Arbitration of Smart Contracts Part 1 – Introduction to Smart Contracts

Kluwer Arbitration Blog
August 23, 2018

Ibrahim Mohamed Nour Shehata (Cairo University & NYU School of Law)

Please refer to this post as: Ibrahim Mohamed Nour Shehata, 'Arbitration of Smart Contracts Part 1 – Introduction to Smart Contracts', Kluwer Arbitration Blog, August 23 2018, http://arbitrationblog.kluwerarbitration.com/2018/08/23/arbitration-smart-contracts-part-1/

As described by Max I. Raskin, a blockchain is simply a decentralized ledger for recording digital data in a verified time-stamped manner without the need for a trusted third party. Blockchain technology provides, according to Joseph Bambara, et al., more "security, traceability, and transparency of records…as well as lower operational costs." In this regard, public blockchains are protected from security threats because they maintain the information on multiple nodes where more than 51% of the nodes would have to be compromised before any security breach could occur.

The best definition of a smart contract is: "a set of promises, specified in digital form, including protocols within which the parties perform on these promises."[fn]Nick Szabo, *Smart Contracts: Building Blocks for Digital Markets*(1996).[/fn] Accordingly, a smart contract is a computerized algorithm which automatically performs the terms of the contract. As Bambara notes, smart contracts lie on a wide spectrum ranging from vending machine contracts to fully blockchain-executed smart contracts. As described here, a recent example of fully blockchain-executed smart contracts is a smart contract for a flood insurance policy, linked to the precipitation data from the Met Office. Once the data from the Met Office feeds into the blockchain, the policy is automatically triggered, and insurance claims are paid out. Our discussion in this series of articles will focus on smart contracts executed on public blockchains such as Ethereum. Please find a chart available here explaining the concept of smart contracts that are executed on blockchains.

As Raskin notes, Smart contracts typically have the following characteristics: (1) execution is automated; and (2) performance is ensured without recourse to law enforcement. In this regard, the main difference between smart contracts and traditional legal contracts is "the ability of smart contracts to enforce obligations by using autonomous code."[fn]De Filippi Primavera and Aron Wright, *Blockchain and the Law: The Rule of Code*(2018).[/fn] Smart contracts do that by recording performance obligations in a strict and formal programming language (like Ethereum's Solidity).

Generally speaking, the code of the smart contract is executed without relying upon a trusted third party[fn]In a forthcoming post, we recommend the inclusion of oracles in smart contracts, whereby we argue that this hypothesis is overestimated when it comes to smart contracts dealing with off-the-chain events.[/fn]; the code is rather implemented in a distributed manner by all of the nodes supporting the underlying blockchain-based network whereby no single party controls the blockchain[fn]This is the case with public blockchains only. There are private blockchains which are usually administrated and controlled by a trusted third party.[/fn] (i.e., Ethereum). This autonomous scheme makes the promises recorded into smart contracts to be – by default – more difficult to get amended or terminated than promises in traditional legal contracts recorded in natural language (i.e., legalese). Accordingly, as Kevin Werbach and Nicholas Cornell have written, unless the parties have

incorporated some logic in their smart contract to enable the amendment and the termination of such a smart contract, then there might be no way to halt the execution of a smart contract after it has been triggered by its parties.

Legal Challenges Related To Smart Contracts

Smart contracts raise numerous enthralling legal challenges. This section will try to shed light upon some of these legal challenges as follows:

(1) Legal Effects

As a starting point, are smart contracts legal binding contracts? The answer to this question depends upon three main factors: (1) the specific use case; (2) the form of smart contract being used (i.e. entirely coded in software or a hybrid smart contract with both an encrypted coded version and a text-based version); (3) the law applicable to the contract. This means that the answer might vary significantly depending on the concerned jurisdiction. As Bambara has observed, often the certainty of the content of the contractual terms and whether they are comprehensive enough is a critical factor in determining the legal effects of any contract in numerous jurisdictions. In order to eliminate such uncertainty surrounding the legal effects of smart contracts, some states like Delaware, Tennessee, and Arizona have passed legislation to recognize the legal effects of smart contracts. In 2017, Arizona has passed the amended Arizona Electronic Transactions Act (AETA), HB 2417, which defines blockchain technology as a "distributed, decentralized, shared and replicated ledger, which may be public or private, permissioned or permission less, or driven by tokenized crypto economics or token less" and indicates that the "data on the ledger is protected with cryptography, is immutable and auditable and provides an uncensored truth." HB 2417 also defines smart contracts as an "event-driven program, with state, that runs on a distributed, decentralized, shared and replicated ledger that can take custody over and instruct transfer of assets." Therefore, parties to a smart contract might be able to ensure that their smart contract is legally binding if they elect the law applicable to the contract to be that of Arizona, or Delaware or Tennessee or any other jurisdiction that recognizes the legal binding effects of smart contracts. Such a choice of law has to be supplemented by choice of forum that would recognize and enforce the parties' choice of law.

(2) Amendment and Termination of Smart Contracts

The original smart contract concept has started with the invention of the vending machine. With a vending machine for soft drinks, one can insert a dollar for instance and gets back a soft drink. However, the process of a vending machine is not flawless. For instance, what if one changed his mind after inserting the dollar and wants to get chocolate instead of a soft drink; or, what if one changed his mind and did not want anything anymore. An even more intriguing question, what if the vending machine does not perform its obligation and dispenses the soft drink; I am sure many of us have faced such a situation and did not know what to do. These examples also apply in the realm of smart contracts which are entirely recorded on blockchains.

(3) Coding limitations

Whenever one mentions coding limitations in the world of the blockchain, the decentralized autonomous organization ("**DAO**") incident has to be mentioned. As described by Raskin, the DAO was formed in 2016 to create an investing fund that "would not be controlled by any one individual, but by shareholders voting based on their stakes on a blockchain." The DAO was able to pool funds worth $150 million. Soon after this money was raised, a hacker was able to divert about what is worth $40 million funds from the DAO in an unpredictable manner. The hacker did not "hack" the code in a malicious way but rather exposed a legal loophole in the smart contracts of the DAO. This incident

shows how coding is limited and how bugs could be simply exploited by hackers. Thus, as David Zaslowsky noted here, it is not really surprising that a 2016 study of Ethereum smart contracts revealed that there are at least 100 errors per 1,000 lines of code. Bambara has raised the intriguing question of who should be liable for such mistakes or errors? In traditional contracts, the parties would be able to sue the drafting lawyer for malpractice, could a similar lawsuit be brought against the coders of smart contracts for coding errors. These are novel legal issues that do not exist with traditional text-based contracts; it will be interesting to see how courts and arbitral tribunals will deal with such incidents.

(4) Ability to design complex contracts

As the adoption of blockchain spreads, smart contracts will become increasingly complex and capable of handling highly sophisticated transactions. Currently, coders are already stringing together multiple transaction steps to form more complex smart contracts. Nonetheless, we are many years away from code being able to determine more subjective legal criteria. For instance, as Stuart D. Levi and Alex B. Lipton have written, there is no yet code that would be able to determine whether a party satisfied a commercially reasonable efforts standard or whether a force majeure clause should be triggered or not.