On the intersection of Ricardian and Smart Contracts

Ian Grigg

February 2015 [wip]

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

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 [AMIX]. So let's try that out: which should you choose? Smart or Ricardian? Let's find out.

The original Ricardian Contract captures the legal contract behind an issuance, which is generally a simple payment system moving money from and even what will be one account to another. It is a document that includes all the good stuff people need to know, and a little technical stuff the program needs to know as well. On the whole, the Riccy looks like a contract, by design. It's intended to be familiar to people, not machines.

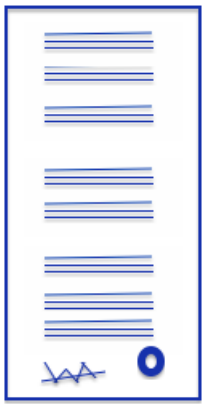

Figure 1. A "prose" contract.

In terms of defining differences, 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 OpenTransactions, OpenBazaar, CommonAccord and Askemos among others are extending it in ways beyond the original context of issuance.

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 introduces the smart contract which is a design to capture the flow of actions and events (e.g., delivery of payments) within the performance of a contract.

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.

These flows and events can be captured within the smart contract idea, 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 (blockchain or at least merkle tree server) 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.

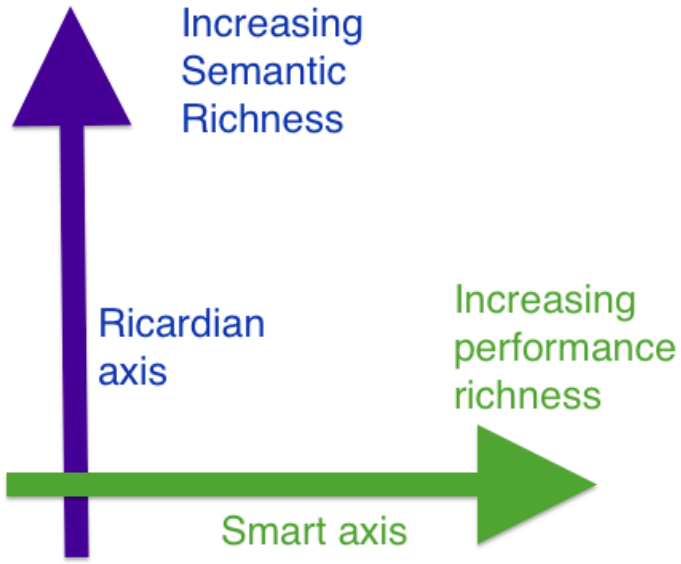

Figure 3. Legal semantics *versus* operational performance.

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 [Evidence].

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.

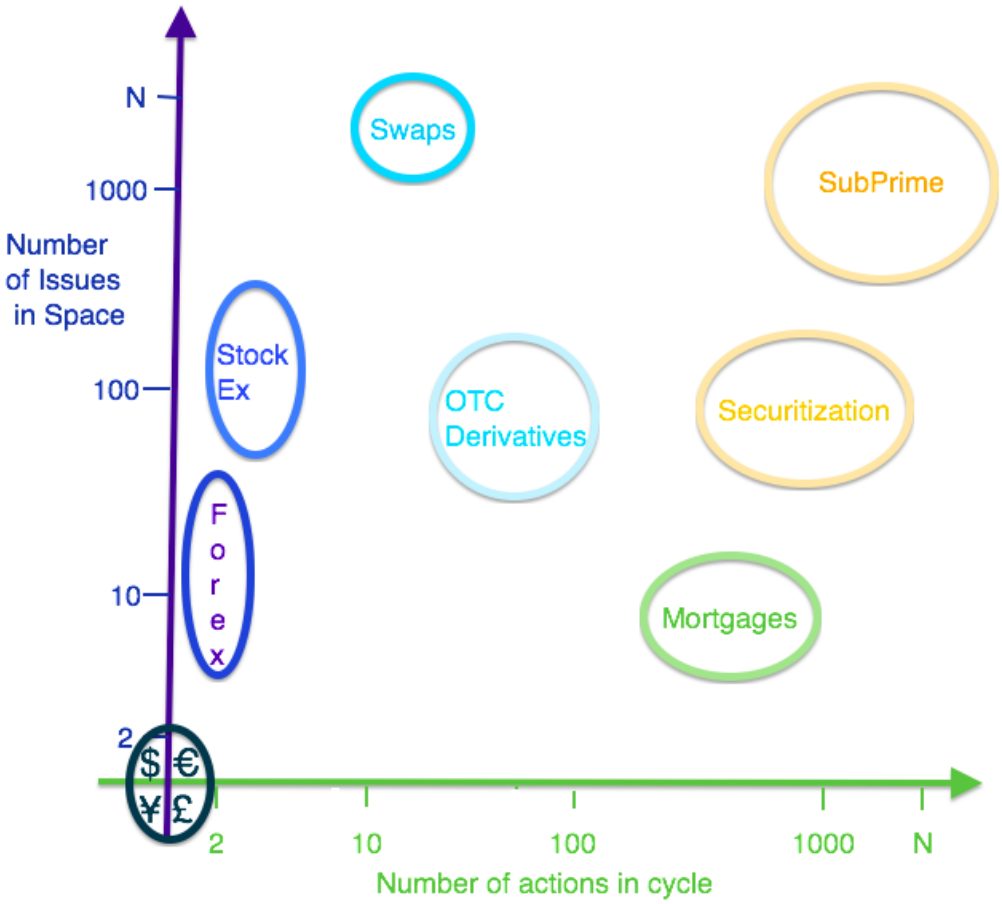

Figure 4. Semantically distinct instruments *versus* operationally complex performances.

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. ¡Ojala!

Meanwhile, the smart contract is really the machine to perform 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? Well, 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 Ricardo 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 [Ricardo]. Whereas Bitcoin can handle a multitude of smart-enabled transaction forms but in only one semantically trivial unit: assumed to be the bitcoin [Bitcoin]. Those axes in the figures cross at 1 !

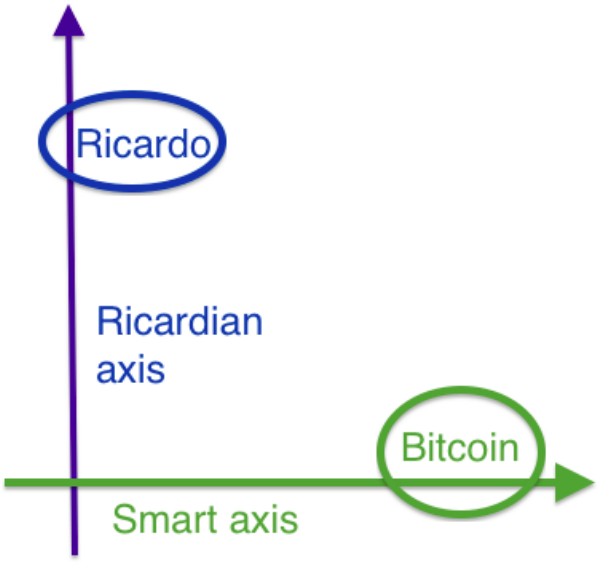

Figure 5. Ricardo *versus* Bitcoin.

Moving on

The task is to encompass both elements into one contract [Path]. Today, we've moved forward.

- OpenTransactions added server-side smart contracts to its technology permitting many more transaction types [OT].
- Askemos clients run agreed smart contracts and insert events and state into a merkle tree as they happen [Askemos].
- A newer system, OpenBazaar composes Ricardian Contracts into trade cycles of invoice, acceptance, payment etc, thus also handing many conceptually complicated transaction types [OB]. In concert, CommonAccord places small smart contracts with Ricardians and then composes these pairs into larger agreements.
- Ethereum is taking the approach called *Natural Specification Format* in its smart contract programming language Solidity [Ethereum]. 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.

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

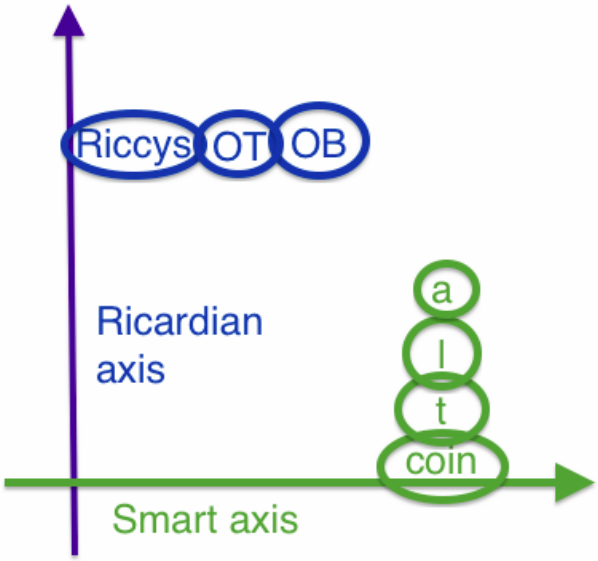

Figure 6. Evolution.

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.

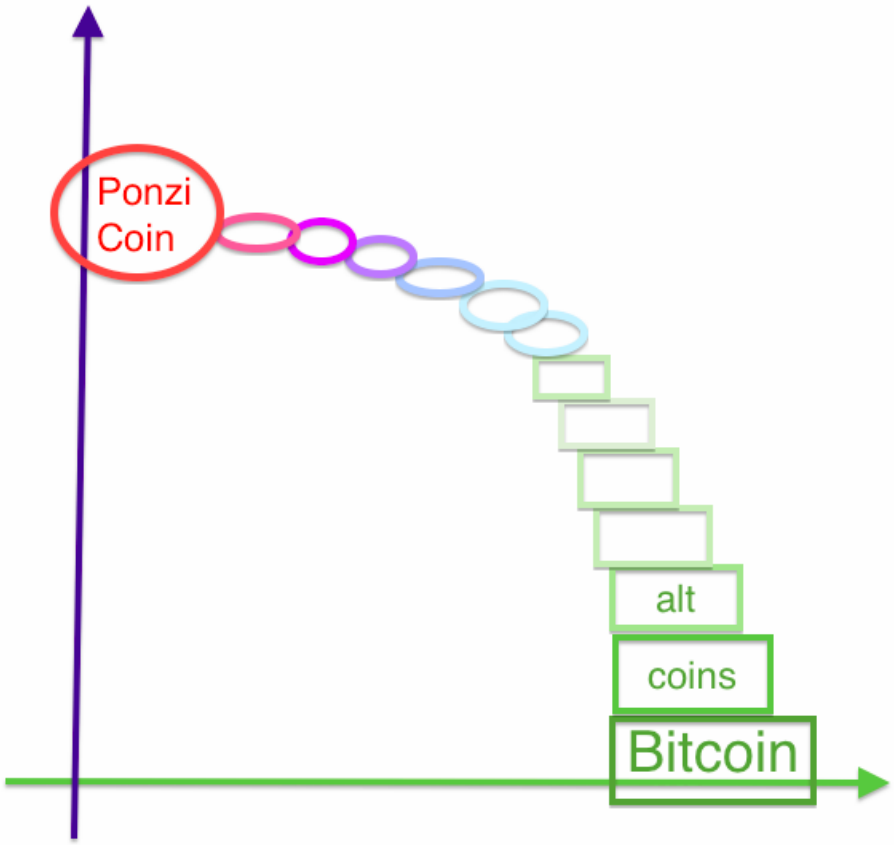

Figure 7. altCoins evolve to anti-semantics.

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.

We can now see that the real challenge between smart contracts and Ricardian Contracts or legal documents is not to choose, but to incorporate. The Bitcoin world will benefit from adding the semantic richness of legal documentation into its service. There are approximately three fronts along which this is taking place:

Firstly, from within, as direct issuances. Sidechains [Back] 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 — an innovation discussed in Mark Friedenbach and Jorge Timón's paper on multi-instrument chains [FreiMarkets]; (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 w expressing a movement of value from one key to another. See also The Sum of All Chains - Let's Converge! for a relevant framework [SOAC].

Secondly from without: the Ethereum, Ripple, or more contractually minded reworkings such as Hyperledger or Eris can more easily add in the integration of the two forms.

Thirdly, projects such as Askemos, CommonAccord and OpenBazaar are using the hybrid text & code form of contracts to compose new constructs such as reusable contracts and trade patterns [CA].

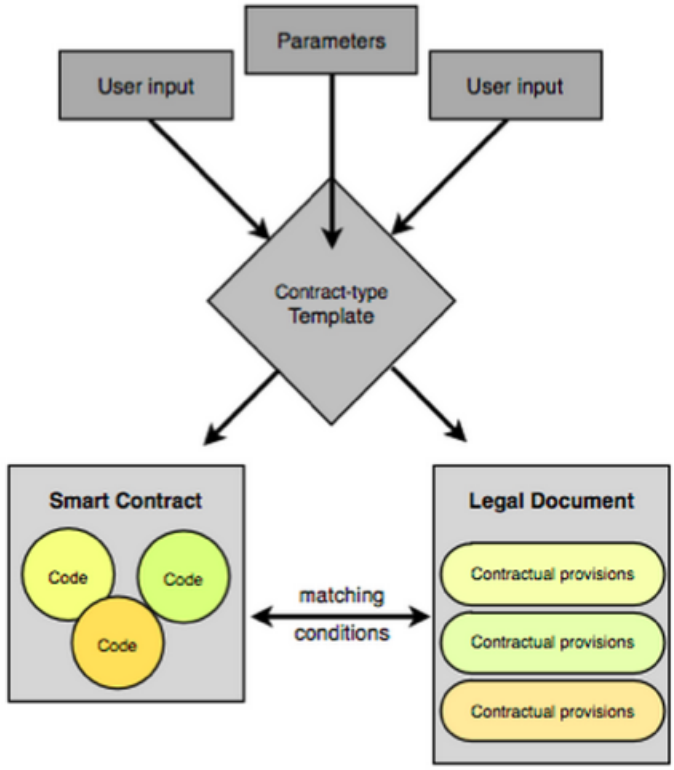

Figure 8. Coupling code snippets to clauses and composing upwards.

Likewise, my and other simple 3-party 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 note, and methods remain hotly debated.

This article received useful comments from Preston Byrne, Florian Glatz, James Hazard, Stephen Palley, Roger Willis and Jörg F. Wittenberger (alphabetical order).

References

[wip] A working draft of this article was located in google docs. This version July 2016 has tiny editorial improvements: citations, a permanent home.

[Szabo] Nick Szabo, "Smart Contracts," http://szabo.best.vwh.net/smart.contracts.html 1994

[Grigg] Ian Grigg, "The Ricardian Contract," http://iang.org/papers/ricardian_contract.html 2004

[WebFunds] WebFunds, "Implementations of Ricardian Contracts," http://webfunds.org/guide/ricardian_implementations.html

[AMIX] John Walker, "Understanding AMIX," The Autodesk File 4th edition, ed. John Walker, 1994

[Evidence] Ian Grigg "A small amount of Evidence. (In which, the end of banking and the rise of markets is suggested.)" http://financialcryptography.com/mt/archives/001299.html 2010

[Ricardo] Ian Grigg, "Financial Cryptography in 7 Layers," http://iang.org/papers/fc7.html in Financial Cryptography 2000, Yair Frankel, Ed. 2002 Springer http://www.springer.com/us/book/9783540427001

[Bitcoin] Satoshi Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," https://bitcoin.org/bitcoin.pdf 2009

[Path] Mark S. Miller and Marc Stiegler, "The Digital Path," http://www.erights.org/talks/pisa/paper/#human in Markets, Information and Communication: Austrian Perspectives on the Internet Economy, Jack Birner and Pierre Garrouste, Eds, 2003 Routledge https://books.google.com/books?id=x-FQrKrjgcYC&hl=en

[OT] Chris Odom, "Open-Transactions: Secure Contracts between Untrusted Parties," 2015, http://www.opentransactions.org/open-transactions.pdf and Chris Odom, "Sample Currency Contract," http://opentransactions.org/wiki/index.php/Sample_Currency_Contract 2013

[Askemos] Jörg F. Wittenberger, "Askemos - a distributed settlement," 2002, http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.11.5050

[OB] Washington Sanchez "Ricardian Contracts in Open Bazaar," 2014, https://gist.github.com/drwasho/a5380544c170bdbbbad8

[Ethereum] Lefteris Karapetsas and Gavin Woods, "Ethereum Natural Specification Format," 2014, https://github.com/ethereum/wiki/wiki/Ethereum-Natural-Specification-Format

[Back] Adam Back et al, "Enabling Innovation with Pegged Sidechains," 2014 https://blockstream.com/sidechains.pdf

[FreiMarkets] Mark Friedenbach, Jorge Timóna, "Freimarkets: extending bitcoin protocol with user-specified bearer instruments, peer-to-peer exchange, off-chain accounting, auctions, derivatives and transitive transactions," 2013 http://freico.in/docs/freimarkets-v0.0.1.pdf

[SOAC] Ian Grigg "Sum of all Chains - Let's Converge," CoinScrum / Proof of Work's Tools for the Future, 2015 http://financialcryptography.com/mt/archives/001556.html

[CA] James Hazard, CommonAccord, http://www.commonaccord.org/