FEDS Notes

Twitter Share RSS

June 23, 2021

What is programmable money?

Alexander Lee[1]

Introduction

Discussions of financial technology have recently started to include the idea of "programmable money," though the specific meaning of the term is often unclear. Different perspectives may presume a particular underlying technology or set of features to be a part of a programmable money system, and lack of agreement on these aspects may lead to confusion. To support clearer discussion of this concept in the central banking community and the financial industry more broadly, this research note offers an investigation into the nature of programmable money. This note focuses on the importance of a mechanism guaranteeing the inseparable functionality of the technical components of a programmable money system rather than prescribing the specific nature of those components. This "coherence guarantee" is crucial regardless of specific technical choices and admits a wide range of potential designs for programmable money. The guarantee also facilitates the view of programmable money as a concrete product, which may provide users with greater certainty about its nature and capabilities than alternative service-oriented models that can automate interaction with particular digital value records.

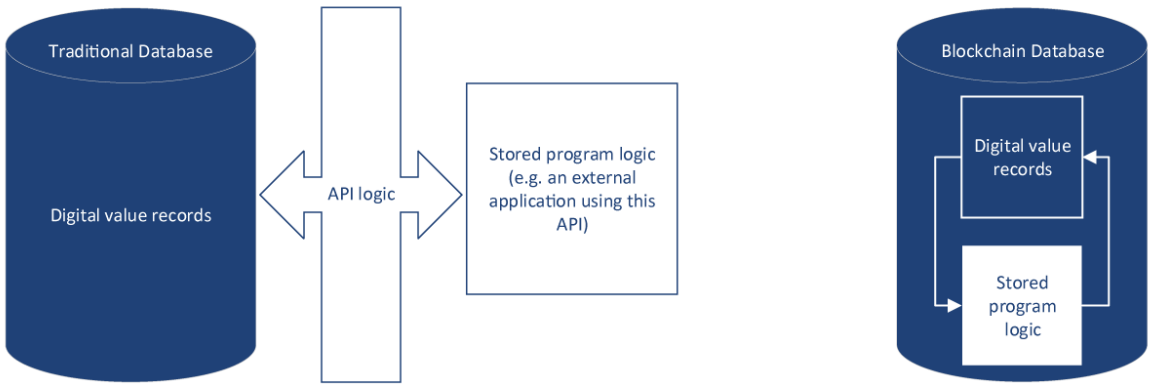
What is programmable money?

Many observers of financial technology have offered interpretations and discussion of potential use cases of programmable money.[2] While such references to programmable money typically describe it as being enabled by distributed ledger technology (DLT) or blockchain systems, this is not universally the case, and the term remains ill-defined.[3] Two natural components of the definition are a digital form of money and a mechanism for specifying the automated behavior of that money through a computer program (this mechanism is termed "programmability" in this note). However, it is not clear whether these components alone are sufficient for a definition, given that various combinations of similar technology for payments automation have existed for decades. It was only after the advent of public blockchain cryptocurrencies that the term "programmable money" became common parlance.[4] So what is it about these new systems that has prompted the recent spate of references to the term, and does the answer somehow imply that DLT has to be a part of any "programmable money" system?

One facet of successful public blockchain systems that may provide some clarity is how they closely link digital value and programmability in a single system that only functions properly

when both are present. In traditional financial technology systems, digital money is typically defined by database entries. Any "programmability" offered for this money, whether internally to the entity maintaining the database or exposed to its customers via an application programming interface (API), involves another technology system built separately from that database and then connected in some fashion. While newer cryptocurrency systems also use a database (often in the form of a blockchain data structure), a key difference is that the records in such blockchains either directly incorporate some programmable script (as Bitcoin records do, for example), or sit alongside a general programming functionality within the system that allows for direct manipulation of those records (the model used by Ethereum, among others).[5] In both designs, the value represented in those systems and the programmability of that value are tightly integrated. There is no notion, for example, of "bitcoins" without an associated script governing their spending conditions, whereas a traditional ledger could certainly hold digital records of money without offering a programming interface to those records.[6]

Figure 1. Simplified visualizations of a traditional database using an API for programmability, and a programmable blockchain



A traditional database connected to an API that exposes basic programmable logic (left) enables stored program logic in the form of applications to interact with database records. A blockchain database (right) can store both value records and program logic in the same database, with some intrinsic mechanism enabling interaction between the two.

The necessary technical components and the notion of tight integration between those components suggests a definition of programmable money as a unified, coherent product that encapsulates both the storage of digital value and programmability of that value. This

note uses the term *coherence guarantee* to refer to a mechanism guaranteeing that the technical components of the programmable money product are "inseparable" and that those components are consistently functional, such that the product is stable and coherent for users. While nominally "separable" technologies (including traditional technologies that predate DLT) may be used to implement the two core technical components depicted above, the coherence guarantee must ensure that they are not separable as far as the overall product is concerned. Without this guarantee, the individual components are nothing novel: Digital money as a product has existed in many forms for years, as has the ability to write computer software.[7] With the guarantee, programmable money may be seen as a new product category sitting alongside existing money products such as central bank deposits, demand deposits, nonbank money, and cash. The benefit of stability that this guaranteed inextricability provides users of the product may not be present in other systems relying on services that can be altered by their providers and which may be subject to outages. The idea of the guarantee itself is broadly agnostic to technical choices, allowing for a range of approaches to the design of the digital money and programmability components using a variety of technologies.

Technical designs for programmable money: possibilities and tradeoffs

There are many ways to approach the technical design of a programmable money system. Such designs may incorporate both traditional technologies and newer alternatives like DLT, with specific technical choices being driven by the goals of the system designers. While a technical design can address the necessary components of a programmable money system (digital money and programmability), a coherence guarantee cannot be provided by technology absent some broader reinforcing incentives (which might be economic, legal, or reputational). It is nevertheless important to understand the possible technical approaches for programmable money system designs, some of which are described below along with attendant tradeoffs.

Recordkeeping for digital money

The most basic building block of a programmable money system is a set of digital records representing transactable value. These records will almost certainly be kept in a persistent database of some variety. Many existing financial technology systems use traditional relational databases for this purpose, such as a bank keeping a digital record of customer accounts, while DLT systems typically use a blockchain-style database to store records of value. A notable difference between these technologies is that traditional databases are often general purpose: They may be used for financial applications as a way to store accounts, but they are designed to be flexible enough to be used to store virtually any type of digital record, and they do not have any "awareness" of the implications of the records they store.

DLT database designs, by contrast, are usually purpose-built for financial accounting of their integrated digital currencies and have intrinsic mechanisms for detecting invalid transactions such as doubly spent value or the spending of greater value than is available to a user. In addition to helping maintain correct value records, such mechanisms can complement

programmability in those systems, performing fundamental accounting checks regardless of what type of programmatic instruction is issued. In the public blockchain context, these mechanisms help ensure the proper operation of the system and provide additional surety to users when combined with a mechanism such as proof-of-work, which is designed to make the (correctly recorded) records practically immutable. Deployed in a private context, DLT could potentially offer some of the same specialized recordkeeping benefits to the operator of the system (with the notable exception of immutability), if not necessarily to the system's users.[8]

The format in which records of value are kept in a database may have implications for the design of the programmability component of a programmable money system. One recordkeeping possibility is the traditional account format, such as a bank or financial service provider might keep for a customer. The account format has the immediate advantage of being a familiar construct and is the de facto standard for traditional financial technology systems. Some DLT systems use the account format as well, enabling a single account to control a variety of digital assets and to issue programmatic instructions.[9] A potential drawback of the format is that it makes tracing the history of any particular unit of value more difficult, if this is a desired feature of the system.

The unspent transaction output (UTXO) format, first popularized by the Bitcoin system, more readily facilitates this type of historical transaction record. This alternative recordkeeping model records all discrete amounts of value (the "unspent outputs") in existence at the current time, and associates ownership of that value directly with the amounts.[10] A rough analogy from the traditional financial world might be banknotes in circulation: Owners are identified by current bearers, and individual ownership of any banknote may be reassigned by its bearer at any time. In systems like Bitcoin, ownership is asserted through satisfaction of a conditional script associated with each UTXO, typically requiring a valid digital signature from a specific cryptographic private key. Possession of the correct private key is thus tantamount to "possession" of the value, and for this reason bitcoin has sometimes been referred to as a "bearer instrument," though the analogy is somewhat imprecise.[11] Benefits of the UTXO model include conciseness of the records being kept (there is no such thing as an "inactive account" or "empty account" under the UTXO recordkeeping model) and the ability to associate specific (programmable) spending conditions with every discrete UTXO. Drawbacks of UTXO-based recordkeeping include the need to maintain external metadata in order to roll up "account balances" for individual users (if such balances are desired) and the potential abuse of the format to fragment value across many UTXOs, which will need to be tracked by the system.[12]

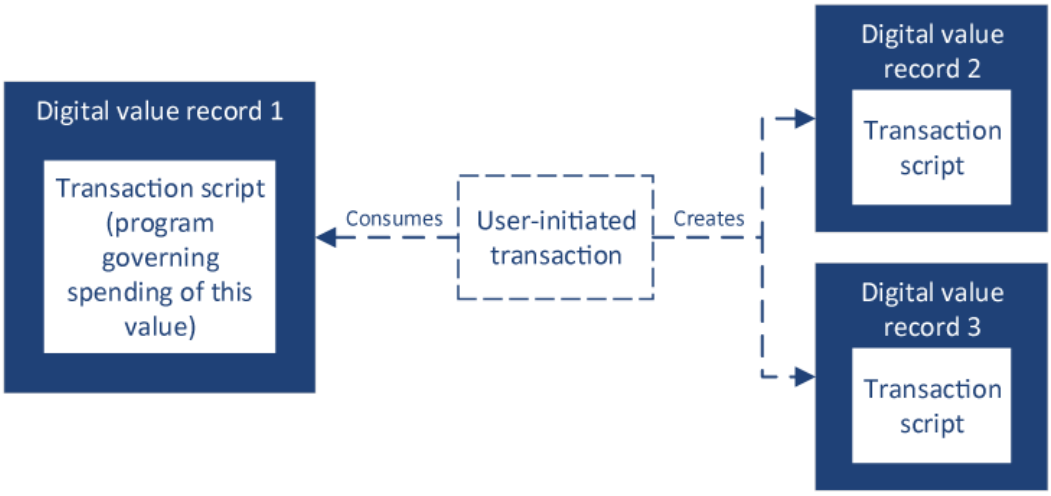
Enabling programmability

In conjunction with technical decisions about how to store transaction and balance records for programmable money, a system designer must also consider how to enable the programming capability for that money. There are several potential methodologies to choose from, some of which are typically used in conjunction with DLT and others that are more often

used for traditional financial technology systems. In some cases, these decisions involve mutually exclusive designs for enabling programmability, but there is usually broad flexibility in how programmability can be enabled.

Public blockchain systems have generally followed one of two strategies for enabling programmability, denoted here as the transaction scripting approach and the virtual machine approach. While these approaches are not mutually exclusive and do not dictate a particular choice of recordkeeping format, certain combinations are logically and technologically complementary. Under the transaction scripting approach, a (small) program is attached to every discrete amount of value tracked by the system, indicating how that amount may be spent. This approach is analogous to having a programmatic spending condition attached to every banknote or physical coin in existence, which can be re-programmed when spent. This "digital cash" model is the approach pioneered by Bitcoin, combining the UTXO recordkeeping format with the transaction scripting model, with one transaction script per UTXO dictating how it may be spent. When a UTXO is consumed as an input to a financial transaction, a new script can be associated with every output of that transaction. Users can control any number of UTXOs and can spend them independently by satisfying the associated transaction scripts.

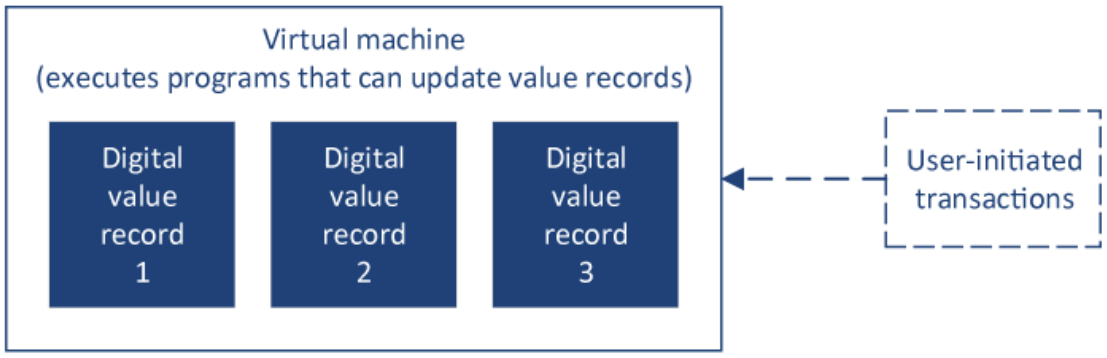
Figure 2: A simplified schematic of the UTXO recordkeeping format combined with the transaction scripting approach to programmability



User transactions satisfying an individual transaction script may spend the corresponding UTXO. Spending a UTXO wholly consumes it, and the user transaction may create one or more new UTXOs with new transaction scripts as a result.

By contrast, the virtual machine approach embeds programming capability in the system itself in the form of virtual machine instructions that can recognize and manipulate the value stored in the system. This is the approach taken by Ethereum, which allows transactions to contain programmatic instructions and further allows programs written for its own virtual machine (the EVM) to be deployed to and stored in the blockchain. Programs deployed in this manner may persist indefinitely, and their functionality can be repeatedly used by future transactions. These programs are typically called "smart contracts," a term that predates public blockchains and need not imply any contractual obligation in a legal sense when used in the contemporary DLT context.[13] This approach is combined with the account recordkeeping format for recording value in Ethereum and several other DLT systems to allow for programmatic manipulation of account balances.

Figure 3: A simplified schematic of the account recordkeeping format combined with the virtual machine approach to programmability

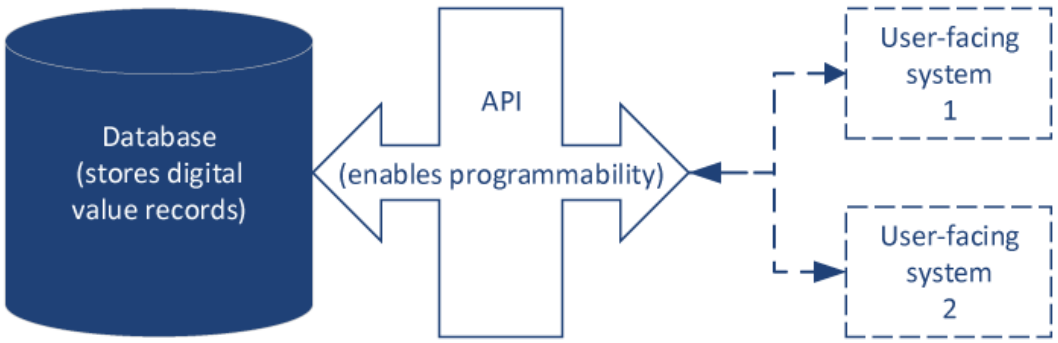


User transactions can invoke program logic that the virtual machine executes, manipulating account balances where authorized.

These two designs present different tradeoffs. Potential benefits of the "digital cash" model using programmable UTXOs are the ability to specify spending constraints on any discrete amount of value and a greater facility to trace the provenance of any particular "virtual banknote." Benefits of the account-based model may include greater fungibility of value and efficiencies from reusing smart contract code for frequently desired program functionality. New technological designs may borrow elements of both models and offer a different range of benefits, and the interpretation of what is "beneficial" may depend on the objectives for the system outside of just the technology.

Financial software systems that do not use DLT, for example programmable payment services like PayPal, have typically enabled programmability through an API layer on top of some combination of underlying technology. Because the API is an abstraction of the system that it provides an interface for, DLT could in fact be used behind an API as well.[14] APIs are an example of a component that does not preclude another technology choice, and these interfaces can be layered on top of one another (and often are in practice). Beyond this flexibility, benefits of APIs include established design patterns and general familiarity among system developers, which may not be the case with newer technologies associated with DLT, such as UTXO-based recordkeeping or smart contracts. Drawbacks of APIs include the potential decoupling of the programmability from the digital representation of money if a discrete API gateway were the sole interface for programmatically interacting with that money (for example, a bank's API for programmability might go down while its databases for querying account balances on its customer portal remain up), and an increasing potential for system failure if numerous critical APIs are layered atop one another.[15]

Figure 4: A simplified schematic of a traditional software system combining a database with an external API gateway



All user transactions flow through the API gateway, which defines the (current) set of permissible programmatic operations. Operation of the database is logically independent from the API layer.

Another key consideration in the technical design of a programmable money system is how to prevent abuse of the programmability. Even if the mechanism correctly functions to only allow valid commands to be programmed, valid commands may be combined (maliciously or inadvertently) in ways that lead to undesirable consequences. One major form of potential abuse is a denial-of-service attack, in which large numbers of instructions to execute some

programmatic function are issued repeatedly by the attacker, forcing the system to perform wasteful computation at the expense of providing service for legitimate requests. Public blockchain systems have implemented multiple deterrence mechanisms out of necessity, lacking any central operator capable of screening out an attacker's traffic: Fees are typically required for every transaction, introducing an economic disincentive for abusive transactions, and the extent of potential program execution is also limited in some fashion.[16] Private DLT implementations may also deliberately constrain program execution, and APIs for existing financial service are often rate-limited to prevent similar abuse; such techniques could be used in conjunction depending on the overall system design. Different designs may present perceived drawbacks to users of the programming facility (such as offering only a limited set of commands), and at a certain level of restriction, whether the system still constitutes "programmable money" may be called into question.[17] Regardless of exact design, however, some level of restriction is likely a necessity to allow the system to operate correctly for legitimate users.

Providing coherence guarantees for programmable money

While the technical design of a programmable money system is important, the technology alone cannot intrinsically provide a sufficient coherence guarantee. The system must also rely on a convincing set of external incentives to uphold the guarantee, lest the overall product not be seen as credible in the eyes of its users.[18] These incentives can be viewed as falling within at least three broad categories of economic incentives, legal incentives, and reputational incentives, though other frameworks are possible. These categories are not mutually exclusive, though certain would-be providers of programmable money may rely more heavily on some of them than on others. In addition, the technical design of a system may lend itself well to reliance on certain types of incentives.

Reinforcing guarantees through incentives

Economic incentives would feasibly apply to almost any offeror of a programmable money product. Contemporary providers of money have a clear economic incentive for simply continuing to offer certain products, such as seignorage from banknote issuance. If a programmable money provider offered a product that had a similar economic benefit to the issuer, this incentive would help guarantee that product's continued provision. While public cryptocurrencies may not have any specific offering entity, they may nevertheless rely on economic incentives to perpetuate the system, as exhibited for example by the subsidies and fees involved in the process of proof-of-work mining, which will be explored further below.

Legal incentives are also likely to apply in some form to nearly all offerors of programmable money, though the structure of these incentives may be highly variable across jurisdictions. From a user's perspective, the nature of what exactly is being legally guaranteed may also be salient. How to legally enforce that a programmable money system be provided on an ongoing basis is unclear, for example, though it is easier to envision a legal requirement that restitution be provided to users of such a system if it were taken offline (at least in cases where the provider is an identifiable entity subject to such requirements in its own jurisdiction,

such as a corporation). While restitution in such a situation is certainly preferable to those users receiving nothing at all, it may be the case that some users would prefer to have the programmable money that they believed they had access to, rather than some alternative form of compensation.[19]

Reputational incentives are perhaps most variable in importance depending on the offeror of programmable money, and in some cases may factor in only marginally. For example, the creator(s) of Bitcoin remain anonymous, and have different incentives to maintain a reputation than an established company or government looking to launch a programmable digital money might. For certain entities with exceptionally strong reputations, such as governments capable of issuing investment grade debt, or central banks capable of issuing widely accepted fiat currency, an incentive to uphold that strong reputation may in fact be the most important part of a guarantee.

Coherence guarantees in action: practical and theoretical examples

A practical example of a coherence guarantee is that offered by public blockchain systems for their native cryptocurrencies. These systems rely on economic incentives for network participants to guarantee their continued operation and the integrity of the associated transaction ledger. For systems that use proof-of-work consensus, such as Bitcoin and Ethereum, network participants performing transaction validation and construction of the shared ledger receive economic rewards denominated in native cryptocurrency for performing this work (in expectation), and must consume economic resources (in the form of electricity) in order to do so.[20] Because validators maintaining the canonical ledger history will reject any invalid transactions, including double-spend transactions, while any attacks on the network and ledger history which involve mining still incur the cost of performing that work, there is a significant economic cost for dishonest miners. The economic cost of proof-of-work is also where the "immutability" of such public ledgers derives from: In order to change the ledger history, an attacker would have to consume enough resources to produce proofs of work for an alternate section of ledger history that would become the longest version of the blockchain.[21]

It is important to note that while these incentives in combination with the technical design of the underlying systems have worked to uphold the coherence guarantee of native cryptocurrencies like bitcoin and ether for several years, the same incentives do not apply to non-native "cryptocurrencies" like ERC-20 tokens built on top of the Ethereum platform.[22] While such tokens may rely on the same consensus mechanism as ether when it comes to recording transactions, they do not derive value from that mechanism in the same manner, because miners on the Ethereum network are paid for performing proof-of-work in ether, not in ERC-20 tokens. Thus, while such tokens may be programmable, they may not represent "money" in the system and may subsequently not fall under this note's definition of programmable money.

Another example of a practical system that could feasibly qualify as programmable money is a privately administered system such as a payment service provider or bank offering

programmability for digital money through an API. Such systems can offer many of the same technical capabilities as other programmable money systems for their users, but the incentives underpinning the maintenance of those systems may be more complex and the coherence guarantee less robust, possibly to the point where the label may be inapplicable. For example, a large financial services company may have legal incentives to serve its customers well enough to avoid lawsuits, reputational incentives as a provider of a well-known programmable API for payments, and economic incentives in the form of taking fees for its services, and some customers may find these incentives credibly strong enough to qualify the company's services as programmable money. However, such services can still be discontinued at any time in whole or in part if incentives were to change.[23] The relative strength or weakness of a coherence guarantee for programmable money being offered by a private entity may thus be seen to rely on the particular mix of incentives that the offeror has for maintaining such a product, and these incentives may shift considerably over time.

A more theoretical example of a programmable money system is a hypothetical programmable digital currency offered by a public-sector entity already issuing other forms of money. Central banks of sufficient reputation and credible technical capability to offer a programmable money system may be able to offer a coherence guarantee primarily on the basis of their reputations, with an implied incentive for maintaining those reputations.[24] This approach is not open to most would-be providers of programmable money systems, but it may be the case that the same authority that allows certain governments to issue widely used fiat money (with its implicit guarantee of continued value in the future enabling its use today) will also allow those entities to issue programmable money "by fiat." Although no central bank has yet offered such a programmable digital currency product, it remains a possible development if general interest in central bank-issued digital currencies persists.

A final note on guarantees

The word "guarantee" in the preceding discussion should always be read with implied quotation marks around it. Public cryptocurrency systems, like Bitcoin, have worked in practice for years, but the behavior of validators in the long run when the incentive structure no longer includes subsidies in the form of block rewards remains an open question, and external factors leading to a loss of confidence in such systems could also affect the guarantee.[25] As illustrated in the practical example of privately offered programmable money, such systems may vanish if the offeror decides to discontinue them, technological capability regardless. Central banks could similarly issue and later elect to demonetize a digital currency. Attempting to assess the comparative strength of various coherence guarantees in these situations and others is beyond the scope of this note. Potentially more important than any such hypothetical framework, however, is the credibility of the guarantee for any given programmable money product in the eyes of its potential users. Observing user attitudes towards preexisting services like payment APIs and newer offerings such as open banking APIs and cryptocurrencies may offer insight into this evolving landscape in the coming years.

Conclusion

While "programmable money" as a term may have originated in the public blockchain community, as a concept it need not involve distributed ledger technology at all. There are multiple possible approaches to the design of a technological system offering a digital representation of money with an associated programming facility; what is more important than the specific technical approach is the guarantee that the system will in fact cohere into a unitary product offering, rather than a service offering associated with an otherwise independent non-programmable digital money. This coherence guarantee offers greater stability and consistency of experience to users over "programmability-as-a-service" models, and must be reinforced by suitable incentives, with different providers of programmable money relying on different forms of incentives. The ultimate assessment of the suitability of such incentives and thus the credibility of the coherence guarantee will come from users of these products, and their evaluations of different forms of programmable money may evolve over time as more practical examples are offered to the public.

References

Bechtel, Alexander, Jonas Gross, Philipp Sandner, and Victor von Wachter (2020, September 29). "Programmable Money and Programmable Payments," Medium. https://jonasgross.medium.com/programmable-money-and-programmable-payments-c0f06bbcd569

Carlsten, Miles, Harry Kalodner, S. Matthew Weinberg, and Arvind Narayan (2016). "On the Instability of Bitcoin Without the Block Reward (PDF) ," Princeton Economics Working Papers.

Harney, Alexandra, and Steve Stecklow (2017, November 16). "Special Report: Twice burned – How Mt. Gox's bitcoin customers could lose again," Reuters. https://www.reuters.com/article/us-bitcoin-gox-specialreport-idUSKBN1DG1UC

Koning, John Paul (2020, November 19). "Programmable money isn't new, we've had it for ages," Moneyness. http://jpkoning.blogspot.com/2020/11/programmable-money-isnt-new-weve-had-it.html

Lee, Alexander, Brendan Malone, and Paul Wong (2020). "Tokens and accounts in the context of digital currencies," FEDS Notes. Washington: Board of Governors of the Federal Reserve System, December 23, 2020.

Lewis, Antony (2020, April 26). "What Actually is Programmable Money? " Bits on Blocks

Nakamoto, Satoshi (2008). "Bitcoin: A Peer-to-Peer Electronic Cash System (PDF) ."

Szabo, Nick (1994). "Smart Contracts ."

---

1. The views expressed in this paper are solely those of the author and should not be interpreted as reflecting the views of the Board of Governors or the staff of the Federal Reserve System. The author would like to thank Shari Bower (Federal Reserve Bank of Atlanta), Angela Lawson (Federal Reserve Bank of Minneapolis), and Gordon Liao, Peter Lone, David Mills, Paul Wong, and Sarah Wright (Federal Reserve Board) for their contributions and

assistance towards this note. Return to text

2. For example, see Lewis (2020), Koning (2020), and Bechtel et al (2020) for other approaches to this topic. Return to text

3. The terms "DLT system" and "blockchain system" are used interchangeably in this note, though this is imprecise, as DLT need not use a blockchain structure specifically for storing records. Where the distinction is meaningful, it will be noted. Return to text

4. At least per Google Trends, there is no recorded usage of the search term prior to November 2013, which postdates both the launch of Bitcoin in 2009 and the release of the Ethereum whitepaper in 2013: https://trends.google.com/trends/explore?date=all&q=%22programmable%20money%22. Return to text

5. Though other data structures such as directed acyclic graphs are possible for a cryptocurrency database, this note will use the term "blockchain" to refer to the database portion of such a system, as this is the most common structure observed in practice. Return to text

6. Ethereum uses a different recordkeeping model which does not require a programmable script to be associated with every discrete amount of its own cryptocurrency (ether), but the integrated system storing ether balances for accounts runs on top of a virtual machine (the Ethereum Virtual Machine, or EVM) which incorporates programming capabilities that can manipulate those ether balances directly. The EVM is inseparable from the overall system storing the balances, which is roughly analogous to the inseparability of transaction scripts from bitcoin amounts in the Bitcoin system. Return to text

7. Some computer software predating the advent of public blockchain systems can of course interact with a representation of digital money, such as payment automation services or financial APIs. Whether this type of software can be considered part of a programmable money product depends on the availability of a coherence guarantee. If there is no such guarantee, this software might better be thought of as "programmability as a service" for a certain form of digital money, rather than a component of a product. Return to text

8. A private operator in control of the recordkeeping system could always revise it to state whatever they wished, regardless of what technology were used to implement the system, potentially to the detriment of users of that system. It is worth noting that an operator of a private system might not even perceive "immutability" to be a beneficial feature if they had legitimate reasons for needing to revise records. In general, while the terms "blockchain" and "DLT" are often associated with the notion of immutability as though it were an intrinsic feature of these systems, this is a misconception. The "immutability" of public blockchain systems is predicated on their public nature (with many eyes looking at the same history and checking for consistency), and an additional mechanism such as proof-of-work that is designed to make that history difficult to rewrite. In the case of proof-of-work based systems, a better description might be that the records stored in such systems are "probabilistically immutable" up to a certain point in recorded history, and tamper-evident if the records are stored in public. Private implementations of DLT which are under the control of a single entity cannot provide the same form of assurance and must use other mechanisms to guarantee their integrity. This latter topic is the focus of the third section of this note. Return to text

9. In practice, most public blockchain systems using the account format do not restrict the number of accounts that a user may create, nor do they associate a real-world identity with those accounts as a traditional financial institution would, though alternative implementations of the technology could enforce such restrictions. Return to text

10. The term "unspent output" refers to amounts that have not (yet) been used as an input to a financial transaction. When spent, a UTXO is wholly consumed as an input to the transaction (losing its unspent status in the process), with the transaction resulting in the creation of one or more new UTXOs. Return to text

11. The actual values (that is, the bitcoin amounts themselves) cannot be "possessed" per se as they are only ever recorded in the Bitcoin blockchain. These values cannot be "removed" from the blockchain to be privately held in the manner that banknotes might be withdrawn from a bank account and privately held. Return to text

12. Public blockchains like Bitcoin attempt to discourage such behavior through transaction fees per UTXO: "dust" UTXOs encode values so small that they cannot afford to pay for their own spending at a certain fee level, so there is an economic disincentive to the creation of dust outputs. Return to text

13. For an early formulation of the concept of smart contracts, see, for example: https://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smart.contracts.html Return to text

14. While APIs are occasionally contrasted with smart contracts as an approach to facilitating programmable money, the interfaces that DLT-based smart contracts expose to users of those systems could reasonably be classified as APIs themselves, providing an interface for the functionality of those specific programs encoded in the smart contract. These generic technologies (smart contracts and APIs) should not be considered mutually exclusive alternatives. Return to text

15. The issue of decoupling could potentially be ameliorated through redundancy of interfaces between the relevant API gateway(s) and the digital representation of money at every intervening layer, but in general may be a more problematic technology choice for programmable money, and the coherence guarantee for a system designed in this manner might rely more heavily on the offeror's incentives to guarantee the intended operation of the technology. Return to text

16. Bitcoin takes the approach of only allowing limited instructions to be processed, in a purpose-built programming language called Bitcoin Script that disallows loops (and thus the potential for programs that run forever); Ethereum allows for "unrestricted" computation in the sense that the EVM's instruction set is Turing-complete, but the total amount of computation that any transaction may incur is capped (thus, a transaction could execute a loop, but only for a finite number of cycles). Both methods work to ensure that the respective blockchain systems cannot be brought to a halt by infinite loops via their programmability mechanisms. Return to text

17. Within the cryptocurrency community, there is a general perception that the computations allowed should be as permissive as possible, while preventing abuse of the system through infinite loop-based denial of service attacks. Return to text

18. For example, consider a different money product, cash: If the offeror of a banknote had no reasonable incentive to continue supporting its use, those banknotes could become worthless as soon as users realized that the (implicit) guarantee of their continued usability had vanished. Return to text

19. See, for example, the ongoing case of Mt. Gox, with creditors valuing claims at the rate of the underlying digital currency lost, indicating a mix of consumer preferences and the general difficulty of working through a legal enforcement mechanism in practice: https://www.reuters.com/article/us-bitcoin-gox-specialreport-idUSKBN1DG1UC. Return to text

20. The terms "miner" and "mining" are often used to refer to these specific network participants and the process of performing proof-of-work to add new transactions to the ledger, respectively. Rewards in proof-of-work systems are probabilistically received in proportion to the share of computational power that a miner contributes towards performing proof-of-work. Return to text

21. The original Bitcoin whitepaper includes some calculations related to the feasibility of attackers being able to rewrite the recent history of the ledger in this manner; a typical rule of thumb used in the Bitcoin community is that transactions buried at least 6 blocks deep in the ledger history can be considered "immutable." See Nakamoto (2008), Section 11 for calculations. Return to text

22. For a more detailed discussion of ERC-20 tokens, see Lee, Malone, and Wong (2020). Return to text

23. This was the case with Fidor Bank UK, a former digital bank offering an open finance API that shut down after less than four years because of market uncertainty. While Fidor UK's customers would have been legally entitled to the currency-denominated value of their accounts, the programmability service was discontinued for those customers, and with it dissolved the coherence of any perceived "programmable money" product on offer. For an

archived snapshot of the (now-defunct) website's FAQs regarding Fidor Bank UK's shutdown and the issues facing users of their financial products as a result, see https://web.archive.org/web/20191222051436/https://www.fidorbank.uk/faq Return to text

24. This form of guarantee and the extent to which others find it credible may be evidenced by long-term public debt issued by governments, and the yield demanded on that debt by investors; such instruments could serve as a proxy for the feasibility of using reputational incentives as the primary basis of guaranteeing the coherence of a programmable money system. Return to text

25. For some research on the security of Bitcoin due to long-run miner incentives, see Carlsten et al (2016). Return to text

Please cite this note as:
Lee, Alexander (2021). "What is programmable money?," FEDS Notes. Washington: Board of Governors of the Federal Reserve System, June 23, https://doi.org/10.17016/2380-7172.2915.

Last Update: June 23, 2021