zensar

Distributed Intelligent Contract Enforcement System (DICES)

White Paper

01. Abstract

Existing contract management systems have streamlined the contract creation and execution process, but contract enforcement remains an area of concern. Blockchain technology and its smart contracts inherently enable enforcement by providing means for rule-based peer-to-peer economic interactions while maintaining complete transparency. However, the current legal ecosystem will find it challenging to adopt blockchain and smart contracts due to the technical expertise required. To make the adoption easier, we are proposing a solution that uses Natural Language Processing (NLP) along with a mapping engine to create and deploy smart contracts directly from plain text contracts, without the need for in-depth knowledge of blockchain and smart contract languages. We propose to use NLP and blockchain technology to automate the enforcement of legal contracts in this Distributed Intelligent Contract Enforcement System (DICES).

02. Background

2.1 Problem with existing Contract Management Systems

Digital contract management systems were built to execute and manage contract obligations in an organized manner instead of doing it manually. Currently, one of the parties draft the initial contract, and all other parties negotiate and arrive at a mutually agreeable legal contract. The parties then store the legal contract and monitor its performance in their own contract management systems(CMS)[2]. However, more often than not, parties come with contradictions in their performance obligations as each one of them has their own source of truth. There is no unified store of information leading to a silos mentality. It also involves high operational costs of monitoring the performance of other parties in real-time. It also runs the risk of longer dispute resolution and arbitration costs, as there is no way to improve the performance of other parties. Especially when malicious parties are involved, there can be data tampering. Existing CMS uses a manual process for value settlement leading to high reconciliation costs in the future[2]. According to an estimate, the inability to effectively manage contracts costs the buyers' bottom lines as high as 20% year-over-year[3].

One source of the problem can be linked to a lack of technology that allows direct human-to-human transactions/interactions digitally with a single source of truth. This has changed since the first successful implementation of digital currency, i.e., bitcoin and the technology that is blockchain.

2.2 Blockchain and Smart Contracts

The advent of blockchain has ushered in a new age of technology wherein transactions between people can be done safely and securely, without any trust or intermediary. Bitcoin solved many of the problems related to the digital implementation of a monetary system. It is a ledger technology that is programmed to attain consensus among participating entities about the order of transactions.

Later, blockchain platforms like Ethereum came into the picture, which are general-purpose blockchains. Scripts or pieces of code of choice, called smart contracts, can be run on such a blockchain network. A smart contract is a mechanism involving digital assets and two or more parties, where some or all of the parties put in assets, and these are automatically redistributed among those parties according to a formula based on certain data that is not known at the time the contract is initiated.[6] These smart contracts brought in the possibility of a new generation of digital contracts that are rigid, modular, dynamic, and less ambiguous than traditional legal prose agreements[5].

Moreover, smart contracts have the ability to enforce performance obligations using strict and formal programming languages like Solidity in Ethereum. Smart contracts will not stop and cannot be tampered with once they are deployed on the network whose nodes execute it independently. Smart contracts decrease monitoring costs, as parties need not repeatedly check and monitor obligations in it. Since they are tamper-proof, new commercial agreements can be created between parties that do not have established trust. As Nick Szabo puts it, smart contracts can allow many kinds of contractual clauses to be embedded in hardware and software in such a way that breach of contract is expensive and near impossible[4] A legal contract can also be made between parties that do not know each other but know each other's pseudo-anonymous identities in the network.

The shift to a blockchain-based smart contract management system also comes with its own challenges. The challenges are mentioned in the following sub-section.

2.3 Challenges with shifting to blockchain-based Contract Management System

The positives sometimes can become a hindrance in some instances. Smart contracts, though, rule-based and deterministic, may not be well suited for enforcing many legal contracts today. The process of creating a smart contract will involve substantive decisions about the meaning, content, and applicability of contracting parties' agreements. Programmers will have to make substantive judgments, interpretations, and substantive decisions about potentially uncertain future events when drafting smart contract code[5].

Therefore, the parties have to place their trust in the developers to code a smart contract, which defeats the purpose of having decentralized peer-to-peer contracts. Therefore, there is a need to make contracts that are human-readable as well as machine-readable, as it will bring the utility of both smart contracts and traditional legal contracts into the fold.

Many companies are trying to tackle these challenges with different approaches. Some of these solutions are briefly mentioned below.

03. Existing Solutions

Many initiatives have come up that are building a library of machine-readable contract modules. Each contract has a natural language component and a smart contract component. Ian Griggs first outlined such machine and human-readable contracts in 2004 as Ricardian Contracts. Every initiative, including Open Law, Agrello, R3 Consortium, Common Accord, Mattereum, Ethereum Enterprise Alliance, etc., tries to achieve a version of Ricardian contracts[1].

Open Law is an open-source repository for smart contract templates. Moreover, it allows lawyers to modify their existing templates using a custom mark-up language that uses if-then logic, aliasing, multivariable expressions, basic calculations, etc. while drafting a contract. Open Law has further expanded its smart contract library with oracles from chainlink[8][12].

Mattereum proposed in their whitepaper the use of a Ricardian contract that maintains natural language even when digitized and facilitates the tokenization of property, property ownership, and eventually full transfer and sale of a property. If registrars of Mattereum network are granted a legal title over the assets, then smart contracts can be employed to tokenize the asset and sold to multiple investors. It is also proposed that asset's usage can also be codified about how the custodian should allow the use of it[8][14].

Multiple projects are trying to make human-readable and machine-readable contracts with a variety of approaches. There are other initiatives like Lexon foundation that are trying to develop human-readable code language with varying degrees of success.

04. Zensar's Point of View – Why DICES?

4.1 What are we doing?

Distributed Intelligent Contract Enforcement System (DICES) is a solution that facilitates the conversion and deployment of plain text legal agreements into blockchain-based smart contracts.

Blockchain platforms like Ethereum, quorum, etc. empower users with the power to codify their contractual clauses, enabling self-enforcement of contracts. Using these platforms, we are proposing a blockchain-enabled ecosystem in which enforcement of legal contracts is done automatically. These codifiable contracts are called Smart Contracts, which are computer programs written in languages like Solidity. Lawyers who make the legal agreements do not know the smart contract language, and the programmer does not understand law and agreement making. This makes creating a smart contract out of a legal agreement very tough and requires skilled programmers too.

DICES attempts to overcome the persisting challenges with a blockchain-based contract management system using Artificial Intelligence. DICES will make converting, writing, troubleshooting, executing, and deploying of contracts on blockchain environment easy for companies, lawyers, and IT services.

In our approach to user-friendly self-executable smart contracts, we propose to use a natural language processing module that reads and understands legal prose. NLP eliminates the need for users to learn programming languages necessary to work on blockchain platforms. It also makes the transition from traditional contract management systems seamless. Moreover, it makes the drafting of legal agreements easier as NLP can continuously learn the nuances and semantics of different legal agreements and suggest necessary changes to make the agreements legally complete.

4.2 How will it solve the problems?

Our solution aims at streamlining the entire Contract Enforcement Process through an integrated platform that involves the conversion of plain text agreement to a standardized form through NLP interpreter and system-comprehensible Solidity language, which is triggered for automatic enforcement whenever a contract specified condition is fulfilled.

Implementation of the NLP engine makes the entire agreement analysis and conversion process easier as it continuously learns the nuances and semantics of different legal agreements and suggests the most relevant statements to the intended user for making the agreement more relatable and accurate.

The immutable and distributed nature of ledgers in blockchain allows for transparent performance tracking and the existence of single source truth for the parties involved in a contract/agreement. Codification of the contractual clauses on blockchain enables the parties to transact and interact with each other within the realm of a contract code and in a decentralized manner, i.e., without involvement or dissemination of information to a third party. This enables the parties to create their own custom APIs for interacting with their respective accountable functions in smart contract code.

Automatic enforcement of the agreement through Smart Contracts without any intermediaries helps in faster accomplishment of the whole process resulting in cost-effectiveness and less time consumption.

Parties can create their custom APIs for interacting with their respective accountable functions in the smart contract's solidity code. Therefore, peer-to-peer interaction can be achieved without any intermediary dependency.

4.3 The value offered by DICES

DICES creates and deploys smart contracts based on your legal contract by identifying the agreement's codable clauses, without the need for a developer proficient in the smart contract language. It enables you to reduce your contract enforcement costs by using data and events from your enterprise systems as well as those of your partners in the blockchain network. These events and data act as triggers to the various smart contracts that auto-execute to fulfill the terms agreed in the contract. It allows the organization to operate without constant oversight over the various terms and conditions listed in the agreement that must be invoked as certain events occur. It also prevents the organization from missing certain benefits of the contract that may be lost in the myriad of terms and conditions. The auto-enforcement of contracts also prevents legal complications that may typically arise between organizations, thus saving costs and protecting the company's reputation.

05. Product Features

Here are some of the feature and capabilities of our product:

- Conversion of legal agreements and contracts written by lawyers and attorneys in natural language to blockchain computable digital contracts written in computer languages.
- Supports all blockchain computable contracts like smart contracts on Ethereum, Chaincode on Hyperledger Fabric, etc.
- Artificial Intelligence assistance to give flexibility to lawyers and attorneys in drafting contracts and agreements.
- No need for a blockchain developer for writing and deploying smart contracts.
- Easy deployment on the blockchain networks.
- Seamless and automatic execution of agreement clauses without any third-party intervention.
- Strict logic-based enforcement of contractual obligations.
- No tampering of data(agreement clauses) is possible as the contract is stored in a decentralized platform, which is visible to all the parties involved in the agreement.
- No extra cost required to hire skilled blockchain programmers to develop smart contracts.
- The whole agreement is stored on a distributed database like IPFS for a resilient legal contract storing, without any dependence on solutions that have a single point of failure.
- Facility to make modification and amendments to the pre-deployed contracts with the consent of involved parties.
- All the activities are logged, tamper-proof in the blockchain and only parties with authority can see them.

06. Common Use Cases

Here is a list of some of the use cases for various industries.

Where could DICES influence your value stream?

BFSI

- Auto insurance
- Parametric insurance
- Dynamic credit cards
- Trade financing
- Flow-based credit
- Micro-lending
- Re-insurance
- OTC commodity trades

Hi-Tech & Manufacturing

- Procurement agreements
- Subcontractor management
- Dynamic supply chains
- Inventory replenishment
- Supply chain financing

Retail

- Product warranty
- Dynamic supply chains
- Procurement agreements

07. How DICES works

DICES aims to convert the plain text contracts to computable contracts, i.e., smart contracts, and then deploy them on the blockchain and execute contractual clauses of the agreements on the blockchain platform.

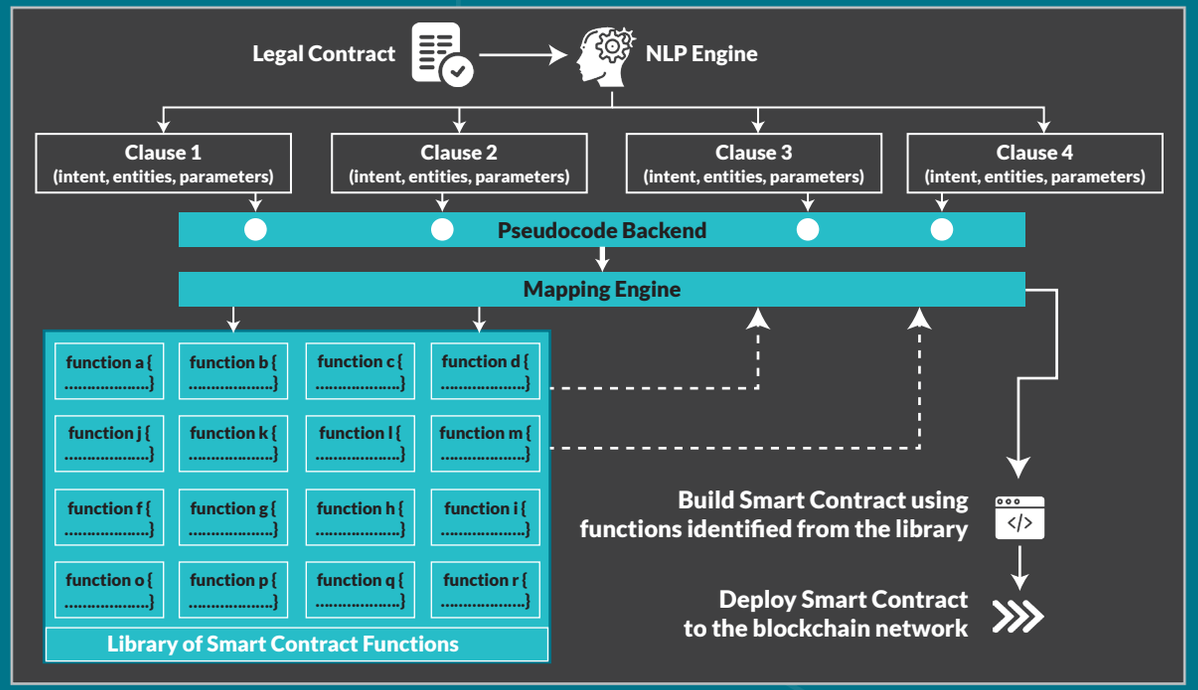
Legal representatives will interact with the application and draft the legal agreement in a human-readable format. The NLP engine working behind the application understands the intent and semantics of each clause in the legal agreement. It outputs the clauses with their intents and extracted entities. The output from the NLP engine then goes into the pseudocode backend. The pseudocode backend generates pseudocode, which has the blueprint of the smart contract. The pseudocode then goes into the mapping engine, which is connected with a smart function library. According to the pseudocode, the mapping engine manipulates the smart functions from this library and creates a smart contract. The generated smart contract is then deployed on the blockchain.

On the successful deployment of smart contracts onto the blockchain network, an application binary interface (ABI) of the respective smart contract and the address at which it is deployed, is provided.

Both ABI and contract addresses will be used to interact with the respective smart contract code. Functions in smart contract codes can be equipped with modifiers that give permission to select users to interact with them.

Parties involved in the contract can choose their own custom APIs in order to interact with the deployed smart contract to perform their duty towards the contract.

The agreement can be stored in a decentralized file transfer system like IPFS so that the users cannot tamper with the agreement, and it is available for all the parties to verify the contents of it using the hash obtained while storing in such decentralized server systems. Like the contract address of a smart contract, the hash of the agreement stored in decentralized servers can also be stored in the respective smart contract.



Application Work Flow

7.1 NLP

Natural Language Processing or NLP is a field of Artificial Intelligence that gives machines the ability to read, understand, and derive meaning from human languages.[9] The plain text used in contracts is in English or any other natural language. Computers cannot comprehend natural languages. Therefore, we are using an NLP engine to make the computer understand the natural language using different models. Natural Language Understanding (NLU), a subset of Natural Language Processing (NLP), is used here to identify the intents and extract the entities in the sentences(clauses) of the agreement.

Intent identification is made using some of the NLP algorithms like Long Short-Term Memory (LSTM), Bi-LSTM, Attention mechanism, and Support Vector Machine (SVM). Entity extraction is done using some of the Machine Learning and NLP algorithms like Conditional Random Field (CRF), Support Vector Machine (SVM), and averaged perceptron. The NLP engine models are trained on a corpus of legal data, and different legal agreements and are continuously learning with the use of the application.

An NLP engine reads the entire agreement and splits the agreement into independent sentences. This spitting is done by using the simple dot '.' separation, newline separation, and NLP engine's intent confidence maximization algorithm. NLP engine then takes these independent sentences and identifies their intent using the algorithms mentioned above. It merges or sorts these independent sentences according to their intents and identifies proper clauses in the agreement.

After identifying the clauses, the NLP engine extracts entities from the clauses which are necessary to make the smart contract.

This information about the clauses, their intents, and entities are then sent to the pseudocode backend.

7.2 Pseudocode backend

Pseudocode backend takes the input from the NLP engine and generates pseudocode, which is a high-level programming language. The pseudocode is kind of a summary of the whole plain text contract and blueprint of the smart contract that is to be generated. It incorporates the intents of all the clauses and respective entities. It also has the logic of sequencing, ordering, and grouping of all the clauses.

7.3 Mapping engine

The mapping engine takes the pseudocode as the input. The pseudocode has the blueprint of the smart contract. The mapping engine then refers to a smart contract template based on the pseudocode generated. According to the pseudocode, the mapping engine modifies the smart functions from the smart function library and generates the smart contract.

7.4 Smart function library

The mapping engine is connected with a smart function library. The smart function library is, as the name suggests, a library of smart contract functions and templates of smart contracts written in the smart contract language(example solidity). The library has some smart template contracts for widely used standardized legal contracts. Here for Ethereum blockchain, Solidity language is used for smart contract coding. The smart contract functions are written in such a way that they can be independently used. The functions are made in such a way that they can be used like LEGO toys. Each function represents a clause, and based on the clauses present in the plain-text agreement, functions in the smart contract will be chosen for deployment.

08. Conclusion

A legal contract is not just limited to individual obligations. They also include clauses that are triggered by time-dependent or sequence-dependent events. Smart contracts are well suited for coding promises that are binary in nature and numerically calculable. Legal agreements often include open-ended terms like “good faith” or “best efforts” whose intent and meaning may vary during the course of the contract. Standard legal agreements include confidentiality, representations, and warranties, which cannot be fulfilled by a code. Hybrid agreements can be utilized in the future where smart contracts are combined with traditional contract management systems.

The system can be scaled up to a fully decentralized solution with user-onboarding, identity management, legal contract storing, events tracking, logs tracking, referencing contracts in case of amendments to a legal contract, and drafting assistance from NLP engine. The system also has huge potential for expanding its utility to arbitration, dispute resolution, complete end-end automation of legal contracts.

09. References

- Andrea Leiter and Jake Goldenfein, July 24, 2018, Legal Engineering on Blockchain Smart Contracts as Legal Conduct, Law & Blockchain, https://www.lawandblockchain.eu/smart-contracts-as-legal-conduct/
- Konfidio, Contract Management: How It Operates & its Current Problems, https://konfidio.com/blockchain/explained/contract-management-how-it-operates-its-current-problems/
- Paul Martyn, July 14, 2018, Blockchain Contract Management -- A Perfect Application, Forbes, https://www.forbes.com/sites/paulmartyn/2018/06/14/blockchain-contract-management-a-perfect-application/#4c1ffd7b49a8
- Nick Szabo, 1997, Formalizing and Securing Relationships on Public Networks
- Primavera De Filippi and Aaron Wright, 2018, Blockchain and the Law, Harvard University Press
- https://blog.ethereum.org/2014/05/06/daos-dacs-das-and-more-an-incomplete-terminology-guide/
- Stalawfirm, Dec 12, 2019, Overview: The Enforceability of Smart Contracts in India, https://www.stalawfirm.com/en/blogs/view/enforceability-of-smart-contracts-in-india.html
- James Metzger, “The Current Landscape of Blockchain-based crowdsourced Arbitration”, Macquaire Law Journal, vol 19, (2019). https://www.mq.edu.au/__data/assets/pdf_file/0008/866294/Macquarie-Law-Journal-Vol-19.pdf
- https://towardsdatascience.com/your-guide-to-natural-language-processing-nlp-48ea2511f6e1
- https://patentbusinesslawyer.com/contract-attorney-contract-drafting-basics/
- https://www.legalmatch.com/law-library/article/contract-drafting-and-review.html
- https://docs.openlaw.io/
- https://jur.io/wp-content/uploads/2019/05/jur-whitepaper-v.2.0.2.pdf
- https://mattereum.com/wp-content/uploads/2020/02/mattereum-summary_white_paper.pdf

10. Authors

Name	Designation	Email
Aditi Yaduvanshi	Assistant Manager	aditi.yaduvanshi@zensar.com
Prem Mehta	Associate Innovation Engineer	prem.mehta@zensar.com
PVS Chanakya Yadav	Associate Innovation Engineer	chanakya.yadav@zensar.com
Anurag Pramod	Associate Manager	anurag.pramod@zensar.com
Shreshtha Mitra	Software Engineer	shreshtha.mitra@zensar.com

zensar

An RPG Company

We conceptualize, build, and manage digital products through experience design, data engineering, and advanced analytics for over 200 leading companies. Our solutions leverage industry-leading platforms, and help clients be competitive, agile, and disruptive as they navigate transformational changes with velocity.

With headquarters in Pune, India, our 10,000+ associates work across 33 locations, including San Jose, Seattle, Princeton, Cape Town, London, Singapore, and Mexico City.

For more information please contact: **marketing@zensar.com | www.zensar.com**