume of data is currently kept in a text form, information systems require Natural Language Processing (NLP) algorithms to extract and use knowledge. Natural language is without a doubt one of the essential tomorrow’s user interfaces. Even now, the services that computers can provide lag considerably behind the services that users receive. Users can do a lot more, however, there is currently an issue with integrating these powerful capabilities for use.

The knowledge that allows the computer and the user to successfully interact with entities present in natural language texts is the solution to this challenge. There are numerous types of knowledge representation approaches, such as ontologies, semantic networks, and frame models, but the most recent solution is “smart contracts”, which allow for the organization and presentation of unstructured text data in the form of knowledge [2].

The following are the primary benefits of smart contracts, which are:

- secure – the smart contract is encrypted and saved on many devices, ensuring that it is not lost or altered without permission;
- cheap and quick – most operations are automated, and most intermediaries are removed;
- standardized – there are several smart contract solutions available nowadays, and users or developers can select the best one for a certain task.

However, smart contracts have several drawbacks:

- human aspect – because the contract is a code developed by people, it may have flaws;
- immutability – because smart contracts are recorded in the blockchain, they cannot be amended.

NLP is the technique of unlocking knowledge expressed using natural language, which may then be used to update and check for consistency against a knowledge graph, which can then be queried and generated using natural language. These two technologies are complementary and improve each other.

Hence, this paper is devoted to research and development of software components for developing smart contracts based on business rules using natural language text processing and knowledge graphs.

**Ethereum blockchain platform.** Today, decentralized systems are frequently developed on the Ethereum public blockchain core. Public Ethereum employs Proof-of-Work (PoW) consensus, whereas its several test networks employ various forms of Proof-of-Authority (PoA) and Proof-of-Stake (PoS) consensus. Such systems meet the strictest security criteria, have been tested in dozens of real-world networks, and are developed for establishing blockchains with various consensus mechanisms and full-fledged, multifunctional smart contracts [3].
It is important to recognize the importance of the POA Network project, whose developers have done an excellent job in launching multiple fast and stable networks. The POA Network is faster than the original Ethereum, but it also has the same stability and versatility for completing any transaction, and the duty of validators is done by computers whose meaningful work is lawfully recognized. This network can be regarded as an environment for launching business Ethereum blockchains [3].

The POA Network makes use of the Ethereum Virtual Machine (EVM) and smart contracts implemented in the Solidity programming language. With a vast quantity of ready-made code and development templates, EVM has long been the standard for virtual machines. The EVM smart contracts code is in charge of enormous sums of cryptocurrency, and any weakness discovered results in a strong community and media response, hence EVM smart contracts security is now highly high [4].

**EOS blockchain platform.** In terms of performance and safety promises, the EOS platform is the second engine. EOS can be launched as a standalone network with PoS or PoA options. This platform, like Ethereum, has already been tested in reality, and it offers strong security and functionality, allowing users to operate blockchain with smart contracts to automate any transactions.

EOS, unlike Ethereum, immediately employs a hierarchical system of accounts and rights to specific acts. All of this makes EOS look like an operating system – “a program that runs other programs”. EOS provides an immediate account management system and quick consensus, as well as the ability to easily incorporate practically any application using C++ extensions and smart contracts written in C++ or WebAssembly (for example, users can add third-party cryptography techniques).

Consensus design in EOS and fast block confirmation enables a very fast response time to the user, which is critical when developing decentralized applications with sophisticated functionality. For example, Cyberway recently completed an incredibly complex migration of all business logic from the old blockchain transparently for users, demonstrating EOS's adaptability and agility [5].

**Related work.** The rapid expansion of the Internet, as well as new options for processing, sharing, and compatible reuse of enormous quantities of information for solving various problems in numerous subject areas, aided in the introduction and development of new types of systems built to operate in the Web environment. These intelligent information systems are built using Semantic Web technologies and the ontological approach to knowledge organization and application [6].

Recently, this technique has grown in relevance and popularity. A Knowledge Graph is a semantic graph made up of vertices (or nodes) and edges. Vertices are used to represent concepts or entities. This term refers to broad categories of objects. A physical item in the real world, such as a person, location, or organization, is referred to as an “entity”. Edges represent conceptual or object-to-concept linkages. Using knowledge graphs, fragmented, i.e., partially observed objects and concepts can be joined together to form a complete and structured knowledge repository that facilitates information content management, retrieval, usage, and understanding [6].

The paper [7] focuses on the creation of smart contracts using documents created according to a certain template to get contract criteria. The research [8]
studies the use of NLP for the generation of smart contracts in accounting and law fields. Furthermore, the paper [9] studies the use of AI for the development of smart contracts regardless of the language used to describe the business rules.

The growth of the blockchain technology market can be seen in Fig. 1 [10].

*Fig. 1. The volume of the world blockchain technology market 2016-2021 (in million U.S. Dollars)*

**Problem statement.** Currently, smart contracts are often developed and deployed manually by blockchain developers. This procedure (see Fig. 2) typically consists of four major steps:

- formalization of business rules;
- smart contract development;
- smart contract deployment;
- smart contract reading.

*Fig. 2. The smart contract development and deployment procedure*

The developer must standardize the rules provided by the client and emphasize the main terms required for the development of a smart contract in the first phase, the formalization of business rules. The smart contract is then built in the second stage under the specifications established in the previous step (see Fig. 2). This usually necessitates familiarity with smart contract development tools. The generated smart contract is deployed to the chosen blockchain in the third stage. Finally, the created smart contract is used through interaction with the blockchain.

The software receives the business rule from the customer at the beginning. The blockchain developer processes this rule and creates the smart contract under
certain specifications. Finally, the customer receives information about the deployed smart contract to interact with it.

This process requires the assistance of a competent blockchain developer, which is either unavailable or impractical for the vast majority of entrepreneurs who could benefit from the usage of smart contracts in their organizations. As the result, it is proposed to create a system that will enable anyone to easily create smart contracts directly from business rules.

**Proposed approach.** The proposed approach will enable the generation of smart contracts according to the requirements without the involvement of a blockchain developer, allowing a greater number of customers to adopt smart contracts in their businesses. The proposed approach (see Fig. 3) consists of the following steps:

- business rules processing using NLP technologies;
- smart contract generation;
- smart contract deployment;
- smart contract reading.

![Fig. 3. The smart contract development and deployment procedure improved by the proposed approach](image)

The initial step (see Fig. 3) includes the automatic processing of business rules using NLP techniques to highlight essential terms for subsequent smart contract code generation. The smart contract code is then constructed in the second phase using the key phrases gathered in the first stage. The generated smart contract is deployed to the chosen blockchain in the third stage. Finally, customers can interact with the produced smart contract deployed to the blockchain network.

The improved procedure takes an English business rule from the customer, which is written in a specified format for tokenization utilizing NLP techniques. On output, the customer receives instructions for using the created smart contract, which was generated by the application under the business rule specified on input. This approach should use NLP technologies, as well as provide a friendly user interface, which can be used by customers without blockchain development skills.

The Python programming language is used for the software implementation of the proposed approach for the generation of smart contracts. It is supposed that business rules will be given in English. The application will then extract essential terms such as “name”, “symbol”, and “initial token supply” from the business rule. This information will be used to develop and deploy the smart contract code according to the ERC-20 standard – one of the most popular smart contract standards nowadays. The created smart contract will be compiled and deployed using the “py-solc-x” library.

For the application to process the business rules correctly, they must be given in a specific structure according to the particular format. It is expected that the business rule text will contain the links “token %token name%”, “symbol %token
symbol%"], and “supply %initial supply%”. The “Spacy” NLP library of the Python programming language is used to detect these links.

The activity diagram demonstrating the algorithm for processing business rules and generating the source code of smart contracts is given in Fig. 4.

Obtained results. Fig. 5 demonstrates the experimental result of smart contract generation from the textual business rule using the proposed algorithm (Fig. 4) and the developed Python application.

According to the given example, the initial business rule “We should create a token Test that has symbol TTT and supply 2000.” is translated into the ERC-20 smart contract, which code is shown in Fig. 5 and the address of the already deployed instance to the Ethereum test network is given below the Solidity code.
Discussion and conclusion. In this study, we proposed the approach to building smart contracts based on business rules using natural language processing. Obtained results demonstrate the viability of the proposed solution capable to create ERC20 token smart contracts from textual business rules. Currently, the proposed approach is limited by the smart contract standard ERC20, even though it is the most widely used token standard, as well as by the necessity for business rules to be given in a specific format. In the future, these limitations should be eliminated and the software solution for end-users that implements the proposed approach should be developed and tested. Furthermore, the intelligent chatbot for the most popular messengers can be created as the blockchain developer’s assistant.

References: