SciDex

The SciDex Protocol - Unlocking Smart Contracts for Businesses

White Paper v.0.0.2

TABLE OF CONTENTS

The State of Smart Contracts

The SciDex Protocol

The SciDex Protocol Lifecycle

The use of oracles in the SciDex Protocol

Storing & Enforcing of the signed smart contracts

The Contracts' Core System

Analyzing a contract

Creating a new contract

The Application of SciDex Protocol, The SciDex Marketplace.

The State of Smart Contracts

In 1994 an American computer scientist by the name of Nick Szabo introduced the concept of smart contracts. Defined as a computerised transaction protocol that executes the terms of a contract, smart contracts are able to fulfill common contractual conditions automatically without involving a third party approval. [1].

In addition, smart contracts facilitate the performance of credible transactions, such as the exchange of money, property, shares or any assets of value, minimizing the need for trusted intermediaries in a transparent and conflict free way [2]. In essence, smart contracts are the best ways to operate a secure transaction whilst ensuring its trackability and irreversibility. The ability to remove the trust factor in a transaction is undoubtedly the future.

Smart contracts are already being utilised for various transactions but have yet to reach mainstream adoption for businesses. The complexity for businesses to generate, deploy and manage smart contracts is undeniable. Furthermore, organizations required by law to have a legal contract for every transaction performed. Current versions of smart contracts are too disconnected from the legal aspect of the transaction.

Additionally, complex transactions are often dependent on the parameters and rules which are defined in the contract. Therefore, translating written contracts to smart contracts and making this process adaptive and automated at scale are necessary for all self-enforcing and self-governing marketplaces [3].

The SciDex protocol is set out to unlock smart contracts for businesses. It starts from scenarios where the transition of direct written to smart contract is currently possible, and ascends the complexity chart by adding additional trusted, secured and proven oracles to the protocol, allowing the expression and execution of more complex transactions.

The SciDex Protocol

The SciDex Protocol is a second layer protocol which facilitates transactions by transforming written contracts into smart contracts. It offers a tool to accurately and deeply analyze complex contracts across verticals. These verticals include insurance, banks and digital asset transactions. In addition, the SciDex Protocol determines underlying patterns that will help to create generic contracts that are easy to be read by both humans and machines. These generic contracts will be used to create secured and unforgeable smart contracts on the blockchain, whilst maintaining compliance standards such as the General Data Protection Regulation (GDPR).

The SciDex protocol generates a Ricardian Adaptive Smart Contract (RASC) which is based on the following three pillars:

- **Ricardian** - A set of digitally signed contracts readable by man and machine first introduced by Ian Grigg during the mid 1990s [4].
- **Adaptive** - The ability for the protocol to generate contracts that changes based on the parameters and rules of the written contract. It is also self-adaptive and enables the buyer to choose the parameters that fit their conditions the best.
- **Smart Contract** - The new hybrid smart contract designed for security, transparency and irreversibility transaction.

The SciDex protocol is characterized by (Prose), (Meta-Tags), (patterns), (parameters),(Oracles)):

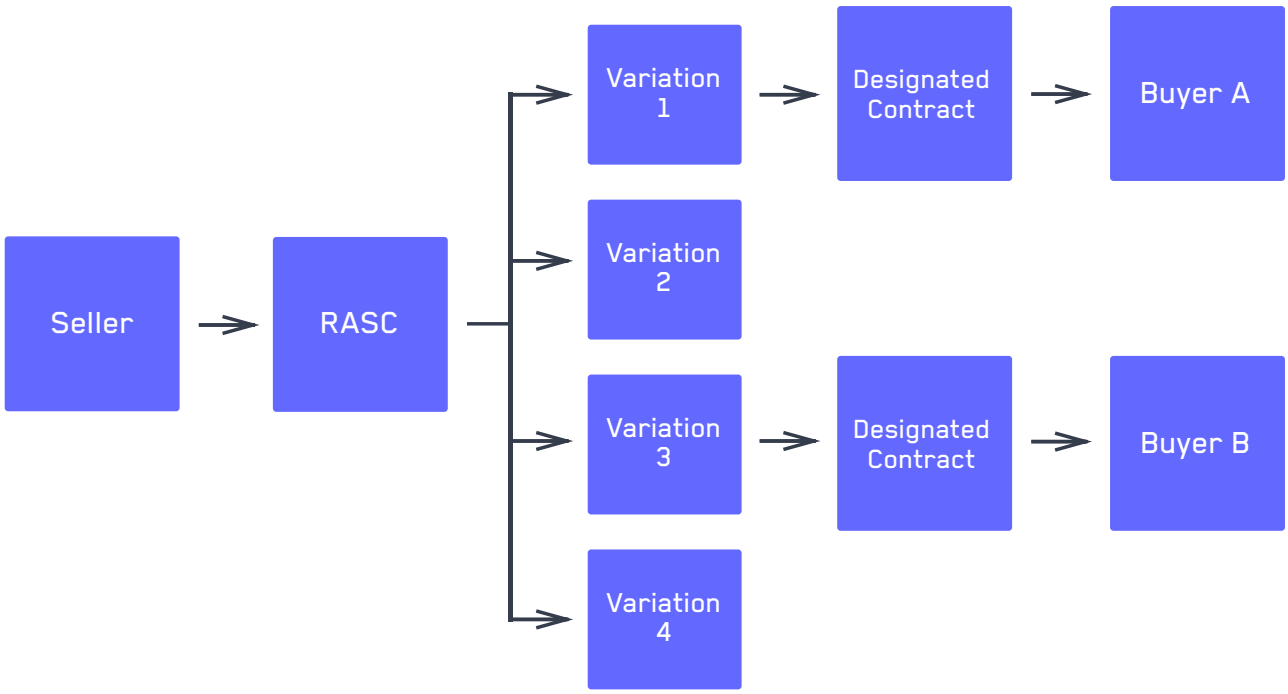
- **Prose** - Prosaic description of the contracts
- **Meta-Tags** - Tags for the smart indexation and research
- **Patterns** - Set of essential business rules & compliance characteristics that are governing the contract. These are crucial to a judge or arbiter in case of a dispute.
- **Parameters** - Variables governing the possible variations of the contract.
- **Oracles** - Third party which validates the transaction in a trustless and dynamic approach.

The SciDex Protocol Lifecycle

When using the SciDex protocol, a few steps are performed to ensure the best possible transaction between all entities. The lifecycle can be summarised with the following three main steps:

1. **Generation** - A seller sets up the basic regulation and compliance rules which governs his side of the transaction when creating a profile. Leveraging the SciDex Protocol the seller can either create a RASC by using an existing written contract or generate one using the contract builder.

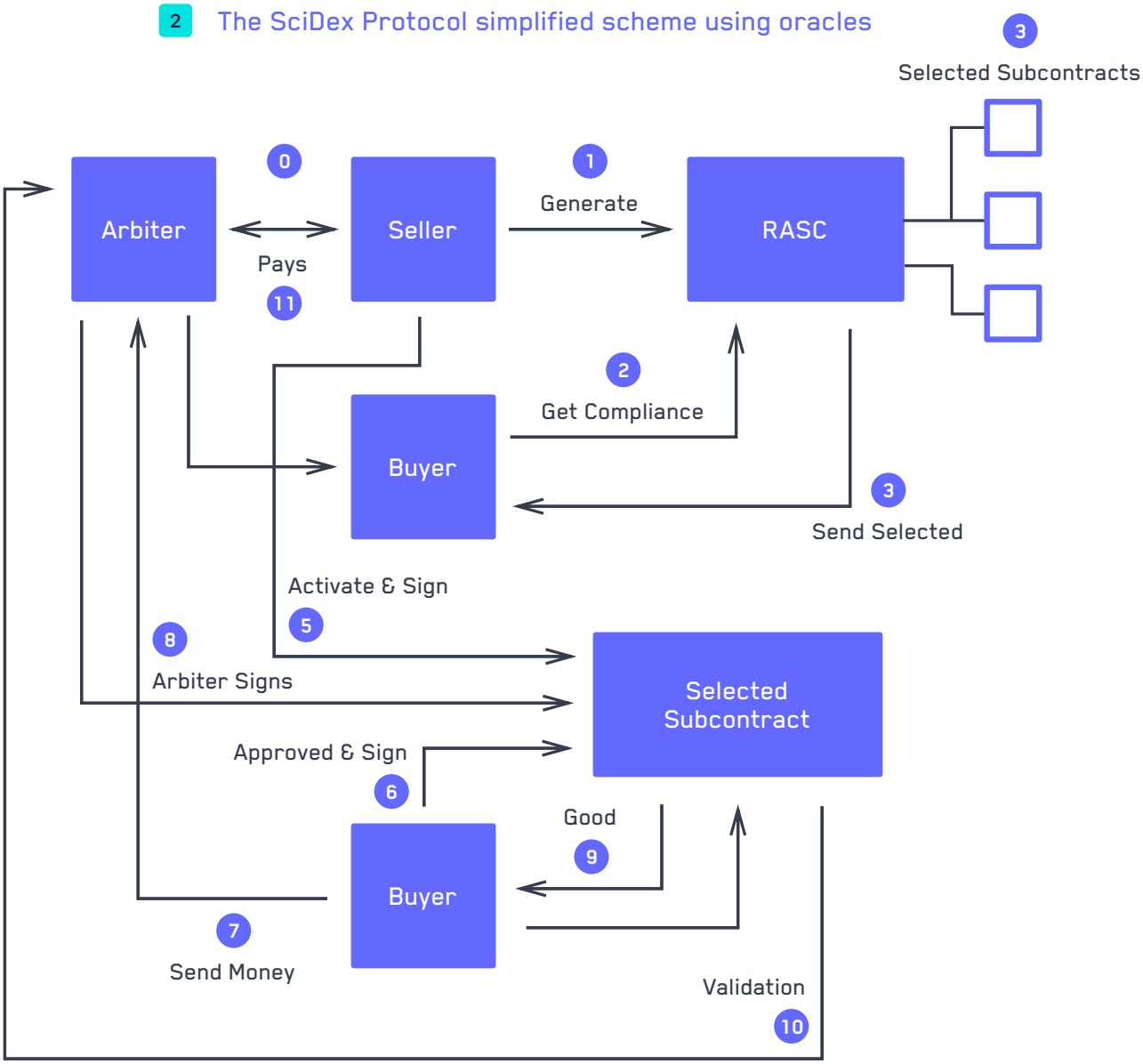
1 The SciDex Protocol generation of multiple designated contracts



2. **Revision & Designated Contract Creation** - Following the generation of the RASC, a buyer reviews the proposed contract variations by the protocol. These variations are based on the different parameters, rules and compliance of both parties, giving the ability to the buyer to select which satisfies his need. When the variation is selected a designated contract containing all the terms the parties agreed upon is then generated and awaits signature of all entities.

3. **Execution** - The execution of the transaction happens when both parties agree to proceed. In the event where a payment is required, payment arbiters can be integrated in the SciDex Protocol as oracles. When multiple oracles are available, the seller can choose amongst the supported ones. Additionally, if the seller decides not to use an arbiter, the RASC will handle the funds collections.

2 The SciDex Protocol simplified scheme using oracles



The use of oracles in the SciDex Protocol

To make the SciDex Protocol adaptive and secure, oracles can be involved in the RASC.
The oracles are the nodes which search and verify real-world occurrences, to validate and approve every RASC being signed.

The main benefits of using oracles are:

- **Security & Being Trustless** - Although transactions are visible, the oracle's internal computations are not, making it trustless. Upon request, can be used for the contracts to contain private information.
- **Scalability** - The oracles are nodes which are built on a side chain. All the information about the transaction and validation are also being stored on the side chain and not on the main chain. In addition, the use of oracles removes the extensive CPU usage of the validation process from the main nodes.

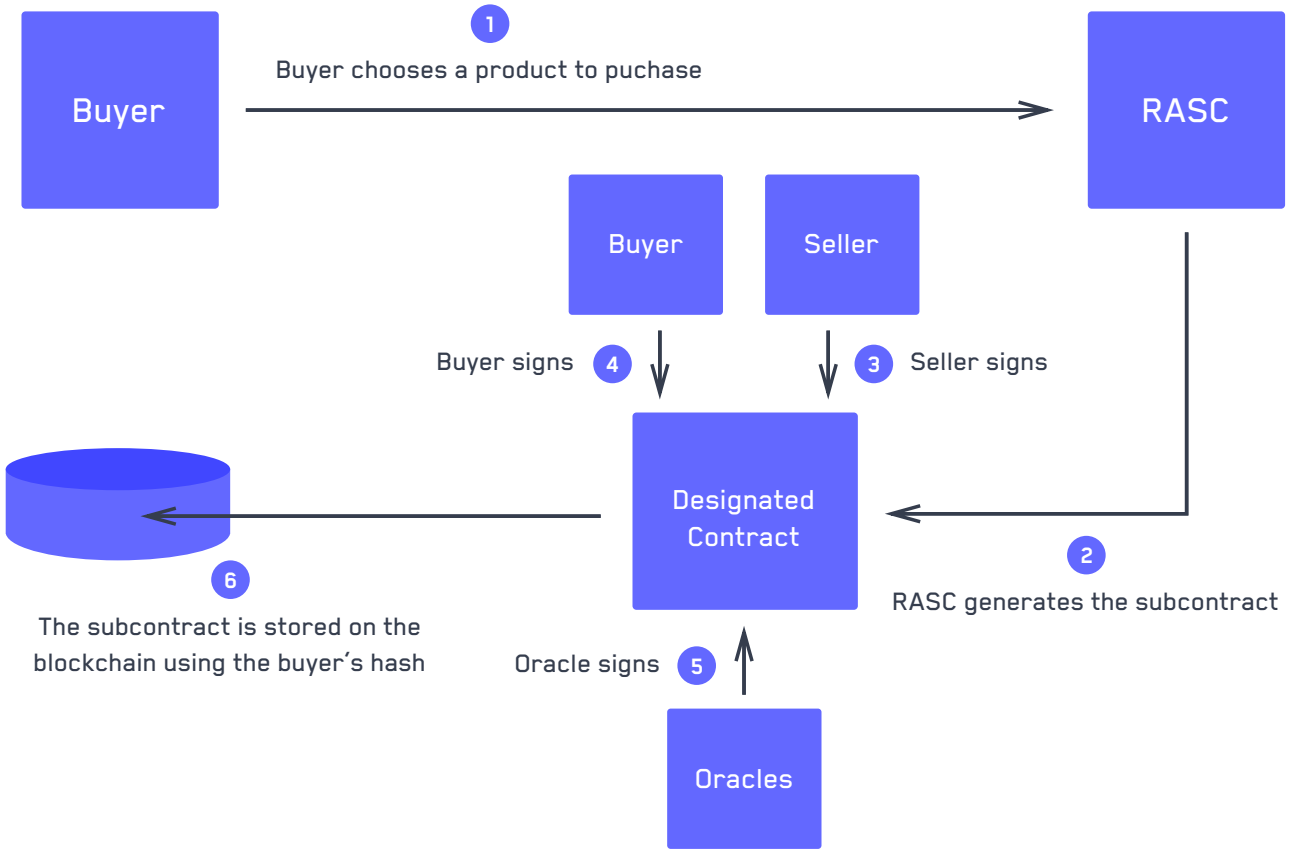
- **Dynamicity** - Thanks to the use of oracles, parameters which define the elements of the transactions can be dynamically changed. It enables for more flexibility in the transaction (ie: price, time, terms of usage) whilst still relying on the immutability characteristic of the blockchain.
- **No Subjectivity** - A transaction without an oracle would require trusting both parties to agree that the conditions were successfully met. The idea of having third party entities which are "all knowing" on the specific transaction conditions removes all subjectivity and makes the the transaction truly trustless.

Storing & Enforcing of the signed smart contracts

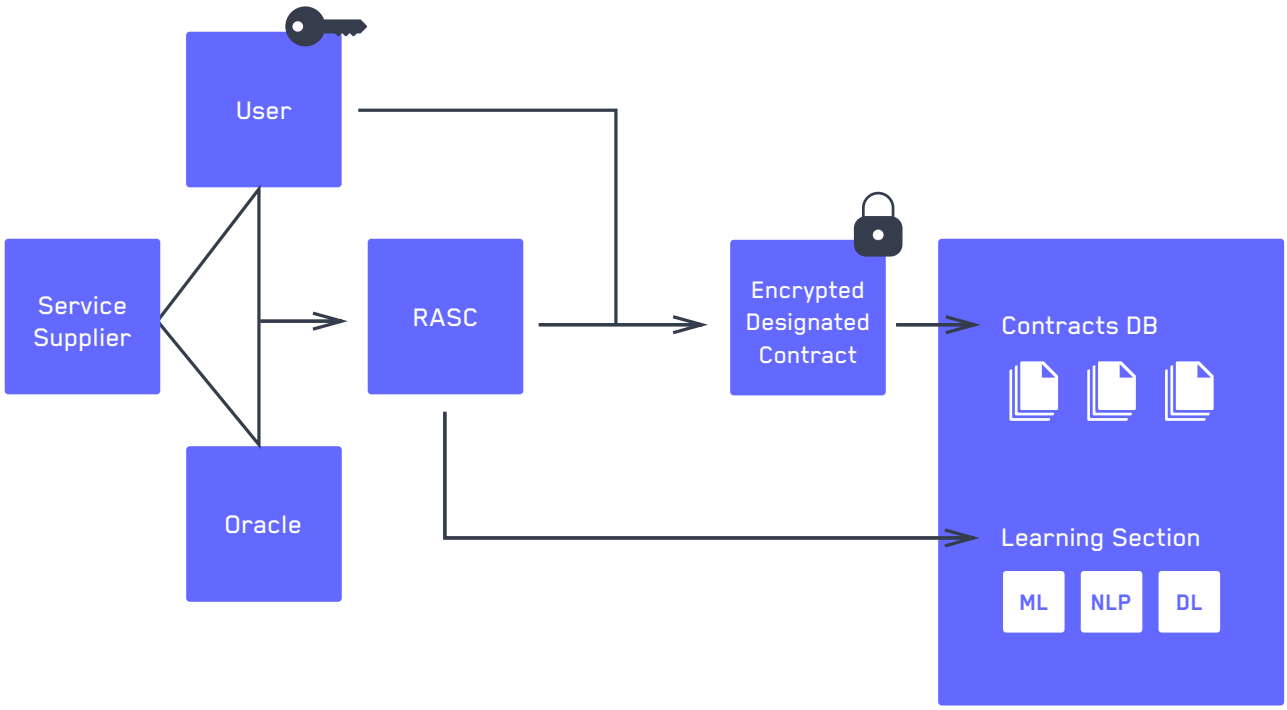
All of the signed designated contracts are encrypted using both parties' public keys and stored on the blockchain. Only the parties who have signed have access to the contract, making it easy for users to keep track of contracts history as well as all currently active contracts.

Thanks to the simple access to the contracts history, every contract allows simple API endpoints to be implemented by the seller. Those API endpoints enable both parties to use the data stored on the blockchain and help them enforce, as well as execute the contract (ie: contract validity, user profile etc..). This mainly helps remove the cloning of repetitive data of the signed smart contracts.

3 The SciDex Protocol main flow



4 The SciDex Protocol transaction stored on the blockchain



The Contracts' Core System

The main technology which powers the SciDex protocol is based on a system which reads and understands written contracts. This enables applications to build on top of the SciDex protocol to ameliorate their platform and simplify transaction flow.

The main components of the core system are made up of the following:

- **The Contract Analyzer** - The component that scans, analyzes and stores contracts. It transforms a simple contract database into a smart knowledge base.
- **The Knowledge Base** - A smart dataset of contracts, which is constantly growing through the addition of both synthetic and authentic new contracts. It contains all the knowledge gathered per domain and per contract in a centralized database. Per-domain contract templates are being generated using this data.
- **The Contract Builder -** This component is combining the per-domain templates built with the user input parameters to fully reflect their needs and compliance on a proposed contract. The output of the Contract Builder is a readable contract that is stored on the Knowledge Base as a synthetic contract.

These components are built using state of the art machine learning and Natural Language Processing techniques. They are used in two main flows - Analyzing Contracts and Building Contracts [5].

Analyzing a contract

The flow of analyzing a new contract entering the knowledge base consists of several techniques that understands its context. The system automatically learns from these contracts, sorts and stores them.

Vector Space Modeling (VSM)
A VSM is built by representing a set of words or a document as a vector.
VSM can be implemented via two unique techniques:

- Traditional Bag of Words (BOW) approach using TF-IDF
- Word embeddings such as doc2vec

Although word embedding techniques perform better than TF-IDF, it requires a very rich training set. For this reason, the VSM model that will be used is the BOW: TF-IDF, will be implemented once the knowledge base is significant [6-8]

Term Frequency & Inverse Document Frequency (TF-IDF)
TF-IDF uses all the words in the Knowledge Base contracts as vocabulary. Term Frequency is the frequency of occurrence of a word from vocabulary in each contract and the Inverse Document Frequency is the number of contracts in which a word occurs. This technique helps by providing a weighting system to words according to the ratio of the number of occurance in a single contract vs the occurrence within all other contracts [9-11].

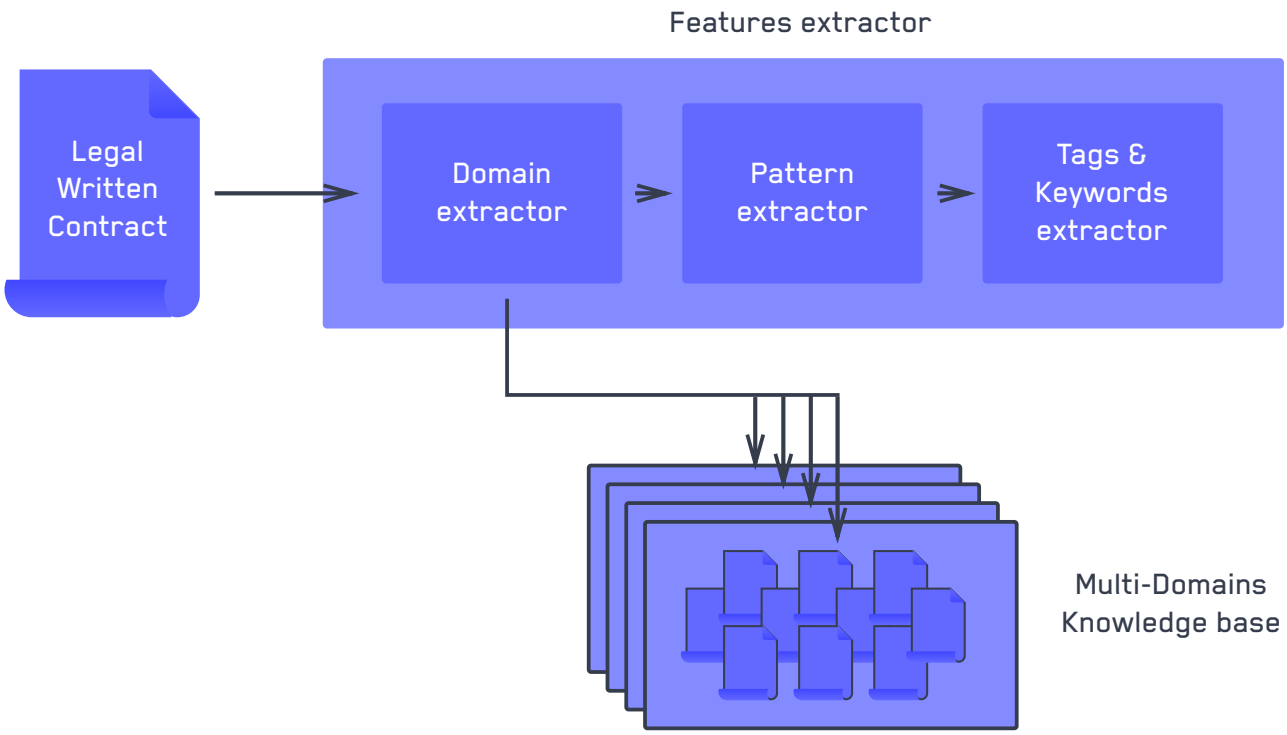
$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

= number of occurences of i in j
= number of documents containing i
= total number of documents

Feature Extraction

The vectors created are being used to recognize and extract features from a contract. The main features extracted are the contracts' domains, patterns, tags and keywords. Using the contracts' extracted features, a multi-domain feature comparing algorithm is applied to find domain correlations [12-14]. These correlated features are being marked to further allow future detection of multi-domain contracts. The analyzed contract is then categorized and stored in the Knowledge Base.

5 A feature extraction flow



Creating a new contract

Users can create a digital contracts and RASC using a predefined profile containing parameters and features such as names, addresses or compliance patterns. like governing state and a predictive assistant based on Neural Network powered by the knowledge base. The flow of generating a new contract, using the user's input and the core's system components, consist of several techniques.

Creating sets of templates

The knowledge base is leveraged to create and maintain sets of templates based on domain's sub-categorization.

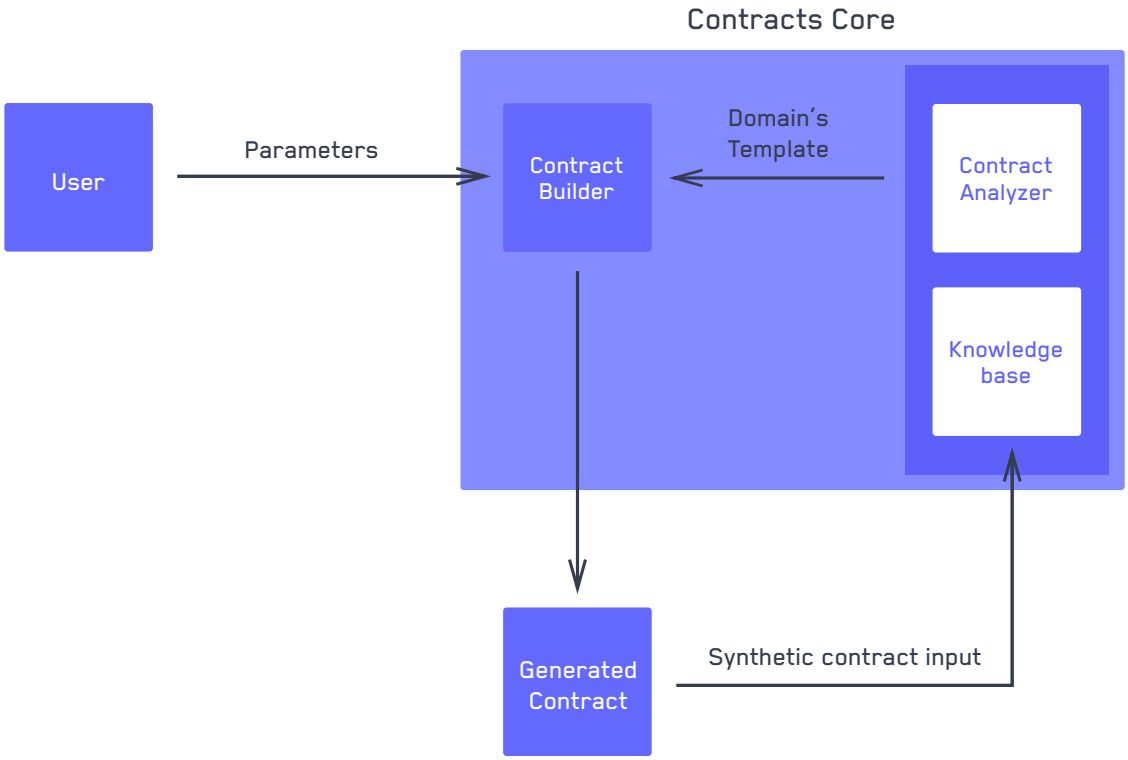
Analyzing user input

The user's input is being converted into a VSM, using the same techniques mentioned before. It is then matched with the knowledge base to find the closest fit domain. In addition, a Named-entity recognition (NER) algorithm is used to find a set of specific parameters within the input (ie: locations, organizations, times etc) [15 -16].

Contract creation predictive assistant

Using the insights derived from analyzing the user's input, a template fitting algorithm is used to find the best fit amongst the templates generated in the previous steps. Once a specific template is selected, it is filled with the properiarty parameters described in the user profile and the user completes it or modifies it using the built-in assistant. The output of this flow is a contract that fits the user's parameters and compliance needs. Once approved by the user, the contract is being fed into the knowledge base as a synthetic input.

6 A contract is being made and fed back into the knowledge base



The SciDex protocol is powered by the SciToken (SDX). During the first stage of the protocol, it will be integrated in the Ethereum mainnet and built for anyone to use and activate.

The RASC generator deploys the contract onto the mainnet by using SDX. Payments set by the RASC can be executed in either Ethereum or SciTokens.

The Application of SciDex Protocol, The SciDex Marketplace.

The first application of the SciDex protocol is data transactions - a data marketspace built by SciDex. In order to develop the SciDex Protocol in the most pragmatic manner, the SciDex MarketSpace will be built side by side with design partners to ensure the ability of frictionless data exchange.

The SciDex MarketSpace offers the opportunity to gain access to an unprecedented collection of scientific data, allowing users to share and monetize data. The core product, the DataDex, is a global index for scientific data listings. Combined with powerful AI tools of the search engine, it enables users to search and correlate multiple sources of data in clusters to find trends, knowledge and causalities.

Built on top of the SciDex Protocol, the objective of SciDex MarketSpace is to secure, standardize and simplify exchange of datasets. It provides the data buyer a secure and parametric smart contract whilst providing the data provider an unforgeable transaction.

In short, the SciDex Protocol empowers transactions of data between businesses in the following ways:

- Managing the complexity of contract creation and execution related to data rights of use, by leveraging the protocol's patterns analyzer & parameters insertion.
- Providing a contract readable by humans
  - Accurate indexation of the dataset in the marketspace (Tags).
  - Clear descriptions of contents and terms (Prose, Ricardian).
- Secured and unforgeable transactions leveraging the designated contracts generated.
- Seamless transaction using Oracles.

More details can be found here about the SciDex MarketSpace Whitepaper.

REFERENCE

[1] Szabo, Nick. "The idea of smart contracts." Nick Szabo's Papers and Concise Tutorials 6 (1997).

[2] "Smart Contracts, Explained". Cointelegraph. 31 October 2017.

[3] Szabo, Nick. "Smart contracts: building blocks for digital markets." EXTROPY: The Journal of Transhumanist Thought,(16) (1996).

[4] I. Grigg. The Ricardian Contract. In Proceedings of the First IEEE International Workshop on Electronic Contracting, pages 25-31. IEEE, 2004.

[5] Radford, Alec, et al. "Improving Language Understanding by Generative Pre -Training."

[6]Bagga, Amit, and Breck Baldwin. "Entity-based cross-document coreferencing using the vector space model." Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics-Volume 1. Association for Computational Linguistics, 1998.

[7]Turney, Peter D., and Patrick Pantel. "From frequency to meaning: Vector space models of semantics." Journal of artificial intelligence research 37 (2010): 141-188.

[8]Wong, SK Michael, Wojciech Ziarko, and Patrick CN Wong. "Generalized vector spaces model in information retrieval." Proceedings of the 8th annual international ACM SIGIR conference on Research and development in information retrieval. ACM, 1985.

[9] Blei, David M., Andrew Y. Ng, and Michael I. Jordan. "Latent dirichlet allocation." Journal of machine Learning research3.Jan (2003): 993-1022.

[10] Aizawa, Akiko. "An information-theoretic perspective of tf–idf measures." Information Processing & Management 39.1 (2003): 45-65.

[11] Robertson, Stephen. "Understanding inverse document frequency: on theoretical arguments for IDF." Journal of documentation 60.5 (2004): 503-520.

[12]C.P. Manning, P. Raghavan, and H. Sch¨utze, Introduction to Information Retrieval, Cambridge Univ. Press, 2008.

[13]Guyon, Isabelle, and André Elisseeff. "An introduction to feature extraction." Feature extraction. Springer, Berlin, Heidelberg, 2006. 1-25.

[14]Abe, Shigeo. "Feature selection and extraction." Support Vector Machines for Pattern Classification. Springer, London, 2010. 331-341.

[15] Tjong Kim Sang, Erik F., and Fien De Meulder. "Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition." Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4. Association for Computational Linguistics, 2003.

[16] Ratinov, Lev, and Dan Roth. "Design challenges and misconceptions in named entity recognition." Proceedings of the Thirteenth Conference on Computational Natural Language Learning. Association for Computational Linguistics, 2009.