Table of Contents

Chapter 3 ..... *2*

ABSTRACT *2*

INTRODUCTION *3*

Technical Designs 4

Crowdfunding the difference 5

So, what is a contract, anyway? 7

CONCLUSION *9*

Evolutions 9

Incorporations 10

Tangentials 12

Biography *14*

REFERENCES *15*

CHAPTER 3

On the Intersection of Ricardian and Smart Contracts

Ian Grigg
Financial Cryptographer

ABSTRACT

Bitcoin's inclusion of the smart contract form invented by Nick Szabo has thrust this design into the forefront (Szabo, 1994). An alternate design, the Ricardian contract designed by the present author, is currently used by a few innovatory systems such as Mattereum, OpenTransactions, OpenBazaar, Askemos and CommonAccord (WebFunds, 2022).

Mark Miller sees these as two halves of a split contract, but a more popular view is to see it as an either/or choice (Walker, 1994). Which should a designer choose? Smart or Ricardian?

An analysis of both, compared, reveals that they are totally different, which begs the question, what is a contract, anyway? A contract in law is an agreement of the parties, and both of these designs fall short of entirely capturing that agreement. More importantly, the Ricardian contract captures the essence of the understanding of agreement, whereas the smart contract captures the performance of that agreement; they are two phases of a wider multi-phase project.

Then, our future goal is to incorporate both the prose of the Ricardian and the code of the smart contract, towards a fuller capture of the entire lifecycle of the contract.

Keywords: Ricardian contract, smart contract, Bitcoin

INTRODUCTION

The Ricardian contract finds its origins in a 1995-96 project to digitally trade bonds on the Internet (Grigg 2022). We found that it was difficult to completely describe bonds to traders, because, unlike say currencies, bonds are often highly but not exactly alike. As well as small differences in salient details such as rates of interest and payment schedules, each new bond's legal terms and conditions would suffer from minor edits as new information came to light.

In contrast, conventional computing logic would have it that a trading platform should present simplified virtual twins of the bonds, as summarised and stored in databases - just the salient details such as name, face, coupon etc - and leave the traders responsible for understanding the unpresented differences. Our view was that, as we were designing for an open world unsupported by institutions, anything less than full information was risky to all participants.

Something more was needed, and a deep dive into the nature of bonds revealed that they were contracts in the legal tradition, which was a nuance perhaps internalised and forgotten by the trading world. Therefore, we reasoned, if we could issue contracts, we could issue bonds. To fully describe such financial instruments, the system should not trade virtual or limited copies of contracts, but should rather find a way to trade the very contracts themselves.

Then, the Ricardian contract is a document that captures the legal contract that, in its issuance or sharing, expresses a financial instrument in full. It is a document that includes all of the prose that the issuer presents to its traders, and some technical parameters that the program needs to know as well. On the whole, the Ricardian contract looks like a contract, as it is intended to be familiar to people, not machines.

In contrast, a smart contract is a metaphor for the execution of a contract that is mediated, directed or actioned by a computer program. As Szabo (1994) noted, even in the late 20th century more and more contracts were being actioned in part in an automatic fashion. *"Some technologies that exist today can be considered as crude smart contracts, for example POS terminals and cards, EDI, and agoric allocation of public network bandwidth."* As a computer program, a smart contract has 2 key differences to other programs. Firstly, a smart contract is not under the direct control of its participants (parties), and instead it operates at least in part autonomously, typically on a blockchain. Importantly, this means it cannot be stopped or changed. Secondly, the smart contract expresses, holds and deals in value. Commonly called crypto today, the program can in principle issue, hold or transact in any form of financial instrument that can be coded up.

These two approaches are radically different. One is a document and the other is a program, yet both call themselves contracts. To choose between these two designs, or to bring them together, a deeper dive is needed.

This chapter seeks to do that. First, we examine both in depth. Then we look at what a contract is, and compare that back to the designs. This allows us to place the two designs in the context of a contract lifecycle. Finally, we close with some newer developments.

Technical Designs

Following Grigg (2004),

> *"A Ricardian contract can be defined as a single document that is a) a contract offered by an issuer to holders, b) for a valuable right held by holders, and managed by the issuer, c) easily readable by people (like a contract on paper), d) readable by programs (parsable like a database), e) digitally signed, f) carries the keys and server information, and g) allied with a unique and secure identifier."*

For our purposes, the Ricardian contract is just a document with a few differences: it can be read by humans, as well as by machines. The former implies prose text of the normal legal form:

> *"That, this Bond will pay a rate of interest of {{interest}} once per year on the 1st of June, until termination."*

That latter means typically that embedded in the human text there are parameters or potentially code of meaning to the machine, things like:

- *face = $100*
- *interest = 5%*
- *coupon = {{face}} * {{interest}}*

Where, the first two above set some basic variables and the last has some simple code. Thus, this document, static as it is, communicates basic intent to its parties, as well as instructions to a program.

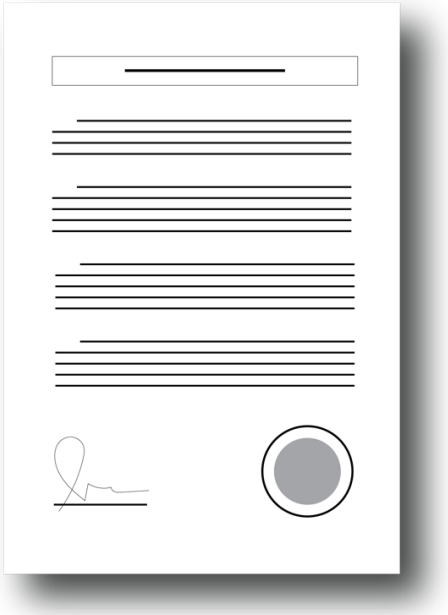

Figure 1. A "prose" contract.

The Ricardian contract works well to describe and differentiate shares, bonds, derivatives, more or less anything that means something to a human. Indeed, a Ricardian Contract is conceptually unlimited in the richness of semantics, and Askemos, CommonAccord, OpenBazaar, OpenTransactions and Mattereum among others have extended it in ways beyond the original context of issuance of a financial instrument.

Compare and contrast to Bitcoin and we can see that there is no writing at all — Bitcoin delivers what might be seen as a null contract, one with zero semantics. In contrast, it introduced Nick Szabo's smart contract into production for the first time, as a design to capture the flow of actions and events (e.g., delivery of payments) within the performance of a contract.

Following Nick Szabo (1994),

> *"A smart contract is a computerized transaction protocol that executes the terms of a contract. The general objectives of smart contract design are to satisfy common contractual conditions (such as payment terms, liens, confidentiality, and even enforcement), minimize exceptions both malicious and accidental, and minimize the need for trusted intermediaries."*

Crowdfunding the difference

An example will help. Imagine a crowdfunding supported by a smart contract: A potential project could mount a smart contract in a chain. Crowders can pay contributions to the smart contract. When the smart contract reaches its stated close time, it has a binary decision, a choice of two options. If, in one case, the threshold of value has been reached by total contributions, it pays the entire amount to a project account. If, in the alternate, the threshold has not been reached, the smart contract returns (pays out) each contribution back to the source crowder's account.

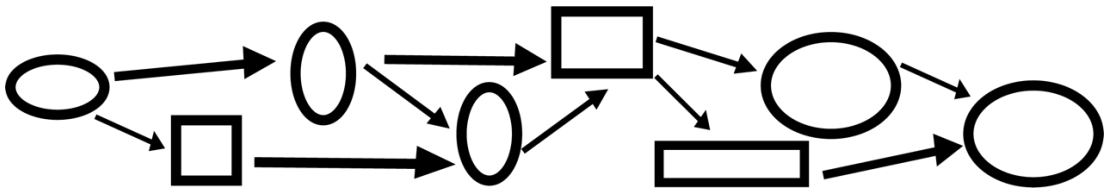

Figure 2. The events and actions of a state machine.

These flows and events can be captured by the idea of the smart contract and could substantially free up the infrastructure needed to cope with this design. Pre-tech, we would have had to employ bank accounts, escrow agents, clerks and cheques, envelopes and paper to manage all this. Even post-web we'd need a small army of programmers and interfaces into payment systems and websites. The hope of smart contracts is to outsource all that to a specialist developer who can insert the entire code into the mediating agent (e.g. a blockchain) for flexibility and scalability.

Smart contracts then can capture unlimited richness in flows of actions and events; computer scientists might prefer to recognise this as *a state machine with money*, see figure 2.

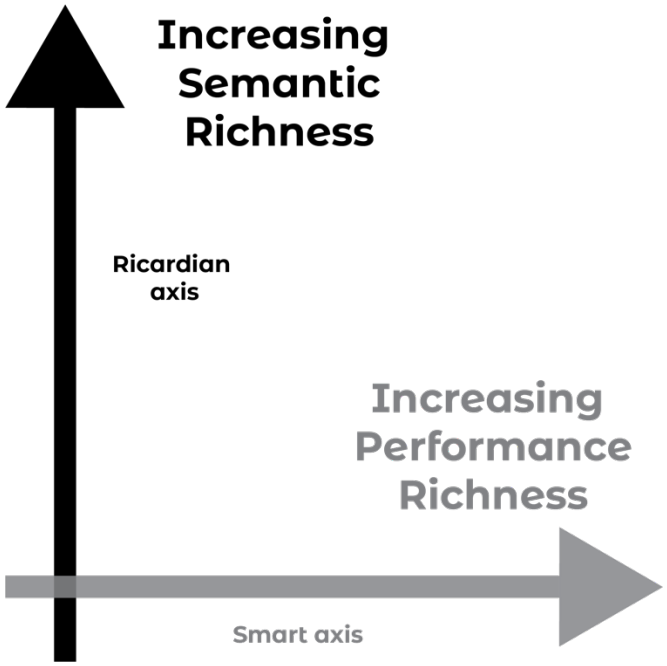

Figure 3. Legal semantics *versus* operational performance.

But what is not captured is the semantics: what is the project? What will it do? How do we know that the contributions are going to our project to design the $100 solar widget to reverse global warming? Or the pension fund for a drug kingpin? How do we even know it is a crowd funding? What do we do when our money doesn't come back or our project deliverables fail?

A simple solution could be to point the smart contract at a URL. But the URL can be intercepted, and the contents of a web page can be changed, or even disappear. Within the webpage there can be a confusing array of claims and counter-claims, and mingling of projects. This arrangement does not reliably capture semantics except in the accidental circumstance that lawyers audited the approach up front — accidental because no crowd funding project would survive the billing process, and no crowder would contribute to such a tedious webpage.

In contrast, a Ricardian contract captures the meaning of the flows in a way that is secured to your actions within the contract. Yet, it says little about how the flows carry forth in any particular cycle, indeed, because it is mostly words created in advance of the action, it fails to capture any flows at all. Historically, Ricardian Contracts were used to support your basic 3 party payment systems: Alice pays Bob through Ivan the Issuer, and note that even that was assumed, not specified in the contract.

The smart contract and the Ricardian contract are therefore doing different parts of the same process. Performance and semantics are approximately orthogonal, so we should be able to construct a graph of two axes, see figure 3 above.

There is a place in human interactions for both, and probably both would be useful in a wider system. Where the challenge lies today is how to combine these approaches so that the technology can better help humans to mediate more complicated agreements with success and a desire to engage again.

In a very stylised sense, we can also see something of the same sense of differing richnesses in classical finance systems, figure 4 below. On the vertical axis we see how many different contracts are in use, and how complicated each can be. For example a typical forex system handles about 20 base currencies,

and an OTC derivative can run to 300 pages. And on the horizontal axis we can see complexity in the performance of the deal. The stock exchange involves conceptually 4 payments, 2 inwards and 2 outwards. Mortgages involve hundreds of payments over their lifetime, and securitization lumps all those into a basket that slices off dividends to holders of the basket. Performance of these things is very complicated (Grigg, 2010).

The national currency, as a paper banknote within its country, sits at the origin, position 1,1, in that there is only one permitted, and it has only one simple hand to hand action.

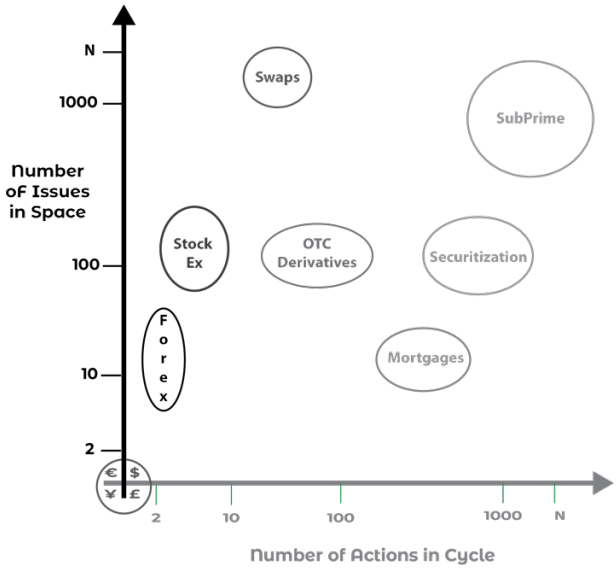

Figure 4. Semantically distinct instruments *versus* operationally complex performances.

So, what is a contract, anyway?

What's going on here? A big part of this confusion is an overloading of the term contract. In the Ricardian case, the thing in question is a document. In the smart case, it is a machine to organise and control the arrival of events and initiation of actions.

As it turns out, in law, neither is precisely the contract. More formally, the contract is the agreement between the parties. A document might represent a good stab at recording this agreement, but it can be augmented by side documents, so while there is often a document called "the contract," this is actually quite hopeful. We might better understand it as "the best and hopefully dominating recording of the agreement."

How do we resolve the difference between the document and the agreement? We go to court, and the court will decide what is the contract after analysis of all the evidence. A court is the power, and simply put, it is free to strike down clauses, add clauses, or indeed send people to jail.

Hence, a Ricardian contract isn't the contract but merely our best efforts at creating a single document that dominates the contract as found by the court.

Meanwhile, the smart contract is really the machine to perform some parts of the contract. As a smart contract is written before it all starts, it is presumably part of the wider agreement, so the court will

likely find the source code as much a part of the contract as any other document. Although, whether the court can read the source code is another matter.

How did society organise all this before the technologies of cryptocurrencies muddied the waters? The court would read the written document looking for indications of performance. It would expect the parties to have done some or all of the stated actions as per the writing, and ask for evidence of these actions.

As smart contracts seek to capture the intent into code, and evidence any actions through it, they therefore relieve humans of much of the drudgery of doing that which they already agreed to do, are intending to do, and may need to prove to the court that which has been done. Where we're left with is that the smart contract isn't entirely fulsome, as it fails to carry the richer framework of words. Likewise, the Ricardian contract is a clumsy vehicle in which to insert difficult code. In this contest, it isn't even a draw, the two devices are fighting to pull together: Both are trying to improve our agreements at different points and in different ways, within the overall framework of a contract in law.

In practical terms we can now look at the original Ricardian system as a system with infinite semantic ability but capable of handling by assumption only one form of action — perhaps the Alice to Bob payment (Grigg, 2000). Whereas Bitcoin can handle a multitude of smart-enabled transaction forms but in only one semantically trivial unit: assumed to be the bitcoin (Nakamoto, 2008). Those axes in the figures cross at 1!

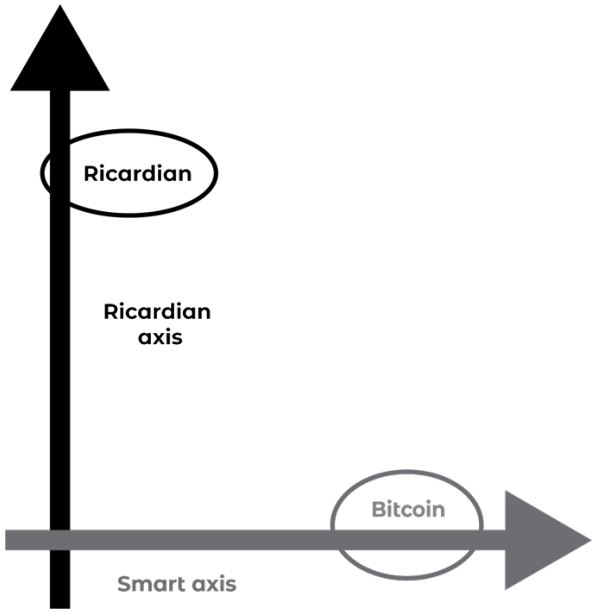

Figure 5. Ricardian *versus* Bitcoin.

CONCLUSION

Evolutions

Today, we've moved forward.

- OpenTransactions added server-side smart contracts to its technology permitting many more transaction types (Odom, 2015).
- Askemos clients run agreed smart contracts and insert events and state into a merkle tree as they happen (Wittenberger, 2002).
- OpenBazaar composes Ricardian contracts into trade cycles of invoice, acceptance, payment etc, thus also handing many conceptually complicated transaction types (Sanchez, 2014). In concert, CommonAccord places small smart contracts with Ricardians and then composes these pairs into larger agreements.
- Mattereum (2022) is using the form of Ricardian contracts to describe and insure physical artifacts, and wrap that all within smart contracts known as NFTs (for non-fungible transactions).

Meanwhile, on the Bitcoin front, exasperation with the one unit led to many altCoins which were essentially direct copies of the code with some params changed.

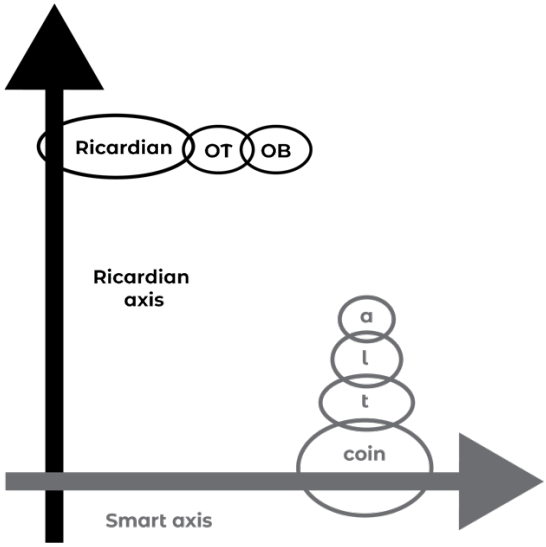

Figure 6. Evolution.

This latter approach by the Bitcoin community led to unfortunate consequences, which can now be interpreted in the context of semantic poverty. As more and more altCoins piled in with inadequate expressions of meaning, the field became noisy. When a booming investment field becomes noise-rich and semantically poor, there is plenty of scope and space for charlatans to siphon off funds of the ignorant investor. altCoins inevitably drift to noiseCoins, and more and more of them ended up looking like one-way contributions to the memory of the late great Charles Ponzi.

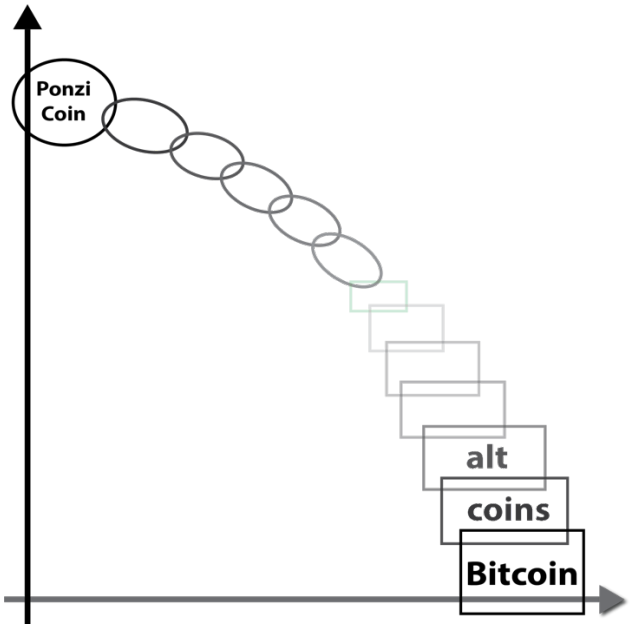

Figure 7. altCoins evolve to anti-semantics.

In contrast, we could also speculate that simple payment flow systems have not managed to garner enough of the total transaction flow, and thus have enjoyed limited success. If they can expand to handle more richness in performance of contracts, success may be easier.

Incorporations

We have seen that the real challenge between smart contracts and Ricardian contracts or legal documents is not to choose, but to incorporate, to encompass both elements into one contract (Miller and Stiegler, 2003). Payment systems should follow the lead of the innovators mentioned above and consider merging the smart contract ideas in to achieve better performance flexibility. How this is done is well beyond the scope of this chapter, and methods remain hotly debated in e.g., the emerging CBDC literature.

Likewise, the cryptocurrency world would benefit from adding the semantic richness of legal documentation into its service. Several efforts have explored the combination of the two.

Clack et al (2017) discuss overarching ways to generate code and prose elements from one source document but prose and code are so far apart, semantically, that treating them as one document remains an open research project for the foreseeable future (see figure 8). More likely, separated code and prose objects can be authored together and then joined by means of references (secure addresses such as hashes or blockchain transaction IDs) from one to the other; preferably with both pointing to each other, in mutual form. This is trivially accompanied by signing off on the prose with the address of the smart code embedded, and then posting the address of the prose Ricardian contract directly into a running smart contract.

Ethereum is taking an approach it called Natural Specification Format in its smart contract programming language Solidity (Karapetsas and Woods, 2014). In-code documentation is augmented by marking comments with /// which can then be parsed and analysed to describe what the contract does and to keep the user informed during contract performance.

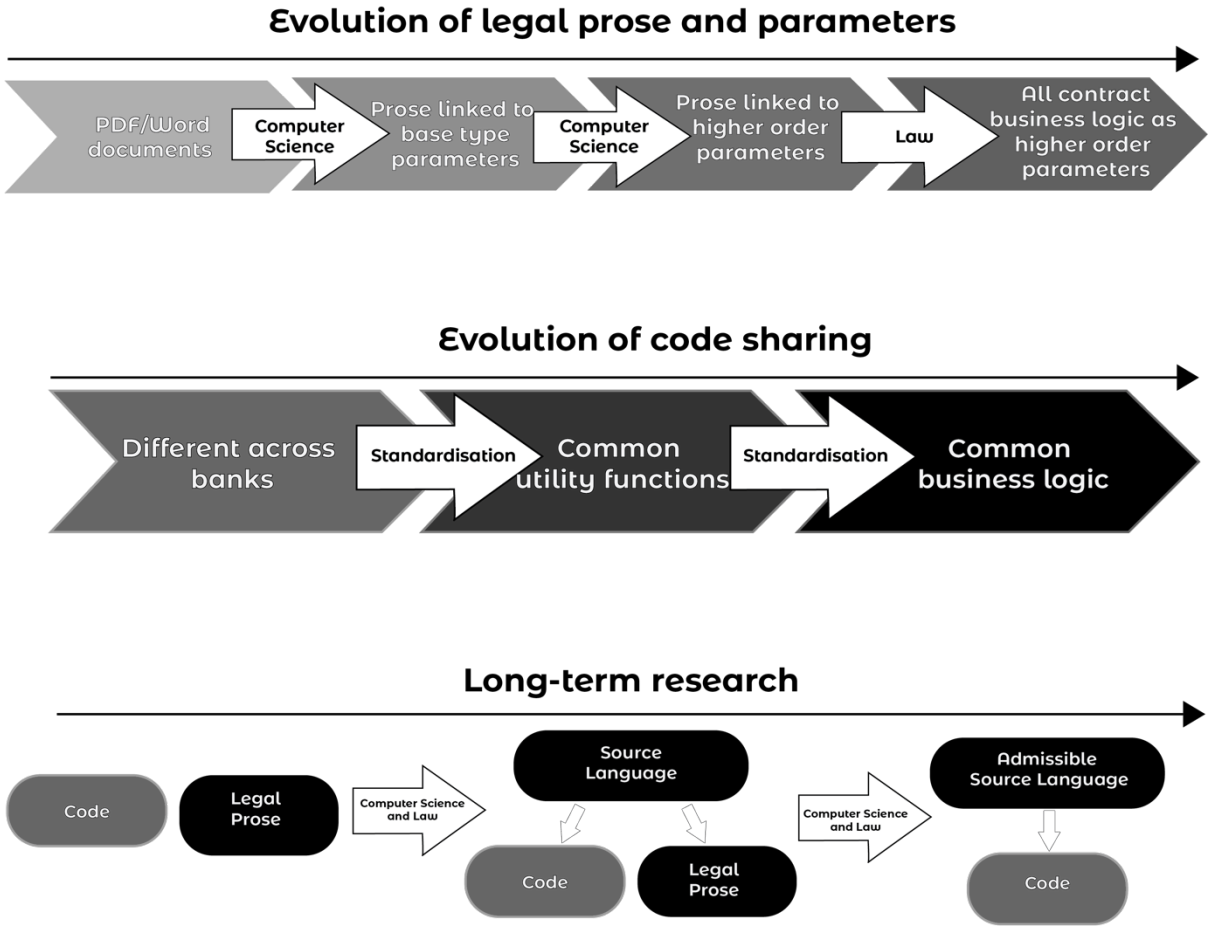

Figure 8. (from Clack et al 2017): Potential evolution of aspects of legally-enforceable smart contracts in three streams: legal prose and parameters, code sharing and long-term research.

From the Bitcoin world, Sidechains (Back et al, 2014) can add issuances with approximately these changes, being (a) create a new genesis transaction for an issuance, as distinct to the genesis transaction for a new blockchain (Friedenbach and Timón, 2013); (b) tie the Ricardian contract into the issuance genesis transaction, (c) identify the chain and the issuance by means of hashes over the combined genesis, and (d), change the transaction record to incorporate the two new identifiers, issue + chain, when expressing a movement of value from one key to another.

Projects such as CommonAccord (see figure 9) and Mattereum are using the hybrid text & code form of contracts to compose new constructs such as reusable contracts and trade patterns (Hazard, 2022). See also (Grigg, 2015) for a framework to identify a blockchain.

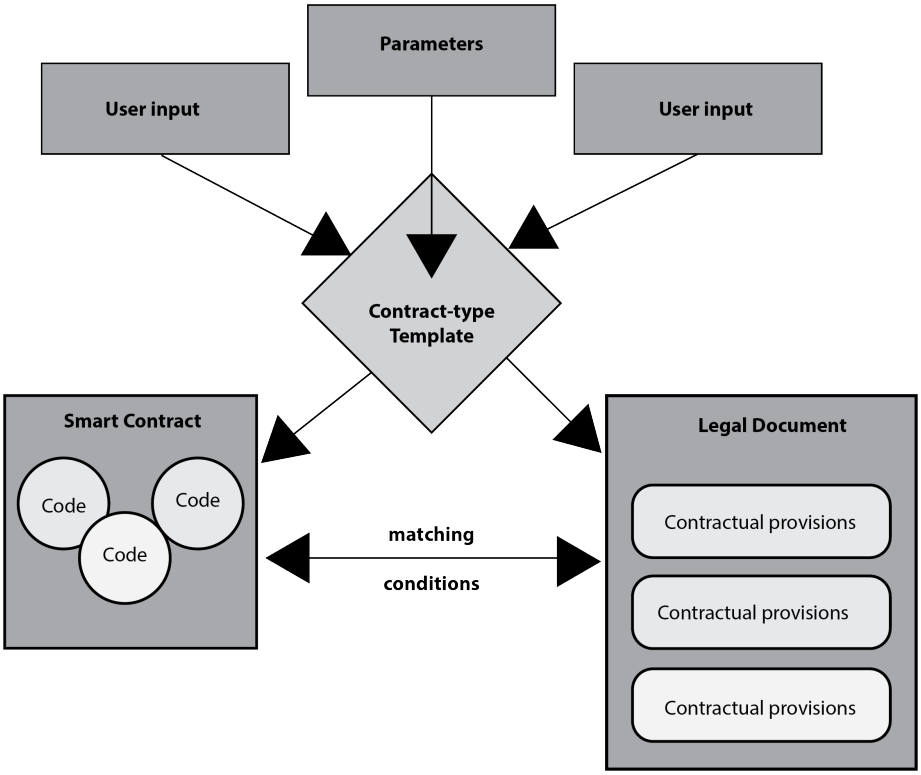

Figure 9. Coupling code snippets to clauses and composing upwards.

Tangentials

A couple of legal points are tangential, but worth mentioning.

Firstly, if we treat the contrast between the two as different phases of the contractual life cycle, then a third phase is missing and may demand some comment: dispute resolution. (Britton 2016) In this more complete view, most contracts, say the 99%, will complete without trouble. Yet some small proportion will end up in trouble, and a dispute arises. Thus, the third phase of contract lifecycle is called into action, albeit rarely: Dispute Resolution. This event is trivially defined in the prose, as the well-known dispute resolution clauses are meat and drink for the classical contract authors. Yet, the smart contract code could equally well participate, and as Szabo (1994) notes, "related economic goals include lowering fraud loss, arbitration and enforcement costs, and other transaction costs." It could be defined in code with a simple parameter that was signposted as such:

- *Dispute_Resolution = "Courts of London, England, to English Law."*

Or, entirely possible is a secure link such as the hash of an online Forum's Rules for Arbitration in Ricardian format or a URL to a website offering such services.

Finally, if there are two artefacts that present themselves as important parts of the lifecycle of the contract, which leads in the event of mismatch or collision in events? By way of logic, it is the Ricardian contract (Britton 2016). Fundamentally, the final word on what the contract is falls to the court. But the court is human, and cannot typically read code. Even if we could rely on the judge to be a programmer (not out of the question in an Arbitration Forum) we would still need to reach the appeal bench, the jury and later readers of any precedents. Prose naturally wins there. Secondly, as prose, it can state that it is

the contract, and further it can state that the terms of the code dominate, or otherwise; whereas code cannot state any sort of thing. Thus, the Ricardian prose leads, because it has the flexibility to pass leadership to the code, when the code is mute on matters outside its mechanical domain.

This article received useful comments from Florian Glatz, James Hazard, Jörg F. Wittenberger, Preston Byrne, Roger Willis, and Stephen Palley. Many thanks to Arthur Doohan, Eva Porras and André Bonello for final production and proofreading.

BIOGRAPHY

Ian Grigg is a recognised inventor and one of the longest standing experts in the blockchain space. For over 25 years, he has architected significant fintech systems. In 1995, he pioneered the world's first cryptographically secured digital cash and assets exchange. He is the inventor of the Ricardian contract, a way of expressing issuable and tradable financial obligations as a contract online. He co-invented triple entry accounting, a concept that does for events between firms what double entry accounting did to accounts inside the firm. He has worked with several notable initiatives ranging from protocols to smart platforms to enterprise solutions, including e-gold, EOS, Mattereum, R3 and Knabu. Ian dedicates his primary efforts to innovations in social savings and identity.

REFERENCES

Back, A., et al. (2014). Enabling Innovation with Pegged Sidechains. *Blockstream,* https://blockstream.com/sidechains.pdf

Britton, M. and Grigg, I (2016) Legal and Dispute Resolution Frameworks. R3.com

Clack, C., Bakshi, V., Braine, L. (2017). Smart Contract Templates: foundations, design landscape and research directions. arxiv.org https://arxiv.org/abs/1608.00771v2

Friedenbach, M. and Timón, J. (2013). Freimarkets: extending bitcoin protocol with user-specified bearer instruments, peer-to-peer exchange, off-chain accounting, auctions, derivatives and transitive transactions. *Freicoin,* http://freico.in/docs/freimarkets-v0.0.1.pdf

Grigg, I. (2000). Financial Cryptography in 7 layers. Frankel, Y. (ed) *Proceedings of Financial Cryptography: 4th Annual Conference, FC 2000.* Springer-Verlag. Paper at http://iang.org/papers/fc7.html

Grigg, I. (2004). The Ricardian Contract. *First IEEE International Workshop on Electronic Contracting 2004.* Paper at https://iang.org/papers/ricardian_contract.html

Grigg, I. (2010). A small amount of Evidence. (In which, the end of banking and the rise of markets is suggested.). *Financial Cryptography blog.* 2010 http://financialcryptography.com/mt/archives/001299.html

Grigg, I. (2015). Sum of all Chains - Let's Converge. CoinScrum / Proof of Work's Tools for the Future. *Financial Cryptography blog.* 2015 http://financialcryptography.com/mt/archives/001556.html

Grigg, I. (2022). Why the Ricardian Contract Came About: A Retrospective Dialogue with Lawyers. Jason Grant Allen & Peter Hunn (eds), *Smart Legal Contracts - Computable Law in Theory and Practice* 2022 Oxford University Press, Chapter 5.

Hazard, J. Common Accord. *CommonAccord.org,* http://www.commonaccord.org/

Webfunds. (2022). Implementations of Ricardian Contracts. *webfunds.org*, http://webfunds.org/guide/ricardian_implementations.html

Karapetsas, L. and Woods, G. (2014). Ethereum Natural Specification Format. *GitHub,* https://github.com/ethereum/wiki/wiki/Ethereum-Natural-Specification-Format

Mattereum (2022). Digital Representation and Ownership of Physical Assets. *LawtechUK* https://resources.lawtechuk.io/files/digital_representation_ownership_physical_assets

Miller, S. M. and Stiegler, M. (2003). The Digital Path: Smart Contracts and the Third World. Birner, J. and Garrouste, P. (Eds) *Markets, Information and Communication: Austrian Perspectives on the Internet Economy*, 2003 Routledge https://books.google.com/books?id=x-FQrKrjgcYC&hl=en Paper at http://www.erights.org/talks/pisa/paper/

Nakamoto, S. (2008). Bitcoin: A Peer-to-Peer Electronic Cash System. *CraigWright.net* https://craigwright.net/bitcoin-white-paper.pdf

Odom, C. (2013). Sample Currency Contract. *opentransactions.org*, http://opentransactions.org/wiki/index.php/Sample_Currency_Contract 2013

Odom, C. (2015). Open-Transactions: Secure Contracts between Untrusted Parties. *opentransactions.org*,, http://www.opentransactions.org/open-transactions.pdf

Sanchez, W. (2014). Ricardian Contracts in Open Bazaar. *GitHub Gist*, https://gist.github.com/drwasho/a5380544c170bdbbbad8

Szabo, N. (1994). Smart Contracts, originally at http://szabo.best.vwh.net/smart.contracts.html now at https://web.archive.org/web/20011102030833/http://szabo.best.vwh.net/smart.contracts.html

Walker, J. (1994). Understanding AMIX. *The Autodesk File 4th edition*.

Wittenberger, F. J. (2002). Askemos - a Distributed Settlement. *CiteSeerX*, http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.11.5050