REDESIGNING THE CONTRACT LAWYER

Dr. Adrian McCullagh Ph.D. LL.B. (Hons), B. App. Sc. (Computing)

16 Nov 2021

Abstract

Traditional natural language contracts have in general been written in a non-deterministic format, which will cause a problem for their conversion into smart legal contract code. To overcome this issue, it is suggested that where possible traditional natural language contracts should be written in a deterministic format. It is further suggested that contract lawyers be trained in developing pseudo-code so that smart legal contract coders are better placed to accurately developing smart legal contract code that correspond with the traditional natural language contract. In doing so, this paper will discuss some boiler plate clauses that should be considered when it is expected that the traditional natural language contract is to be embodied into smart legal contract code.

Key words: Smart Legal Contracts, Pseudo-code, contract law, deterministic language, non-deterministic language, boiler-plate clauses

Introduction

Since, early 2015, there has been a lot of published hype concerning the advancement of smart contracts. The first thing to understand is that most smart contract code is neither smart nor is it a contract recognised at law. This is not to say that some smart contact code is not aligned to a legally enforceable contract. A better view is that most smart contract code is really ***Transaction Event Response Code (TERC).*** That is, the code is directly connected to a transaction, and the code will respond to various events and append to a blockchain a record that relates to the particular transaction.

Much of the publications concerning smart contracts have been plainly incorrect and this has impacted how the legal community views the development of smart contract possibilities. There is a lot of scepticism in how the law will deal with smart contract deployments and interpretation. Some technologists[1] have opined that a smart contract can and should be able to deal with contract non-performance more efficiently by advocating in part that *code is law*[2]. But there is a legal maxim that *it is not possible to oust the court's jurisdiction*[3]. The court system has been designed as the final independent arbiter of fact and law, which in common law jurisdiction has evolved for nearly 1000 years since the Norman Conquest in 1066. As will be discussed below, arbitration clauses have had some impact into this general rule. But for policy reasons at least for the present time. Code is not law[4].

This was highlighted in 2016, where the DAO failure occurred on the Ethereum platform[5]. The failure of the DAO caused the promoters of the Ethereum platform to implement a hard fork in the Ethereum blockchain. There are probably many reasons as to why a hard fork was implemented. It has been speculated that this included the substantial loss of value by various investors which would have substantially undermined the confidence generally in the security and commercial viability of the platform. Afterall, the Ethereum platform had only been in production for approximately 12 months and this event could have scuttled the commercial future of the platform. Code is not law, but code can assist in the development of commercial transactions and has done so for the last 40 years[6]. As smart contracts mature so will the legal framework to support its advancement. It has been opined that one real benefit of Smart Legal Contracts is that they can substantially reduce contract administration costs especially for longitudinal contacts such as mortgages and leasing arrangements or even software licencing arrangements[7].

This short note will only be concerned with Smart Legal Contracts. Smart Legal Contracts are traditional legally enforceable contracts where some or all of the terms can be expressed in computer code[8]. The advent of blockchain platforms[9] and in particular the Ethereum platform has enhanced the imagination of many groups of technologists in creating self-executing applications that emulate the traditional performance of contracts. That is, a Smart Legal Contract can provide efficiencies in performance where automation is available due to some repetitive nature of the arrangement in performance[10]. For example, in a contract of personal services a Smart Legal Contract may not be able to fully account for performance but aspects of the performance such as payment for such services can be automated. It may also be possible to tie performance to the completion of milestones that trigger payment. But traditional contracts also possess a substantial vulnerability as lawyers are trained to draft in a non-deterministic style.

In addition to stylistic changes in drafting, the traditional contract may also need to have some special provisions which could become classified as standard boilerplate clauses to accommodate Smart Legal Contract code.

This short note is not designed to cover all aspects of designing a Smart Legal Contract as that would require a major book on this important topic. This short note proposes a redesigning of contractual styles to better accommodate the intricacies expected in the design of enforceable Smart Legal Contracts. It is proposed that law schools need to prepare the next generation of lawyers to draft contracts that can readily be converted into Smart Legal Contract code. In doing so, law schools need to arm the next generation of lawyers with the ability to write pseudo-code based on their natural language contracts so that properly trained coders can develop Smart Legal Contract code which should ultimately reduce contract management costs in the life cycle of the natural language contract.[11]

Finally, it is recommended that the Judiciary be trained in understanding how a Smart Legal Contract is developed and being able to link logically the Smart Legal Contract code to the original natural language contract. For the near future this may require the Judiciary to rely upon independent third party experts who are generally appointed as a "Friend of the Court" (Amici Curiae).

Drafting Traditional Contracts

Most law schools abide by some standard drafting structures/rules. These can be encapsulated through some simple but effective questions followed by some general comprehension rules.

The questions are:

1. Who,
2. What,
3. When,
4. Where, and
5. How.

The first 3 questions are really essential questions that the drafter should ask for every clause being drafted. That is, who is obligated to perform the activity, what is the activity, and when must the activity be completed. This will give certainty as to performance. But in some cases, where the activity is to be performed and how it is to be performed may also be important as between the parties to the contract. Noting that Lord Diplock[12] stated that every contract contains both primary obligations and secondary obligations. A primary obligation is a specific obligation that is self-contained in the contract itself, whereas a secondary obligation arises by law such as when there has been a material breach of the contractual terms and the defaulting party maybe be required to compensate the no-defaulting party through an action for damages. An example of a clause that involves not only the first question but also how the obligation is to be performed can often be found in a lease clause as follows:

> *The Lessee must pay the Rent to the Landlord by no later than the 15th day of each month in clear funds into the Designated Bank Account.*

This clause tells the parties who has the obligation to pay rent, what the activity is which in this case is the payment of rent and when the payment must be completed. The clause also includes how the payment is to occur which is by cleared funds in the Designated Bank Account.

Another example could be:

> a) *The contractor must attend the Principal's offices on a daily basis during business hours to provide the Service.*
>
> b) *Whilst the Contractor is located at Principal's offices the Contractor must follow all reasonable legal instructions issued by the Principal's representative.*

These two clauses not only provide for who, what and when but also include where the activity is to occur. Clause also includes a logical construct. All contracts basically involve two types of clauses:

- Absolute statements, and
- Logical statement.

Clause (a) above is an example of an absolute statement whereas clause (b) is a logical statement. This is significant for Smart Legal Contract coders as all computer programs only comprise a series of absolute statements combined with logical statements. The combination of these statements is regarded as a set of instructions given to a computer to produce some logical result. This is one of the important attributes that is common to traditional natural language contracts and smart legal contract code. But how would a computer deal with "reasonable legal instructions"? The use of a vague term such as "reasonable" could be interpreted as being vague for uncertainty but the courts have dealt with this matter consistently by utilising an objective test. That is, would a reasonable disassociated/independent person regard the instruction as being reasonable. As Lord Reed explained[13]:

> *The Clapham omnibus has many passengers. The most venerable is the reasonable man, who was born during the reign of Victoria but remains in vigorous health. Amongst the other passengers are the right-thinking member of society, familiar from the law of defamation, the officious bystander, the reasonable parent, the reasonable landlord, and the fair-minded and informed observer, all of whom have had season tickets for many years.*

That is, the reasonable person test is an objective test that is theoretically applied by an independent reasonable person of the general public. Note that the test does away with any bias since it is an object test and subjective aspects of the decision making is removed. It is expected that Smart Legal Contract code should also emulate the reasonable person test.

Another example is:

> *The Lessee must within a reasonable time after being notified by the Landlord that the Lessee is in default of the terms of the Lease rectify such notified default.*

The issue with this clause from a coding perspective is how would a computer determine what is a reasonable time. Now the reason why a drafter may use such language is that prior to execution of the lease the drafter will not know every type of default that could occur during the subsistence of the lease. Different defaults may give rise to different time frames. For example, if the default is a failure to pay the rent on time, then a short period may be imposed whereas if the default concerns some building rectification like fixing a hole in a wall in the premises, then a longer period may be regarded as reasonable, as it could involve the engagement of building contractors which could take time due to availability. Consequently, traditional drafting as taught in law schools involves both deterministic and non-deterministic language.

Finally, most law schools encourage their students to also utilise the following rules:

- Do not mix concepts in a single clause. That is, each clause should only deal with a single obligation so that it is easy to understand and interpret.
- Try to limit clause lengths to no more than 50 words at most. Any longer and it is possible that the reader will misinterpret the object of the clause. That is, who is do what and when?
- Try to draft in active text as opposed to passive text. This type of drafting links succinctly the subject to the object in the clause.

There are other more complex rules, but these simple rules can enhance the possibility of developing Smart Legal Contracts that could provide commercial efficiencies. But the combination of non-deterministic language with deterministic language will create some difficulties. The development of Smart Legal Contracts has centred upon the advent of blockchain technology. But a blockchain is not a panacea for all problems. The blockchain possess some very specific characteristics which can be used to solve certain some commercial problems efficiently.

The value of a Blockchain Solution

Blockchain technology and Smart Legal Contracts can provide advantages in reducing contract administration costs by automatically monitoring obligation activity and responding by recording immutable information on the blockchain[14]. The **FITS model** can assist in understanding where a blockchain and in particular a Smart Legal Contracts may be of assistance in providing improved contract management and thus reducing contract administration costs.

FITS is an acronym that stands for:

- **F**raud,
- **I**ntermediaries,
- **T**ransaction through put, and
- **S**table data.

Fraud

If the problem being solved involves a sector where there has been a history of fraud, then a blockchain and in particular Smart Legal Contracts may be beneficial. The automatic self-executing structure of Smart Legal Contract code provides an objective independent position in the performance of the contractual obligations. Further, since the Smart Legal Contract code will be published even in byte code which can easily be reversed engineered it will be a very naive programmer who tries to get away with fraud on a blockchain. Not only is the byte code published but every record appended to the blockchain will provide self-auditable transaction of evidence. Consequently, smart legal contracts can substantially reduce the likelihood for fraudulent activity as sufficient evidence will be recorded and this recording keeping should deter most fraudulent activity. Of course, as has been published, it is not possible to legislate against stupidity[15] and as such a certain criminal element could try to beat the odds unsuccessfully.

Intermediaries

If there is a noticeable delay from the start of a transaction to its completion due to the involvement of various intermediaries, then a blockchain and Smart Legal Contracts could be used to dis-intermediate some or all of the intermediaries. The fundamental basis behind the Nakamoto paper[16] was the creation of a transaction environment where ”A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution”. Prior to the advent of blockchain and in particular bitcoin, an online transaction would require the involvement of a “trusted” third party. These trusted third parties over time established themselves as necessary intermediaries who provided a valuable service to ensure that transactions concluded successfully.

The involvement of some intermediaries is historical in that prior to various technological advancements their involvement arose out of the tyranny of distance and thus the inclusion a trusted third party was needed. Coase in 1937 explained the economic value of the firm[17] and introduced the concept of transactions costs. Williamson[18] expanded on Coase’s work when he explained that transactions costs arose due to a governance of transactions where the tyranny of distance made it difficult for the parties to the transaction to adequately monitor the trust of each other. Hence, the involvement of third parties provided the governance framework to ensure the success of the transaction.

Further, the inclusion of intermediaries also incurred an additional cost which caused a corresponding delay in settlement of transactions. Consequently, transactions cost involves a cost in performing a transaction as opposed to other costs such as productions costs[19]. The advancement of fast telecommunications and the ability to remotely authenticate the parties involved in the transaction without the need of a trusted third party has created the position where it is possible to remove the involvement of these traditional trusted third parties.

A premise of a blockchain is that the technology creates a trusted environment even if the parties to the transaction do not trust each other. The technology creates a transaction environment where neither party to the transaction is in a position to defraud the other party. For example, prior to the creation of BITCOIN[20], all non-face-to-face financial transactions involved a third party such as PayPal to ensure that funds transfers were available, and neither party could defraud the other party.

If a transaction can be carried out using a blockchain and smart legal contracts, then an analysis of the transaction environment should be undertaken as it may be legally possible to remove some if not all intermediaries where they do not add value to the transaction. But some intermediaries even if they do not provide any real value may not be able to be removed because a regulator will not permit their removal. So, care needs to be taken when undertaking such an analysis as a regulator may insist upon the continued involvement of a particular trusted third party.

Transaction Throughput

Some transaction environments require fast transaction processing. For example, VisaNet handles on average of 150 million transactions every day (1700 transactions per second) and is capable of handling more than 24,000 transactions per second.[21] Bitcoin, on the other hand has been generally recorded as operating at 7 transactions per second[22] and Ethereum has been recorded at between 15 to 28 transactions per second[23]. The Ethereum platform is expected to substantially increase its processing speed when it releases its proof of stake solution, and the lightning layer 2 solution can increase the processing speed for Bitcoin[24]. There are other blockchain platforms which claim to have substantial throughput of transactions, but many do not currently match the throughput required as a maximum as what is available on the VisaNet platform. Further, transaction settlement time for the VisaNet platform is 1-3 seconds depending on telecommunications speed. According to alphazero the ranking of currently available blockchains indicate that further research work is required.

Table 1: Transaction throughput of some popular blockchain platforms.

Platform Name	**Transactions per second**	**Transaction time**
Bitcoin	7	60 mins
Ethereum	25	6 mins
Solana	29,000	2.575 seconds
Cardano	250	10 mins
XRP	1500	4 seconds
Avalanche	5,000	1-2 seconds
Dogecoin	33	6 mins
Bitcoin SV	224	60 Mins
Bitcoin cash	300	60-180 mins
Tron	2000	5 mins

https://alephzero.org/blog/what-is-the-fastest-blockchain-and-why-analysis-of-43-blockchains

Apart from the Solana platform none of the other deployed and active blockchain platforms would be able to handle a peak throughput possible via the VisaNet platform. Further, the 29,000 transactions per second attributed to the Solana platform is actually a theoretical throughput and not an actual throughput rate in a commercial environment. Consequently, if the throughput required for a particular environment is particularly high then a blockchain may not be a suitable solution. This restriction could impact the role of smart legal contracts in an operational environment.

Stable Data

Finally, if the environment involves stable date as opposed to volatile data, then a blockchain could be a suitable solution. Stable data means individual data records that is not altered frequently. An example of this would be land title data or identity data. Land title data is very stable. It is not uncommon for the ownership of land title data to not be adjusted for many decades especially inner-city land where ownership is usually held by some corporate entity and very little movement in ownership occurs on an annual basis. Further, in general terms identity information is very stable data. Apart from marriage adjustments either through getting married or through divorce the identity of a person rarely changes though certain attributes that evidence an identity may be adjusted from time to time.

The immutable characteristic of blockchain technology makes the stable data component as being very attractive for the custodians of this type of data. In many cases, the main purpose of this data is to ensure the availability and integrity of the data is maintained. Confidentiality may not be an issue as the information may be a public record as in a land title registry or a births, deaths, or marriages registry.

The above **FITS model** can assist in determining whether a blockchain would be commercially beneficial. If the deployment of a blockchain is beneficial then the next stage is whether a contract can take advantage of a Smart Legal Contract that could be deployed on the relevant blockchain. Noting that not all blockchains are designed to accommodate Smart Legal Contract code. If the selected blockchain can accommodate Smart Legal Contract code, then the drafter of any contract will need to ensure that their drafting is able to be later converted into some computer code that can operate on the selected blockchain.

Deterministic vs. Non-Deterministic Constructs

In computer science, a deterministic algorithm, given a particular input, will always produce the same output ensuring the computer goes through the same states to reach a result. But in the case of a non-deterministic algorithm, for the same input, the algorithm may produce different output on different runs.

As noted above, the lease rectification clause involved the correction of the default within a reasonable time frame. The difficulty with the clause is that there is no certainty as to when the landlord can terminate. What does "within a reasonable time" mean? How would a computer deal with this situation in a Smart Legal Contract environment?

The Smart Legal Contract code may revert to some machine learning application that will assess the facts and review all the known court cases of a similar fact nature and return a time frame that it determines to be reasonable in the circumstances. That is, the Smart Legal Contract may rely upon an oracle as being its source of truth to determine a reasonable time. This process could substantially slow down the processing. Further, every time there is a breach a different time frame could result. Hence, the language of the above lease default clause is non-deterministic. The involvement of an oracle adds complexity and cost. On the Ethereum platform as at the date of this paper every Smart Legal Contract instruction must be supported by Gas[25] which is measured in Gwei[26] and charged to cover the cost of running the Smart Legal Contract code. Some transactions may involve hundreds of instructions and as such the underlying cost can amount to a not insubstantial value in Gwei.

An alternative to non-deterministic language is for the drafter to concentrate on deterministic language in their contracts. Hence the construct of the lease could be:

> *The Lessee must within 10 business days after being notified by the Landlord that the Lessee is in default of the terms of the Lease rectify such notified default.*

This is an example of deterministic language as it is much easier for a computer to calculate time and react accordingly but what happens if a court determines that 10 business days is insufficient time to rectify all types of defaults. If deterministic language is to be utilised, then an analysis should be undertaken prior to execution to understand the various types of defaults and take the longest time frame no matter what the default may be. Alternatively, the drafter may specifically list various kinds of defaults and allocate a specific rectification period for each list default followed by a catch all period for anything not listed. The problem with this type of solution is that clients generally do not appreciate such thoroughness and could object to the length and complexity of the resultant document. Clients in general prefer succinct language and as a result they desire short length contracts which that can understand even though the final agreement may not cover all contingencies.

The contract may also provide for a nested solution which can be found in some restraint of trade clauses. Again, the more flexibility provided in the contract also makes the contract somewhat more complex to understand from the layperson's perspective. Consequently, care needs to be undertaken at the front end in analysing the contract's objective. Lawyers are familiar with this type of analysis as a similar analysis is taken when a contract involves a liquidated damages clause.

In general, an innocent party to a contract has the onus of proof that there has been default on the part of another party to a contract and that proof extends to the quantum of damage that the innocent party has suffered. It is possible for a party to a contract at the time of contracting to agree to a liquidated damages clause. The principal function of a liquidated damages clause is to pre-quantify the damages payable in the event of breach of the contract. The clause only becomes relevant once liability is proven or admitted. There is a fine line between a valid liquidated damages clause and a clause which could be assessed by a court as being excessive and regarded as a penalty[27]. A penalty is unenforceable on policy grounds and a such care must be taken at the time of contracting to genuinely pre-estimate the damage that could arise due to a default. Issacs J. in the case of *Boucaut Bay Co. Ltd v. Commonwealth*[28] explained the role of damages as follows:

> *To recover in an action for breach of contract damages more than nominal, those damages must be proved unless they are admitted. If they are admitted there is an end of it. But they may be admitted by a pre-assessment; and if a contract is produced in which a sum is named and that is relied on as a pre-assessment or pre-estimation of damages, the contract is looked at to see whether it really is so in order to satisfy the rule that damages must be admitted or proved.*

For a liquidated damages clause to be upheld, two conditions must be met.

a. The amount of the pre-estimated damages must roughly approximate the damages likely to fall upon the party seeking the benefit of the term as assessed at the time when the agreement of contract was entered into.
b. The damages must be sufficiently uncertain at the time the contract is made that such a clause will likely save both parties the future difficulty of estimating damages.

Any pre-assessment of a possible damages claim must be a genuine pre-estimate at the time of contracting otherwise the pre-assessment will be determined to be a penalty and thus unenforceable. It not uncommon for IT contracts to include a liquidated damages clause. These clauses require the drafter to seek information from his/her client as to the possible damage that could arise from a default so as to ascertain the possible quantum that should be incorporated into the contract. The benefit of a valid liquidated damages clause is that the innocent party can simply rely upon the value set out in the contract and thus potentially saves substantial litigation costs in not having to prove the value of the damage suffered. Another benefit is that the innocent party can elect to rely upon the liquidated damages clause and thus not be required to prove the actual damage or the innocent party can elect to not rely upon the liquidated damages clause and thus be required to prove the actual damage suffered. The innocent party may determine that the actual damage is far greater that what could be received by relying upon the liquidated damages clause. If a Smart Legal Contract has been deployed, then the right of election will be removed as the Smart

Legal Contract will automatically apply a liquidated damages instruction. Further, since the Smart Legal Contract code is automatically applied, the resultant record will be written to the blockchain which again causes an immutable record. If a Court later decides that the so-called liquidated damage provision applied by the Smart Legal Contract code is actually a penalty or is for an unconscionable amount, then the court may require some roll back of records which will not be easy to accommodate.

A similar investigation will need to be undertaken by a drafter in determining time frames required in a contract and thus avoiding any non-determinative language. Consequently, the same pertinent questions above still need to be answered but if a time frame is required words like "reasonable time" must be avoided if the parties are to take advantage of Smart Legal Contract coding. Consequently, new style of drafting will need to be utilised which will require a redesigning of the thought processes of the legal drafter. This may take some time as this may require a new intermediate step in the training of lawyers which it is suggested involves lawyers understanding how to produce pseudo-code that corresponds to the natural language contracts.

The next issue that arises is who is to write the Smart Legal Contract code and what will they work from to ensure that the Smart Legal Contract code corresponds to the tradition language contractual obligations.

Suggested Boilerplate clauses for Smart Legal Contracts

In addition to drafting clauses using deterministic language, the drafter may also need to consider the following boilerplate clauses[29] which will assist a court to better understand the operations of the contractual structure.

Entire Agreement Clause

Noting that in a traditional contract, the court will first confine its attention to the terms and condition detailed in the written document and will not in general look towards extraneous material. To ensure that the written agreement is self-contained it is not unusual for the contract to include an entire agreement clause. These clauses are designed to remove any collateral promises that may have arisen during negotiations. Since the new contractual framework will now involve a written word but also some executable code then the contractual framework will now involve at least 2 components namely the written word and the Smart Legal Contract code.

Hence its suggested that the drafter may wish to consider the inclusion of the following clause:

This agreement comprises the terms and conditions detailed in the document and also incorporates the operations of the Smart Legal Contract. If there is a discrepancy between the terms and conditions of this document and the operations of the Smart Legal Contract, then ***the terms and conditions of the document /Smart Legal Contract*** *will have precedence to the extent of the inconsistency.*

The parties should delete which ever has not been agreed; though it is suggested that the natural language version should take precedence.

Governing Law

The Smart Legal Contract code could be distributed and captured at each of the nodes involved in the relevant blockchain. The various nodes may reside across multiple jurisdictions. Consequently, a governing law clause is imperative.

Conflict in Language

What is to occur if there is a conflict between the natural language version of the contract and the Smart Legal Contract code. It is recommended that the natural language version should have a precedence clause. This will assist the court if such a situation arises, as a court is expected to determine which version has precedence. This has been covered above in the entire agreements clause.

Variations

The traditional natural language version should set out the procedure if the parties need to vary the Smart Legal Contract code. This should include how a kill switch may need to be activated and a good faith clause to only change the obligations to the extent of the agreed variation that matches the original intent of the arrangement.

Resolutions of Disputes

The parties may want to agree up front that if there is a dispute then the court can and should be encouraged to call upon a friend of the court to assist the judge in interpreting the Smart Legal Contract code. Traditionally this has been recognised as an amicus curiae (literally, "friend of the court"; plural: amici curiae). The Friend of the Court is someone who is not a party to a case but who assists a court by offering information, expertise, or insight that has a bearing on the technical aspects of the case.

Further, the parties may wish to engage in an alternative dispute resolution mechanism which they agree upfront. If there is a dispute, then they will engage an expert in pseudo code and smart contract coding to determine if the code accurately corresponds to the traditional language contact. If they cannot agree on the expert, then the expert can be appointed by the President from time to time of some nominated association which association has members who have the requisite skills to arbitrate or can provide some determination concerning the dispute.

Oracles

The parties may wish address the issue of the role of oracles and under what circumstance the information provided by an Oracle will impact the operations of the Smart Legal Contract code.

- The drafter should ensure that the parties understand who the accepted oracles are, and how they may impact performance.
- If the oracle determines that a force majeure event has occurred, then the operations of the Smart Legal Contract can be suspended pending reactivation by the parties. This will involve a good faith and cooperation clause to restart the Smart Legal Contract code.

The above are only a suggestion as to what the legal drafter may need to consider. Every case will be different and as such there should be a specific analysis undertaken by the drafter to ensure that their client understands the implications of the above. Further arising out of this analysis there may arise other terms that will need to be considered especially if the drafter has also developed some pseudo-code from which the final Smart Legal Contract Code is developed. As noted earlier, the incorporation of Smart Legal Contracts has substantially complicated the contractual arrangement and as such care must be undertaken to ensure that the arbiter of fact and law is able to understand what the parties have actually agreed to.

Pseudo-code Development

There are very few lawyers who would have sufficient skills to draft a natural language contract incorporating deterministic language and then converting the terms of the contract into executable Smart Legal Contract code[30]. It is suggested that an intermediate step now be incorporated. That step being the production of pseudo-code that will correspond to terms detailed in the natural language contract. The benefit of this intermediate step is that it will create a feed-back loop on the lawyers thought process which should identify any faults in the logic of the contract. It should result in tighter contractual obligations and thus be a more efficient process especially from a contract management perspective. But in suggesting this intermediate step the drafter will need to understand what pseudo-code is and how to translate a natural language contract into accurate pseudo-code which is capable of later being used to code the corresponding Smart Legal Contract code that accurately reflects the intention of the natural language contract.

Pseudo-code is an informal way to express the design of a computer program or algorithm. In computer science an algorithm is a well-defined finite set of rules that specifies a series of elementary operations to be applied to some data known as the input so as to produce in a finite time some output. By understanding the intricacies of an algorithm, it is possible to develop effective programs such a Smart Legal Contracts.

The aim of pseudo-code is to develop a document that expresses the solution being sought at a high level which can be used by a software coder to develop a detailed program which will become the Smart Legal Contract. Pseudocode often uses structural conventions of a normal programming language but is intended for human reading rather than machine reading. Consequently, a fundamental goal, when writing pseudocode, should always be to communicate the process clearly, with as little room as possible for misunderstanding.

According to Nicholas Bennett the writing of pseudo-code involves the following steps[31]:

1. *Avoid mixing and matching of natural languages (just as you should when naming variables and methods in program code). For example, if you're writing pseudocode in English, avoid including terms or variable names from other languages, unless there's a compelling reason to do so.*
2. *Strive for consistency, unless doing so would make the pseudocode less clear. For example, if in one part of your pseudocode you use a particular symbol or verb to denote assignment of a value to a variable, use that same symbol or verb throughout.*
3. *Where a step in the algorithm is expressed primarily in natural language, with few symbolic notations, use proper grammar. However, remember that mathematical expressions are often less ambiguous than natural language, even with correct grammar.*
4. *Since most programming languages borrow keywords from English, it's to be expected that pseudocode will resemble programming code to some extent. However, pseudocode should not be tightly coupled with any single programming language. Instead, it should employ control structures, verbs, and other keywords that are common to most programming languages. For example, most imperative programming languages have if-then-else, for-next, and while flow control statements; when combined with mathematical symbols for calculation of simple expressions and assignment of values to variables, these are sufficient for expressing virtually any algorithm.*
5. *It isn't necessary (or even desirable, in many cases) to have a one-to-one correspondence between each line of pseudocode and a corresponding line of program code, or between*

the symbolic names used in pseudocode and those used in program code. In particular, many common high-level operations (e.g., sorting values, searching for a minimum or maximum value from a list, opening a file to read a value from it) should generally be stated in a single line of pseudocode, rather than including all of the steps necessary to perform those operations in practice—unless, of course, the algorithm being described is for performing just such an operation.

6. *Use indentation to make any non-linear structure apparent. For example, when using an if-then statement to show that some portion of the algorithm should be performed conditionally, place the conditional portion immediately below the if-then statement, and indent it one tab stop to the right of the if-then statement itself. Similarly, if some portion of the algorithm is to be performed iteratively, under the control of a for-next or while statement, place that portion of the algorithm immediately below the for-next or while statement and indent it one tab stop further to the right. (By the way, these are good indentation practices for program code as well—even required in some cases.)*
7. *If an algorithm is so long or complex that the pseudocode becomes hard to follow, try breaking it up into smaller, cohesive sections, each with its own title; we can think of these sections as the pseudocode analogues to methods, functions, and procedures. When you do this, make sure that the pseudocode also includes an articulation of the higher-level sequence, showing the order in which, the more detailed sections should be performed.*

An example of pseudo code can be found in Donald Knuth's tome known as "The Art of Computer Programming: Volume 1 Fundamental Algorithms".[32] Professor Knuth sets out in the introduction some pseudo code dealing with the procedure for reading the set of books that comprise his scholarly publication as follows:

1. "Begin reading this procedure, unless you have already begun to read it. *Continue to follow the steps Faithfully.* …
2. Read the Notes on the Exercises on pages xv-xvii.
3. Set N equal to 1.
4. Begin reading Chapter N. Do not read the quotations that appear at the beginning of the Chapter.
5. Is the subject of the chapter interesting to you? If so, go to step 7; if not go to step 6.
6. Is N ≤ 2? If not, go to step 16; if so, scan through the chapter anyway.
7. Begin reading the next section of the chapter; if you have already reached the end of the chapter, however, go to step 16. …

16. Increase N by one, If N= 3 ,5, 7, 9, 11, or 12 begin the next volume of this set of books".

As can be seen, this pseudo code describing how to progress through this publication by Knuth is written in a natural language and is easy to understand. Incidentally, steps 8 through to 15 include such things as "step 15, go to sleep. Then wake up and go back to step 7." The point of this explanation is that pseudo code must be capable of being understood by a novice as well as an experienced coder so that the pseudo-code can later be converted in some smart legal contract code.

The application of Bennett steps will involve a realignment of the thought process in drafting traditional contracts; especially if the resultant traditional contract is to be encapsulated in some form as Smart Legal Contract code.

The Judicial involvement

As stated above, for policy reasons it is not possible to oust the court's jurisdiction. Hence, even though a Smart Legal Contract is designed to be some automated determiner of rights it is always possible for a party to a contract that has been in part or in full converted into Smart Legal Contract code, to object to the accuracy of the deployed code.

If there is a dispute involving a Smart Legal Contract that requires the adjudication of a court, then a further complexity arises. The Courts will approach the determination of any contractual dispute by initially identifying the relevant facts and then try to work out who had what obligations and who is at fault or whether there exists some extenuating circumstance that caused the failure of the contract. The complexity with Smart Legal Contracts is that as noted above the terms of the natural language version will not be self-contained but will include the Smart Legal Contract code which may or may not accurately reflect what the parties agreement.

Further, if the Court determines that the Smart Legal Contract code does not accurately reflect the agreed bargain, then the court has the difficult issue of how to reverse the already written records to the blockchain. If the court grants an interim injunction to stop the further processing of the Smart Legal Contract code, then it is important that the code have a kill switch mechanism built into the code and possibly a restart trigger in case the final determination concludes that the code is accurate.

The issue of Smart Legal Contract code disputes is a complex issue that really need substantial research as these disputes will most likely arise in the next 10 years.

Conclusion

The training of the next generation of contact lawyers should include the upskilling these lawyers to understand the difference between deterministic language and non-deterministic language. Further, some of these lawyers should be provided with sufficient skills to develop pseudo-code which can be utilised by a skilled Smart Legal Contract coder to develop a corresponding Smart Legal Contract which can be deployed on a blockchain.

Smart Legal Contracts are in this authors opinion an inevitable progression in commerce and as such the legal profession needs to accommodate this new development and in doing so the judiciary should be made aware of the advancement as in reality, they will be the final arbiter of the contractual rights as code is not law.

---

[1] Szabo, N., "(1997). "View of Formalizing and Securing Relationships on Public Networks" | First Monday". First Monday. doi:10.5210/fm.v2i9.548 (Accessed 11 November 2021)

[2] Lessig, Lawrence, "Code and other Laws of Cyberspace" New York Basic Books, 1999. Lessig does not advocate that code is law, instead he identifies that code can shape a user's ability to function by restricting what would be otherwise be an unlimited environment. That is, coders can and do impose their values upon users in cyberspace through their coding constraints.

[3] See Scott v. Avery 5 CAS 811 (HL 1856)
In Russell on Arbitration, a Scott v Avery clause is explained as follows:
"While parties cannot by contract oust the jurisdiction of the courts, they can agree that no right of action shall accrue in respect of any differences which may arise between them until such differences have been adjudicated upon by an arbitrator. Such a provision is often termed a Scott v. Avery clause."

[4] Hassan, S., and De Filippi, P. "The Expansion of Algorithmic Governance: From Code is Law to Law is Code" Field Actions Science Reports, s [Online], Special Issue 17 | 2017, Online since 31 December 2017, connection on 30 April 2019. http://journals.openedition.org/factsreports/4518 (accessed 11 Nov 2021)

[5] The DAO Attack: Understanding What Happened – CoinDesk
https://www.coindesk.com/learn/2016/06/25/understanding-the-dao-attack (accessed 11 Nov 2021)

[6] EDIFACT: United Nations/Electronic Data Interchange for Administration, Commerce and Transport (UN/EDIFACT) is an international standard for electronic data interchange (EDI) developed for the United Nations and approved and published by UNECE, the UN Economic Commission for Europe. This standard principally relies upon a set of standard communication messages and generally there will exist between the parties a trading partner agreement which will detail the various obligations of the parties.

7 Roeck, D., Sternberg, H., and Hoffmann, E., "Distributed Ledger Technology in Supply Chains: A Transaction Cost Perspective". International Journal of Production Research, https://doi.org/10.1080/00207543.2019.1657247 (Accessed 9 Nov 2021)

[8] The term "Smart Contracts" has been attributed to Nick Szabo from his 1997 paper : "The Idea of Smart Contracts" Https:// nakamotoinstitute.org/formalizing-securing-relationships
The concept of some computer code operating autonomously in the performance of a contract had been implemented for a number of years prior to the Szabo publication.
The EDIFACT standard which involved the drafting of trading partner agreements and then implementing the agreements utilising the EDIFACT standard has been around from the early 1970s.

9 The Ethereum Platform was the first blockchain platform that could accommodate Smart Legal Contract code but since the inception of the Ethereum platform in 2015, other Smart Legal Contract blockchain platforms have been established such as the Cardano platform, the EOS platform, the Solano Platform, and the Avalanche Platform to name just a few.

[10] Drummer, D; and Neumann, D "Is code law? Current legal and technical adoption issues and remedies for blockchain-enabled smart contracts". Journal of Information Technology. 35 (4): 337–360. https://doi.org/10.1177/0268396220924669 (Accessed 11 November 2021)

[11] Zheng, Z, Xie, S., Dai, H., Chen, X., Weng, J., and Imran, M., "An overview on Smart Contracts: Challenges, Advances and Platforms" https://arxiv.org/pdf/1912.10370.pdf (accessed 8 Nov 2021)

[12] Photo Production v. Securicor *[1980] AC 827*

> *A contract is the source of primary legal obligations upon each party to it to procure that whatever he has promised will be done, is done… Breaches of primary obligations give rise to substituted or secondary obligations… The contract is just as much the source of secondary obligations as it is of primary obligations."*

[13] Healthcare at Home Limited (Appellant) v The Common Services Agency (Respondent) (Scotland) [2014] UKSC 49.

[14] Temte,. M., "Blockchain Challenges Traditional Contract Law : Just How Smart are Smart Contracts?" 19, Wyo. L. Rev., pp87-117

[15] Australian Prime Minister Gough Whitlam and former Governor of Minnesota, Governor Jesse Venturer. https://www.brainyquote.com/quotes/jesse_ventura_583272?src=t_legislate

[16] Nakamoto, S., "Bitcoin: A Peer-to-Peer Electronic Cash System", first published on the Cryptography Mailing List on- Fri Oct 31 14:10:00 EDT 2008.
https://www.metzdowd.com/pipermail/cryptography/2008-October/014810.html
(accessed 12 November 2021)

[17] Coase, R., "The Nature of the Firm". Economica, 1937 Vol 4, Issue 16
https://doi.org/10.1111/j.1468-0335.1937.tb00002.x (Accessed 16 November 2021)

[18] Williamson, O., "The Economics of Organization: The Transaction Cost Approach". American Journal of Sociology. 1981 87 (3): 548–577
https://www.jstor.org/stable/2778934?seq=1#metadata_info_tab_contents (Accessed 16 November 2021)

[19] According to Williamson "transaction costs are the costs of running an economic system of companies, and unlike production costs, decision-makers determine strategies of companies by measuring transaction costs and production costs. Transaction costs are the total costs of making a transaction, including the cost of planning, deciding, changing plans, resolving disputes, and after-sales. Therefore, the transaction cost is one of the most significant factors in business operation and management"

[20] see note xvi.

[21] https://usa.visa.com/run-your-business/small-business-tools/retail.html
The MasterCard platform has a similar throughput whilst American Express is slightly lower though still substantial compared to Bitcoin and Ethereum transaction throughput.

[22] Hou, B., and Chen, F., "A Study on Nine Years of Bitcoin Transactions: Understanding Real-world Behaviors of Bitcoin Miners and Users", 2020 IEEE 40th International Conference on Distributed Computing Systems
http://bit.crc.lsu.edu/~fchen/publications/papers/icdcs20-bitcoin.pdf

[23] Anwar, S., Anayat, S., Butt, S., and Saad, M., "Generation Analysis of Blockchain Technology: Bitcoin and Ethereum", Int. Journal. Information Engineering and Electronic Business, 2020
DOI: 10.5815/ijieeb.2020.04.04

[24] It is very difficult to ascertain an accurate assessment the Transaction throughput on the Lightning network. Some sites state that the throughput in between 30-60 transactions per seconds whereas others have stated that the speed CAN be between many hundreds of thousands to millions of transactions per second. The later is only theoretical as the processing speed is very dependent upon the number of nodes involved.

[25] On the Ethereum blockchain, gas refers to the cost necessary to perform a transaction on the network. Miners set the price of gas based on supply and demand for the computational power of the network needed to process smart contracts and other transactions. Gas prices are denoted in small fractions of ether called gwei.

26 The digital token used on the Ethereum platform is the Ether. The atomic value of an Ether is known as a "Wei". There are 10 ^ 18 wei in an Ether. Now Gas is measured in Gwei (Gigawei) which is 10 ^ 9 wei.
27 See Paciocco v Australia and New Zealand Banking Group Limited [2016] HCA 28.
28 (1927) 40 CLR 98
29 Anuruddika, S.M. and Senevirathne, G., "Contracts formed during Frustrations and Force Majeures: An Anti-Crisis Shield for Consumer Protection against Boilerplate and Limited Liability Clauses".
30 Richards, D., "Teaching Non-determinism",
ACM SIGACT News, 2021 - dl.acm.org.
31 Bennett, N., "Introductions to Algorithms and Pseudo-code",
https://www.researchgate.net/publication/309410533 (accessed 11 November 2021)
DDC Style Guide: Pseudocode: Coding conventions for the Deep Dive Coding Java + Android Bootcamp.
https://ddc-java.github.io/style-guide/pseudocode.html (Accessed 11 November 2021)
[32] Knuth, D,. "The Art of Computer Programming: Volume 1 – Fundamental Algorithms" 3rd ed. Addison Wesley,2 009