FINANCIAL INNOVATION SERIES

GLOBAL RISK INSTITUTE

FINANCIAL INNOVATION SERIES

SMART CONTRACTS

AUTHORS: **Alex LaPlante PhD,**
Managing Director of Research, Global Risk Institute

Alexey Rubtsov PhD,
Research Associate, Global Risk Institute

Charlotte Watson,
Research Analyst, Global Risk Institute

ABSTRACT:

In this paper we discuss smart contracts- computer programmes running on top of a blockchain. We start with some intuitive examples that demonstrate the concept of a smart contract and then consider the Ethereum platform, the most notable platform for decentralized applications. We also outline some applications of smart contracts and discuss associated risks.

INTRODUCTION

In our daily life, contracts are valuable mechanisms to uphold the promises between individuals in a fair manner. By definition, a contract is a voluntary agreement between two or more parties that creates certain obligations and is intended to be enforceable by law. A smart contract is a form of digital contract.[1] It contains a set of coded rules to which the parties to that smart contract agree and by which they must abide. This code is stored, replicated, and executed on a blockchain.[2]

Currently, the most prominent blockchain platform for writing smart contracts is Ethereum. Ethereum is an open-source, blockchain-based distributed computing platform that has its own currency (Ether) and that also provides several blockchain-based features including smart contracts. This paper will use the Ethereum framework to illustrate the functionality and implementation of smart contracts.[3]

1 *The concept of digital contracts traces back to the mid-nineties when the term smart contract was first formally introduced by Nick Szabo: Nick Szabo, "The Idea of Smart Contracts", (1997)*

2 *To learn more about Blockchain, please see our report: Blockchain and Its Applications to Cryptocurrencies. Global Risk Institute (June 4, 2018)*

3 *Note: There are other open-source and proprietary platforms that can support smart contracts*

SMART CONTRACTS

Smart contracts are self-executing programmes which run on a blockchain and are capable of enforcing programmed rules. In simple terms, smart contracts do the following:

1. take some data as input;
2. perform computations on that data according to programmed rules;
3. and produce an output (for example, make a decision).

This framework provides two main benefits: First, smart contracts are executed as they are programmed and are thus not subject to ambiguous interpretation. Second, not only can smart contracts improve efficiency, but they can also provide a way to completely bypass human decision-making, helping to avoid contract ambiguity and decision bias.

To help conceptualize the idea of smart contracts, think of a vending machine which is programmed to release an item after a customer pays for it. In this sense the machine can be regarded as a "contract" between the customer and the owner of the machine. It is important to notice that there is no intermediary here and it is the vending machine that enforces the rule "a customer receives an item after he/she has paid for it".

As a more involved example, assume that you want to play chess for money with someone but you do not trust that person to pay if they lose. To solve the problem, you write an Ethereum program (smart contract) that implements the rules of chess and upload it to Ethereum. Then you send your betting stake to the contract.[4] The other person can see this contract and if they accept it, they can start the game by sending their own betting stake to the contract. Of course, before doing this the other party has to check that the contract is correctly written in that it implements chess and will ultimately send all of its value to the winning player. Once the game has started, there should be no way for either player to extract the money from the contract prior to winning the game, or for anyone else to extract the money under any circumstance. Thus, it is the smart contract, not a third party, that enforces the rules.

4 *Transactions in Ethereum are in "ether", Ethereum's built-in currency.*

ETHEREUM PLATFORM

In this section we use Ethereum to illustrate a particular implementation of smart contracts.[5] Since the launch of Bitcoin in 2009, there have been many alternative cryptocurrencies, so-called altcoins, that offer additional features that some users find useful. These additional features can be seen as applications that support specific tasks (gambling, stock issuance, etc.). However, instead of launching a separate cryptocurrency to achieve each task, it is more convenient to have one general decentralized platform that can support any application.[6] Ethereum is one such example.[7]

Instead of a trusted third party, Ethereum has a network of mutually untrusted peers, called miners, who follow the consensus protocol to ensure that all transactions in the blockchain are valid. The miners bundle the transactions into blocks and append the blocks to the Ethereum blockchain. The use of blockchain technology makes it practically impossible for transactions to be reversed. This feature of the blockchain eliminates the need for trusted third parties that usually authenticate and settle transactions. Based on these properties, smart contracts built on top of the blockchain have a high degree of immutability and security that guarantees that their execution is solely based on their coded terms.

ACCOUNTS

There are two types of accounts in Ethereum: Externally Owned Accounts (EOAs) and Contract Accounts (CAs). The main difference between the two types of accounts is that EOAs are controlled by users (humans), whereas CAs are controlled by their respective codes and not directly by any user (Figure 1). On Ethereum, smart contracts are CAs.

Figure 1. Two types of accounts on Ethereum blockchain. Contract Accounts cannot be controlled by any user.

An EOA can be created completely "offline" without any interaction with the Ethereum network. A sophisticated math algorithm provides an account number and a 'private key' that is used to prove ownership of the account. The algorithm ensures that one cannot figure out the key based on the account address. Once an EOA is created, it may be used to send transactions in order to:

- *transfer currency (ether) to CAs or EOAs;*
- *create new CAs (smart contracts);*
- *and invoke (call) functions of CAs.*

5 *Note: This report is written using the current (August, 2018) state of the Ethereum platform. Ethereum has already experienced several process and structural changes and it is expected that more will occur in the future. The reader is encouraged to consult the most current Ethereum Homestead Documentation to ensure their understanding of the platform is up to date.*

6 *Ethereum allows the use of Turing-complete programing languages to write applications. One may think of Turing completeness as something that allows you to write very involved applications. It is important to notice that many existing altcoins use programming languages that are not Turing complete (for example, Bitcoin's scripting language does not allow loops).*

7 *Other examples include Lisk, Qtum, and Ubiq.*

Currency transfer is rather straightforward and the corresponding procedure is similar to that of other cryptocurrencies. The creation of a CA, however, is more subtle. To create a CA one has to send a transaction from their EOA and include the contract's code in a special field within this transaction.[8] Once the code is written, it must be compiled into an Ethereum-specific binary format (Ethereum Virtual Machine bytecode) that is then verified (executed) by the miners (Figure 2). If there are no mistakes in the code, a CA with a new and unique address is created and added to the blockchain.[9]

Figure 2. Smart contracts are verified/executed by the miners in Ethereum network before being deployed to the Ethereum blockchain

Once a smart contract or CA is on the blockchain it can be triggered by a transaction from any EOA or CA, unless the contract specifies otherwise.[10] Once this occurs, the smart contract's code is executed by each node (miner) in the blockchain network. The invalid execution of a smart contract is prevented by the consensus protocol which is based on "Proof-Of-Work" puzzles.[11] In this sense, the majority of the nodes decide on the correct execution of the contract.

To better understand this mechanism, consider the following smart contract that we assume has already been deployed to the Ethereum blockchain and can be accessed by its address.[12] The code of the contract, written in the programming language ***Solidity***, is:

```
contract NameRegistry {
      mapping(bytes32  =>  address) public
registryTable;
      function claimName(bytes32 name) {
            if (msg.value  <  10) {
                  throw;
            }
            if (registryTable[name]  ==  0) {
                  registryTable[name]  = msg.sender;
            }
      }
}
```

8 *Note: The Ethereum platform only supports certain programming languages including Solidity, Serpent, or Lisp, while other blockchain-based platforms may use different programming languages. For example, Lisk allows to one to deploy blockchain applications in JavaScript.*

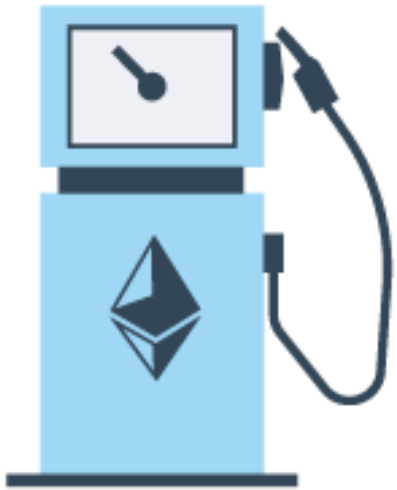
9 *Before uploading the contract to the actual blockchain, contract developers can test their creations on the testnet which is a test network where one can run their applications without spending real money.*

10 *If unrestricted, once a smart contract is on the blockchain it can be used by anyone with an EOA. However, in some cases smart contracts can only be used by those EOAs or CAs specified within the contract.*

11 *To learn more about Proof-of-Work, please see our report: Blockchain and Its Applications to Cryptocurrencies. Global Risk Institute (June 4, 2018)*

12 *Narayanan, A., Bonneau, J., Felten, E., Miller, A., Goldfeder, S.: Bitcoin and Cryptocurrency Technologies. Princeton University Press, 2016.*

This contract implements a name registry in which names can be assigned to any address that invokes this contract (mapping an address to a name). The contract also makes sure that the caller of the contract has sent 10 wei for registering, otherwise registry will not occur. To use this contract, one must send a transaction to the contract that specifies the 'name' to be assigned to the sending address, some amount of ether to pay the miners for running the contract, also known as gas and the 10 wei registry fee. This transaction is then executed by the miners meaning that they all run the code of the contract and agree, according to consensus protocol, on its updated state, that is, the registry (list of mapped names). It should be emphasized that only the miner who adds the block with this new transaction to the blockchain collects payment for running the contract (see Risk Section for criticism of this scheme).



GAS

Smart contracts are written in a Turing-complete programming language, which means it is possible that a given code will never terminate. This can cause problems for miners who run (verify) the code. To make sure that a code's runtime is finite, Ethereum introduced the idea of gas.

Gas has two main purposes:

1. Ensure that a contract code does not contain an infinite loop offering protection against denial-of-service attacks caused by adversaries submitting computationally intensive operations that slow down the network
2. Compensate the miner who adds the smart contract (transaction) to the blockchain

The gas can be thought of as the complexity and cost of operations that the code implements. Every operation that needs to be completed to execute the contract code costs a certain number of gas (an integer quantity); operations that require more computational resources cost more gas than operations that require less computational resources. For example, the addition operation ("+") costs 1 gas. When users send their codes to the network they have to specify the amount of gas required to execute the code and the price per unit of gas (in ether). The miners will run the code as long as there is a sufficient amount of gas supplied in the transaction. The total gas used multiplied by the price per unit of gas represents the fee paid to the miner who includes the transaction in the blockchain.

While the amount of gas required per operation is fixed, the amount users pay per gas varies and is determined by market conditions. Ethereum's built-in

currency, ether, is used to buy gas.[13] Every transaction must specify the price that the user is willing to pay for each unit of gas consumed. Miners can refuse to process a transaction if the price is too low, thus the higher the price per gas, the quicker the transaction will be processed. As should be expected, the gas price specified in a transaction is inversely related to the amount of time it will take for the transaction to be included in the blockchain.

MARKET FLUCTUATIONS: GAS AND ETHER

One thing to keep in mind if you are thinking of using Ethereum (or similar) for smart contract deployment is that the cost of inclusion on the blockchain will depend on the market fluctuations of both the price of gas in ether and the exchange rate of ether and your fiat currency of choice. Moreover, if speed is an issue you will need to pay more gas to have your smart contract added to the block more quickly.

EXAMPLE:

At the time of block number 5,772,702, the average gas price was 2Gwei (1 ether = 10^9 Gwei) and 1 ether was worth approximately $520 USD. This corresponds to a price of approximately 0.0001 cents USD per 1 gas. (This price is evaluated for transactions that were added to the blockchain in less than 5 minutes.)

13 *The smallest denomination of ether is called Wei: 1 ether = 10^18 wei. Other units used in the Ethereum network include: Kwei (10^3 wei), Mwei (10^6 wei), Gwei (10^9 wei), microether (10^12 wei), and milliether (10^15 wei).*

APPLICATIONS FOR FINANCIAL SERVICES

In recent years, there has been a notable uptake of blockchain-based technologies across several industries. (Table 1) The World Economic Forum, for one, projects that by 2027, 10% of global GDP will interface with these technologies.[14] As such, financial institutions have begun to investigate, test and implement various blockchain applications, including smart contracts, within their organizations. Amongst other things, smart contracts have the potential to reduce settlement time for syndicated loans; decrease processing costs in the mortgage loan industry; and automate and reduce processing overheads in the personal motor insurance.[15] Ultimately, smart contracts can provide cost savings for both the financial institution and their consumers.

It is important to recognise that the Ethereum blockchain is permissionless (public) and anyone can have access to it. On the other hand, some of the applications are based on permissioned blockchains in the sense that only a limited number of members can have access to it (for example, a consortium of banks or insurance companies). Not only do such permissioned blockchains provide better data privacy, but they also allow the members to be independent of various factors that affect the dynamics of a public blockchain (for instance, the price of the underlying cryptocurrency, the need for gas, etc.).

14 *Global agenda Council on the Future of Software & Society: Survey Report, (2015), "Deep Shift – Technology Tipping Points and Societal Impact", World Economic Forum, pp. 24,*

15 *B. Cant, A. Khadikar, A. Ruiter, J. Bolgen Bronebakk, J. Coumaros, J. Buvat, A. Gupta, (2016), "Smart Contracts in Financial Services: Getting from Hype to Reality", Capgemini Consulting*

Table 1. Range of use case applications for smart contracts

Use Case		What the Smart Contracts Can Do
Financial Services	Trade Clearing and Settlement	Manages approval workflows between counterparties, calculates trade settlement amounts, and transfers funds automatically
	Coupon Payment	Automatically calculates and pays periodic coupon payments and returns principal upon bond expiration
	Insurance Claim Processing	Performs error checking, routing, and approval workflows, and calculates payout based on the type of claim and underlying policy
	Micro-Insurance	Calculates and transfers micropayments based on usage data from an internet of Things-enabled device (ex. Pay-as-you-go automotive insurance)
Life Sciences and Health Care	Electronic Medical Records	Provides transfer and/or access to medical health records upon multi-signature approvals between patients and providers
	Population Health Data Access	Grants health researchers access to certain personal health information; micropayments are automatically transferred to the patient for participation
	Personal Health Tracking	Tracks patients' health-related actions through IoT devices and automatically generates rewards based on specific milestones
Technology, Media, and Telecom	Royalty Distribution	Calculates and distributes royalty payments to artists and other associated parties according to the contract
Energy and Resources	Autonomous Electric Vehicle Charging Stations	Processes a deposit, enables the charging station, and returns remaining funds when complete
Public Sector	Record-Keeping	Updates private company share registries and capitalization table records, and distributes shareholder communications
Cross-Industry	Supply Chain and Trade Finance Documentation	Transfers payments upon multi-signature approach
	Product Provenance and History	Facilitates chain-of custody process for products in the supply chain where the party in custody is able to log evidence about the product
	Peer-to-Peer Transacting	Matches parties and transfers payments automatically for various peer-to-peer applications: lending, insurance, energy credits, etc.
	Voting	Validates voter criteria, logs vote to the blockchain, and initiates specific actions as a result of the majority vote

Source: Deloitte, David Schatsky, "Getting smart about smart contracts"

In October 2016, Swiss Re, Zurich Insurance, Munich Re, Allianz, and Aegon formed the Blockchain Insurance Industry Initiative (B3i) as a collaborative attempt to explore the potential of blockchain-based technologies, including smart contracts. The ultimate goal of the initiative is to streamline paper work and reconciliations for reinsurance and insurance contracts and to help accelerate the flow of information and money. B3i has since produced a working prototype for a distributed smart contract management system for property catastrophe excess of loss (Cat XOL) contracts. The first deployment of the system is expected in late 2018.[16]

JP Morgan has built a new system based on the Ethereum platform called Quorum which aims to replace numerous interconnected legacy databases with a single, shared, and unchangeable record of transactions. The system allows transactions between permissioned groups of known participants through the use of smart contract-based governance tools that permit an agreed upon entity to maintain operational control and enforce cybersecurity best practices.[17]

The French insurance firm, AXA, has developed a blockchain-based insurance product, Fizzy, for flight delays. The insurance smart contract is connected to global air traffic databases and automatically compensates insured flyers if they experience a delay of more than two-hours.

In the pension space, APG and PGGM have joined forces to experiment the applications of blockchain in pension. They have successfully completed the first phase which involved the development of an administration blockchain application. Their end goal is to use the technology to lower participant

16 *Our Product, The Blockchain Insurance Industry Initiative, www.b3i.tech*

17 *J.P.Morgan , Quorum: Advancing Blockchain Technology, wwwJpmorgan.com*

costs, simplify pension administration, enhance security, and increase data accessibility to participants.[18] Start-up and pension-challenger Auctus claims to be the world's first smart contract-based retirement plan platform that provides robo-advisory services over both traditional and cryptocurrency-based assets.[19] The platform will offer automated contribution collection, pension payouts, and audit reporting.[20]

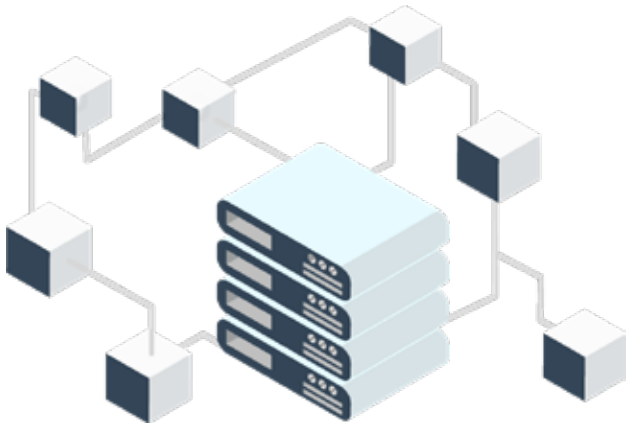
There are yet other examples where financial institutions have paired up with technology companies to develop solutions. AIG, IBM, and Standard Chartered have successfully piloted a multinational smart contract-based insurance policy which provides real time access to policy data, documentation, and coverage and premium payments. This has enabled both AIG and Standard Chartered to execute multinational coverage more efficiently.

BMO, CaixaBAnk, Commerzbank, Erste Group, UBS, and IBM have joined together to form Batavia, an open trade platform built on the IBM hyperledger system, designed to help automate international trade transactions, effectively decreasing the costs of global trade.[21]

Industry leaders have also been actively collaborating with academia and startups on blockchain-based technologies. BBVA, Credit Suisse, ING, JP Morgan, the Depository Trust and Clearing Corportation (DTCC) and over 500 others have joined the Enterprise Ethereum Alliance (EEA) which connects industry with academia, startups, and tech vendors with Ethereum expertise. EEA's goal is to customize Ethereum's smart contract technology for industry players[22]

Similarly, the Initiative for Cryptocurrencies and Contracts (IC3) is a collaboration of finance and banking experts, entrepreneurs, regulators, and open source software communities who are looking to bring blockchain-based solutions from concept to reality. IC3 is led by faculty members at Cornell University, Cornell Tech, UC Berkely, UIUC and Technion, and partners with Microsoft, JP Morgan, FidelityLabs, and Intel, amongst others.[23]

These are by no means the only examples of smart contract activity within the financial services sector and, given the rapid advancement in the space, we can certainly expect more smart contract applications to come to fruition in the near future.



18 *APG, "APG and PGGM develop a blockchain application for pension administration", (2017)*

19 *Auctus, https://auctus.org/*

20 *Jasmine Ye Han, "Blockchain Technology Hits Retirement Plan Industry", Bloomberg BNA, (2017)*

21 *Andrew Munro, "IBM trade blockchain completes first smart contract transactions", finder, (2018)*

22 *Enterprise Ethereum Alliance, https://entethalliance.org/*

23 *IC3 Partners, "The Initiative for CryptoCurrencies and Contracts", www.initc3.org*

RISKS

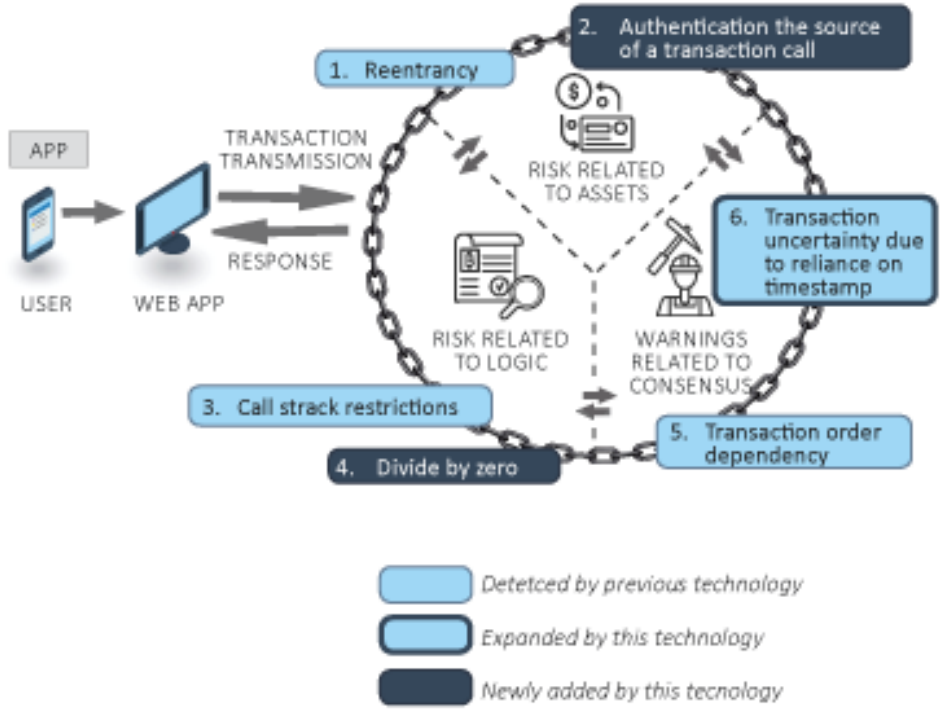
As with any technology, smart contracts have the potential to introduce a variety of risks to financial institutions including those associated with human error, cyber vulnerability, legal issues, and regulatory compliance.

Software is inherently susceptible to defects (bugs). In normal circumstances, once a bug is identified in the underlying code, the code can be edited and the bug can be fixed. Since smart contracts are stored on a blockchain which is immutable, there is no way to edit a smart contract once it is deployed.[24] For example, consider Decentralized Autonomous Organizations (DAOs), a smart contract on the Ethereum blockchain and the largest crowdfund in history. Due to the vulnerabilities in its code, a hacker was able to steal approximately $50 million from DAOs. In a similar situation, a smart contract coding company, Parity, reported that $30 million was stolen in the summer of 2017 due to a bug in a specific multi-signature contract.[25] To address such problems, in March 2018 the Japanese IT giant Fujitsu announced that it has developed a verification technology capable of identifying six major risks in Ethereum smart contracts' source code (Figure 3).[26]

Highlighting another risk of employing smart contracts is a recent dispute between Singapore-based Soar Labs and Australian Byte Power Party Ltd.[27] When Soar Labs was acquiring a 49% stake in Byte Power, the $5 million USD value of the stake was paid mostly in Soarcoins, Soar Labs's blockchain-based crypto-token. A conflict between the companies resulted in Soar Labs withdrawing $6.6 million worth of Soarcoins from Byte Power's e-wallets. Having not carefully looked at and understood the smart contract before agreeing to it, Byte Power was not aware that Soar Labs was contractually allowed to do this. A careful inspection of the smart contract revealed that the code contained a function that permitted Soar Labs to rewrite all balances in Soarcoins at will.

Figure 3. Types of risk that can be detected in smart contracts

Recreated from source: Fujitsu, Press Release March 7, 2018



24 *Atzei, N., Bartoletti, M., Cimoli, T., "A survey of attacks on Ethereum smart contracts". Working paper. (2017)*

25 *Wolfie Zhao, "$30 Million: Ether Reported Stolen Due to Parity Wallet Breach", Coindesk, (2017)*

26 *Fujitsu, "Fujitsu Develops Technology to Verify Blockchain Risks", (2018)*

27 *Jeremy Kirk, "Exclusive: Aussie Firm Loses $6.6M to Backdoored Cryptocurrency", Bank Info Security, (2018)*

Although in this case the subversive behavior of the code was not hard to detect, computer scientists admit that there are coding tricks that could make it rather difficult to find subversive behaviour in a smart contract. This highlights the need for highly qualified specialists that can thoroughly audit the code of a smart contract.

When employing Ethereum, the gas payment can also cause concern. As mentioned, the gas payment indicated in a transaction goes to the miner who includes the transaction in the blockchain. However, all miners who build up that block should verify the transaction as well even though they are not paid for doing so. Consequently, there is an incentive for the miners to skip the validation step which may, in the long run, impact the health of Ethereum's blockchain.[28]

Many contracts require real-world data to be submitted to a blockchain. Since blockchains cannot access external data directly, specially designed agents, known as oracles, provide these data to the blockchain. Should the data show that a pre-defined condition within the smart contract is met, the contract will execute. Oracles run off-chain and are not typically subject to the consensus mechanism underlying public blockchains. As an example, consider a contract that needs a quote for a stock price or the average global temperature. If the contract is provided incorrect data that causes the contract to execute, this cannot be undone. Smart contracts cannot revert execution. It is for this reason that it is crucial that smart contracts are provided with trusted information sources. In this sense, smart contracts are susceptible to unintentional human errors as well as malicious attacks aimed at corrupting the data being fed to the blockchain.[29]

While smart contracts have the potential to bring clarity, predictability and auditability to contractual relations, there are still several key legal and regulatory issues that need to be resolved. For starters, smart contracts will still need to follow current contract law and may not be legally enforceable if they do not. For instance, if a party of the smart contract is not of legal contracting age. Or, as seen in the Soar Labs and Byte Power case, smart contracts can also contain hidden nefarious code that may not hold up in a court of law.

Since they are deployed on a decentralized system, smart contracts also present unique jurisdictional issues. It is not explicitly clear what legal jurisdiction a smart contract would fall under. This becomes increasingly uncertain when dealing with smart contracts between several parties situated around the globe.

Looking to the future, financial institutions will need to evolve their contracting practices if they hope to employ smart contracts while minimizing the legal risks. Likely this will mean the implementation of 'hybrid' contracts consisting of both a smart contract code and a more traditional English language contract.[30]

28 *Narayanan, A., Bonneau, J., Felten, E., Miller, A., Goldfeder, S, "Bitcoin and Cryptocurrency Technologies", Princeton University Press, 2016.*

29 *Deloitte, "Blockchain Risk Management", Deloite Malta Risk Advisory, (2017)*

30 *Matthew McMillan, "Smart contracts: Legal and regulatory challenges of smart contracts", (2017,*

CONCLUSION

In this paper we introduce smart contracts—fully-functioning computer codes acting on top of a blockchain. Inheriting many features from the underlying blockchain technology such as immutability and decentralization, there have already been several notable and promising applications of smart contracts within the financial services sector. There are risks that come along with this new technology, however, that both users and developers of smart contracts should be aware of and manage as effectively as possible.