INTERNATIONAL MONETARY FUND

Programmability in Payment and Settlement

Concepts and Implications

Prepared by Xavier Lavayssière and Nicolas Zhang

WP/24/177

2024
AUG

WORKING PAPER

 WP/24/177

IMF Working Paper
Information Technology Department

Programmability in Payment and Settlement — Concepts and Implications

Prepared by Xavier Lavayssière and Nicolas Zhang[1]

Authorized for distribution by Hervé Tourpe
August 2024

ABSTRACT: Programmability in payment and settlement has yet to realize its potential to support policy goals such as efficiency, safety, and innovation. This paper proposes a comprehensive framework for understanding and evaluating programmability. It explores two key dimensions: external programmatic access, which is the ability for external participants to access the system data and functions with code, and internal programmatic capabilities, the extent to which internal execution of programs is supported and guaranteed. By developing strategies based on these dimensions, financial institutions, regulators, and related actors can better improve resilience, reduce costs and interoperability, all while managing associated risks. The resulting hybrid systems are coordinated efforts balancing the advantages of permissionless blockchains, such as composability, with regulatory requirements and a wider range of technologies. The paper describes these programmatic models to inform and guide the development of digital finance, bridging policy discussions with technical considerations.

RECOMMENDED CITATION: Lavayssière, Xavier; Nicolas Zhang (2024). Programmability in Payment and Settlement — Concepts and Implications. Working Papers WP/24/177, International Monetary Fund.

JEL Classification Numbers:	E14, L86, F33, G15, O33
Keywords:	Programmability; Tokenization; Automation; Payment; Settlement
Authors' email addresses:	XLavayssiere@imf.org, NZhang@imf.org

[1] The authors would like to thank Sonja Davidovic, Clement Berthou, Paul Desprairies, Andreas Veneris, Veljko Andrijasevic, Itai Abraham, German Villegas-Bauer, Yaiza Cabedo, Raunak Mittal, Ashley Lannquist, Arvinder Bharath, Victor Budau, Frankosiligi Solomon, Pearl Kuebel, and Hervé Tourpe for their careful reviews and comments.

WORKING PAPERS

Programmability in Payment and Settlement

Concept and Implementations

Prepared by Xavier Lavayssière and Nicolas Zhang

Contents

BOXES

FIGURES

Glossary

Application Programming Interface (API): A digital interface enabling programs to interact with a digital platform in a standardized, secure, and reliable manner.

Asset Smart Contract: A program running on a blockchain infrastructure representing an asset digitally. It contains data on ownership, and operation functions.

Atomicity: Indivisibility of a digital operation. Atomicity is employed to guarantee the mutual conditionality of both legs of settlement mechanisms such as DvP or PvP.

Conditional Payment: A payment category transmitted with instructions to settle once specific conditions are met. Such a condition could be a delay or the confirmation of the other leg of a trade.

Composability: The capacity to programmatically combine operations. For example, a tokenized debt can serve as collateral in other automated operations. Composability relies on shared interfaces, trust-minimization, and connected infrastructures.

Delivery vs. Payment (DvP): A settlement mechanism ensuring the mutual conditionality of the transfer of a financial instrument, and its corresponding payment.

Distributed System: A software solution implemented over multiple agents, processes, or computers. Properly designed distributed systems enhance scalability, availability, and resilience of digital platforms.

Distributed Ledger Technology (DLT): A state management distributed system inspired by Bitcoin's blockchain. DLTs are primarily used in finance to maintain a shared ledger across various entities.

Fast Payment System (FPS): A digital infrastructure enabling immediate or near-real-time transfer and settlement of funds between parties.

Ledger: Register of financial assets ownership. It can denote debt, money, or financial instruments.

Native Digital Asset: A financial asset directly issued on a digital platform. In the context of permissionless blockchains, it refers more specifically to the underlying digital asset used to pay for the network's security (e.g., Bitcoin or Ether on their respective platforms).

Open-loop vs. Closed-loop: An open-loop payment system is accessible to different payment companies (e.g., card networks), while a closed-loop payment system is generally limited to one company (e.g., transportation cards or voucher cards).

Oracles: Services providing external data to smart contracts on blockchains. Oracles provide data, such as stock prices or interest rates, in a decentralized or centralized manner.

Partitioning: Process of dividing a database into smaller logical partitions. Each partition can be stored on distinct servers while managed as a unified database.

Payment and Settlement System: arrangements, infrastructures and schemes that facilitate financial transactions between institutions. They include systems provided by banks, Fintech firms and central banks such as Fast Payment Systems and Financial Market Infrastructures.

Payment vs. Payment (PvP): A settlement mechanism ensuring the mutual conditioning of two parties' payments. It reduces settlement risk in foreign exchange transactions.

Permissionless blockchain: Shared ledger maintained by a distributed network where anyone can participate to the validation according to consensus rules.

Primitive: Basic functions of a platform accessible to internal, and external programs. Programs use them to fetch data and trigger actions.

Programmable Financial Platform: Digital platform enabling code-based financial operations through interfaces like APIs, smart contracts deployments, or other code-based tools.

Programmed Asset: A financial asset with code-defined properties that maintain its integrity and contain its usage. On a blockchain, it uses an "Asset Smart Contract."

Secure Element: A hardware component that guarantees secure code execution and sensitive data storage. A secure element can consist of a dedicated chip or a secure enclave within a microprocessor. Secure elements are commonly used in payment cards and electronic devices for security.

Smart Contract: A program running in a trust-minimized manner on a public blockchain or a DLT network. Smart contracts can be used to represent assets, create atomic operations, or implement Decentralized Finance (DeFi) protocols.

Tokenization: Process of issuing a financial asset on a shared, programmable, and trust-minimized platform. This process involves legal and technical operations.

Trust Minimizing Technologies (TMT): Technological tools and methods employed to minimize the need for trust among parties of a financial transaction, e.g., cryptographic signatures and secure elements.

Zero-Knowledge Proof: A cryptographic method allowing one party (the prover) to prove a statement's validity to another (the verifier) without revealing extra information. They can be used to ensure that financial transactions have been executed according to coded rules without revealing their details.

Introduction

Programmability in payments and settlement has yet to fully realize its potential to support policy goals such as fostering innovation, enhancing efficiency, improving safety and reducing fragmentation.[2] Active experimentations, such as Central Bank Digital Currencies (CBDC) and asset tokenization,[3] and even live implementations are emerging in the financial sector. However, technical, regulatory, and financial risks introduced by new capabilities like smarts contracts need to be understood and addressed. This paper aims to explore these points and identify an optimal balance between well-controlled systems and more innovative approaches.

In the context of payments and settlements, programmability is the capability to perform financial operations through logic implemented in computer programs.[4] These programs can read balances, trigger payments, or interact with more advanced features on behalf of users (Figure 2). For illustration, programmable access to account information at the retail level can enable the creation of a visual dashboard consolidating assets of an individual across multiple financial institutions. At the wholesale level, programs can safely execute operations such as Delivery-vs-Payment (DvP)[5] settlement mechanisms.

In practice, payments and settlements are executed by systems operated by financial institutions, technology firms and central banks. These payment and settlement systems[6] provide services that facilitate financial transactions, serving both direct participants such as banks and Fintech companies, and, indirectly, their clients such as businesses and consumers. As a result, the development of programmability in the context of payments and settlement primarily affects the financial industry, with potential broader effects.

An incomplete understanding of the innovations that programmability can bring has occasionally resulted in setbacks and missed opportunities. Poor project and risk management have led to the failure of some digital finance initiatives.[7] Misunderstandings have sparked debates over the benefits and risks of programmability.[8]

[2] On the potential of programmability to reduce technical and financial fragmentation, see Banque de France (2023).

[3] Tokenization is the issuance of financial assets on a ledger that presents certain characteristics, such as being shared by several participants and providing trust. See Adrian et al. (2023), Lavayssière (2023) and Abraham et al. (2024).

[4] In computer science, a program is a set of instructions written in code that can be executed by a computer.

[5] Settlement mechanism ensuring the mutual conditioning of the transfer of a financial instrument and its corresponding payment.

[6] Payment and settlement systems are arrangements, infrastructures and schemes that facilitate financial transactions. They include card networks, Fast Payment Systems, Real-Time Gross Settlement Systems, and Financial Market Infrastructures.

[7] An early example is project Taurus, commissioned by the London Stock Exchange in 1983 aimed at unifying data environment and execution system for London's share settlement procedures. Despite investment of almost 500 million BPD, the project ultimately failed due to vested interests, regulatory challenges, and management issues (See Drummond 1996). See below for a description of the more recent DLT CHESS project.

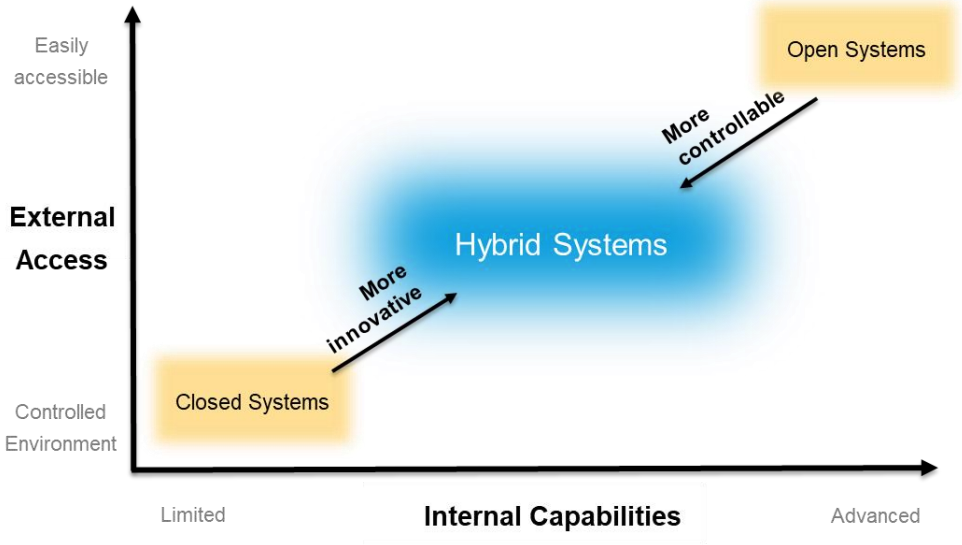
[8] See box *Challenges of Programming Money*.

This paper proposes a simple framework to understand programmability in the context of payment and settlement systems through two dimensions. 1) How, and through which program or interfaces, can systems be accessed and by whom; 2) and what tasks and functions can the system execute and support and with what guarantees. We elaborate on this framework throughout the paper, referring to these dimensions as *external programmatic access* — the ability for external participants to access data and functions via programs — and *internal programmatic capabilities* — how a system supports programs that offer execution guarantees. An example of such internal programmatic capability is the automatic execution of programs based on other changes that happened within the system.

To illustrate programmability, digital systems could be compared to music boxes and sound systems in the 20th century. A mechanical music box plays a melody engraved in its internal cylinder. The melody is predefined and cannot be changed. Only the speed can be controlled by rotating an external crank. In contrast, an analogic sound system offers greater flexibility internally and externally. High-quality audio tracks can be played and easily changed by swapping the vinyl record. Moreover, different controls, speakers, and amplifiers from various manufacturers can be connected to a turntable to offer a wide range of experiences.

Figure 1. Simplified Matrix of the Two Dimensions of Programmability

Both closed and open systems can be adapted to achieve better policy trade-offs



External programmatic access and internal programmatic capabilities define a matrix to classify programmable systems. To illustrate this matrix, we describe two design options at opposite corners: programmatically closed systems with low access and low ledger capabilities, such as some legacy systems, and open systems with maximum access and extensive capabilities, such as permissionless programmable

blockchains[9] (Figure 1). Between these extremes lie hybrid systems with varying degrees of programmability which we will explore further, such as Open Banking frameworks, modern Fast Payment Systems (FPS), or permissioned Distributed Ledger Technologies (DLT) arrangements.

This paper explores these dimensions to propose a structured approach aimed at informing and guiding the development of programmable finance. The first section discusses how *external programmatic access* can be adversely impacted in closed payment and settlement systems by characteristics such as non-standardized interfaces, opaque internal systems, and disjointed services. The second section describes how enhanced *internal programmatic capabilities*, through standardized technology stacks, distributed computing, delegation of services and roles, can offer robust and sophisticated execution. The last section uses these two dimensions to show how different strategies can be combined. By improving upon opposite models and mitigating their respective drawbacks, hybrid models of programmability can potentially achieve better policy and compliance trade-offs.

This framework contributes to the discussion on programmability requirements for payment and settlement systems. The adoption of common concepts is necessary for designing and regulating the next generation of financial infrastructures. As infrastructures for tokenization, retail, and wholesale CBDCs are being developed, this paper contributes to bridge policy discussions with technical considerations. The insights provided in this paper set the foundation for further research and discussion on programmability, including transition management and risk mitigation.

[9] We define permissionless programmable blockchains as blockchains which can be accessed and validated by anyone and that offer advanced internal programmatic capabilities. However, the two properties are not necessarily synonymous, see section *Internal Programmatic Capabilities of Open Systems.*

External Programmatic Access in Closed Systems

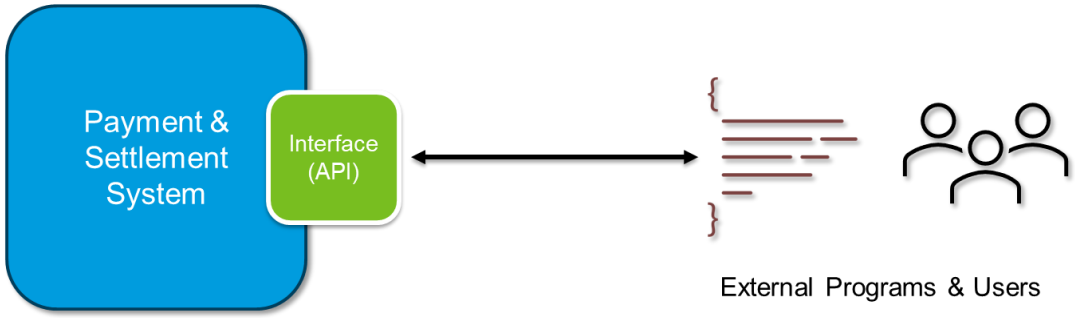
Payment and settlement systems already support financial operations that can be triggered by code. For instance, these systems can offer functions such as the ability to read balance inquiries, or to send payment instructions.[10] We define *external access* as the capability to access data and to trigger functions within the payment and settlement system, by external participants. [11]

Benefits of Opening Programmability

Thanks to Application Programming Interfaces (APIs), functions of a system can be accessed by external participants.[12] APIs are collections of procedures and functions that enable authorized external parties to access digital resources. An API gateway handles external requests and authorizations (Figure 2). In the analogy of a sound system, external components such as a microphone or speakers exchange audio signal via input and output sockets (or ports) of the turntable. These sockets are the interface for audio signals like an API is the interface for programs.

Figure 2. External Programs Access the Payment and Settlement System through APIs

External programs interact with the payment and settlement system via the API (in green) on behalf of institutional and retail users.



This simple model of programmability facilitates automation, interoperability, and innovation across systems. For example, merchants can connect their systems to functions such as payments, rewards, and additional financial services. Facilitating payments is even a central part of some business models, such as

[10] The basic functions available are sometimes called "primitives." See Deutsche Bundesbank (2020) on the notion of trigger solution and Lavayssière (2024). More advanced features are available in restricted environments, see section *Hybrid Systems*.

[11] External programmatic access enables what Hojo and Hatogai (2022) call "external programming approaches" and Toh et al. (2004) "client-side programmability" (as opposed to "bank side programmability").

[12] While the term "API" is generic, it commonly refers to a particular type of interface: standard web API, which typically employs REST (Representational State Transfer) architectural styles, standardized URLs (Uniform Resource Locator), and HTTP protocols.

freelancing platforms.[13] Fintech companies build their entire business models on offering services layered on existing core banking functions, using bank-provided APIs. As a result, businesses[14] and consumers may be offered a choice to use new financial services, and innovative features developed on top of legacy systems. Moreover, other payment and settlement systems can both consume, and offer APIs to facilitate interoperability across previously isolated systems.

As a result, advanced features can be built on top of simple interfaces. While powerful, the model described so far is relatively simple. It corresponds to the first of three levels of programmability defined in Lavayssière (2024) as "programmatically actionable": basic operations can be triggered via APIs provided by the system, such as a payment platform. Despite its simplicity, financial service providers can develop some powerful and innovative services by aggregating APIs from different external systems, offering advanced functionalities to a second or third tier of actors. [15]

Implementation Challenges in Closed Systems

Closed payment and settlement systems have limited options for programmability as they do not provide such access interfaces natively. These systems consist of either closed-loop payment systems managed by a single entity or multiple ledgers managed by separate legal entities, with communication facilitated by proprietary messages and common infrastructures. Access to data or the ability to trigger payments is limited due to legal or technical constraints. For example, central banks typically limit direct participation to their payment and settlement infrastructures to some categories of regulated financial institutions.

Legal constraints can significantly impede external access. Financial institutions may limit access to external developers due to regulatory constraints, such as the preservation of personal data, to mitigate operational risks[16] or, for strategic commercial considerations.[17] Opening access to facilitate integrations can raise questions of market structure and liability as illustrated by the 2024 bankruptcy of Synapse Financial Technologies.[18]

[13] E.g., Embedded payments in platform business models (Uber, Mercari, Malt) and larger technology ecosystems (Tencent, Ant group, Apple, Amazon).

[14] Businesses may directly use APIs to automate processes, potentially reducing human errors and lowering operational costs, such as integrations with Enterprise Resource Planning (ERP) for better reconciliation with invoicing and accounting, optimizing B2B payment routes, and improving on cash flow management.

[15] E.g., merchants can offer directly on their website Buy Now Pay Later (BNPL) to their customers via the integration to a payment processing platform (e.g., Stripe or Ayden) or a comprehensive solution (e.g., Shopify) that will solicit a Fintech company (e.g., Klarna or RatePay) or Banking as a service solution (BAAS, e.g., BBVA or Finastra) to offer a loan.

[16] N.B., each bank has a specific risk profile that determines its cybersecurity posture.

[17] E.g., regulatory capture and security concerns often lead to limited third-party access and operating hours. See Petry (2021).

[18] Synapse Financial Technologies is a United States-based company providing "Banking as a Service" to other Fintech firms. Its filing for bankruptcy in 2024 resulted in uncertainty regarding millions of dollars deposited by customers. The court cases highlight the fragility of the relationships between Synapse, its servicing bank and Fintech clients.

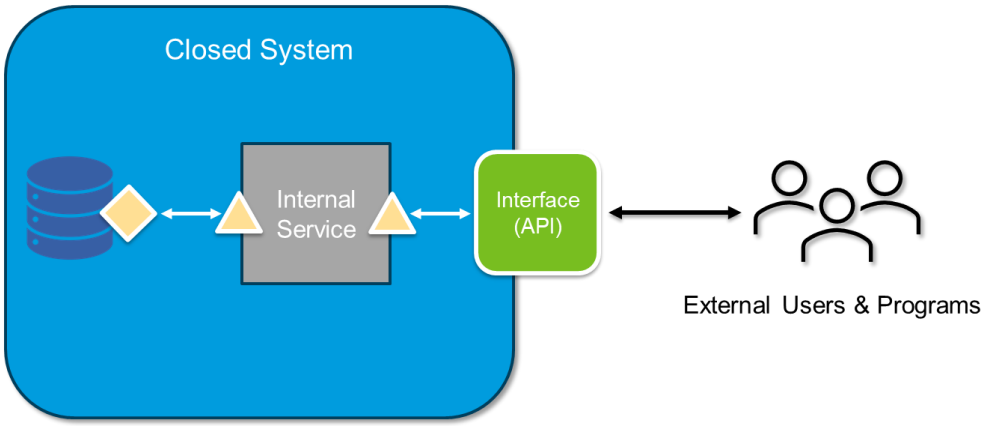
Restrictions are often reflected within the terms and conditions of contracts between participants or established by specific regulations.[19]

Interface quality and lack of standardization present major operational challenges. The performance of applications depends on the quality of interfaces and can fall short of meeting expectations regarding latency, failure rate, and availability. Moreover, the lack of standardization across systems leads to high integration costs. For example, if several banks offer an open API to access their services with slight differences in terms of format and response time, uniform service delivery across banks becomes impractical and costly.

The internal architecture of closed systems is often complex and not conducive to enabling new programmatic capabilities. Over time, these systems have evolved by adding and sometimes duplicating databases or other components, leading to a complex and fragmented structure that requires careful reconciliation. The complexity arises from multiple factors, including legacy components, disparate technologies, and inconsistent interfaces (Figure 3). Enhancing programmability in such legacy infrastructures is challenging and often incurs high costs and extended timelines, particularly when multiple entities are involved, such as in the case of multilateral platforms and Financial Market Infrastructures (FMIs).

Figure 3. Representation of a Closed System with One Ledger

The orange shapes illustrate the different internal APIs that can be involved when triggering services.



Regulatory efforts have been insufficient in addressing these challenges. Despite the push towards regulatory frameworks such as Open banking[20] to improve external programmatic access, these efforts have remained limited in scope and effectiveness.[21] The lack of strong technical and regulatory standards, and the

[19] For an example of restricting conditions to access a RTGS, see the section "Harmonized Conditions for participation in Target" in Guideline (EU) 2022/912 of the European Central Bank of 24 February 2022 on a new-generation Trans-European Automated Real-time Gross Settlement Express Transfer system (TARGET).

[20] E.g., Japan's 2018 revision of the Banking Act and the 2016 Payment Service Directive (PSD2) in the EU.

[21] For a detailed analysis of Open Banking implementation in the UK, see Dinçkol et al. (2023).

market power of closed systems, perpetuates service fragmentation and stymies efficient operations across platforms and jurisdictions.

However, modern developments such as Fast Payment Systems show workarounds to develop programmability in closed systems. The development of Fast Payment Systems in Brazil, India or Australia has created the opportunity to develop common interfaces without changing core banking infrastructures.[22] Regulators, governments, banks, and Fintech firms have developed strong and widely accepted standards that facilitate payments between financial institutions. These strategies are built around existing systems, working on facilitating access to various actors and the quality of this access. Internationally, the same logic applies to projects such as Nexus, creating common API interfaces between countries.[23] Further developments of these interconnections beyond payments and other core banking activities, often referred to as "Open Finance," are anticipated.

[22] See section *Hybrid Systems*.

[23] Nexus is a BIS (2023) project establishing API gateways to facilitate cross-border interoperability between payment systems.

Internal Programmatic Capabilities of Open Systems

In contrast to closed systems, open systems provide architectures designed for easy access by external participants, and often feature high internal programmatic capabilities (upper right corner of Figure 1). Permissionless programmable blockchains, such as the Ethereum network, are a well-known example of such openness combined with programmability.[24] Their success relies on common standards and the availability of open-source code. Even the data stored in the shared ledger is accessible to everyone. Furthermore, these systems have evolved to be conducive to composability.

A Standardized and Transparent Environment

Permissionless blockchains have introduced a model that offers programmability through comprehensive distributed platforms. Ethereum, for instance, was developed in response to the limited internal programmatic capabilities of Bitcoin, proposing a general-purpose system customizable with code.[25] By providing a distributed infrastructure designed to be resilient, open, and able to exchange digital assets programmatically, permissionless blockchains have fostered rapid technological and business innovation.[26] This has led to the emergence of Decentralized Finance (DeFi), which leverages composability to incrementally build elements of a financial ecosystem.[27]

The standardized messages and communication protocols used by these systems contribute to their interoperability. Blockchain clients such as nodes, enable users and programs to interact directly with blockchain networks to access and modify the financial ledger (whereas for closed systems only internal functions can do so).[28] All nodes of the system have a similar interface, using common standards, facilitating developer adoption.

[24] "Permissionless" refers to the ability of anyone with an internet connection to read information, submit transactions and even participate in the validation of the information on the ledger without depending on any intermediary.

[25] Platforms such as Ethereum, Avalanche or Solana allow for the deployment of full programs, so-called smart contracts, enabling an advanced form of internal programmability known as "Turing Complete". Other models of programmability are promising, particularly using verifications via ZKPs such as Scroll or Hylé.

[26] For instance, zero-knowledge proof algorithms (ZkP) have been implemented in production within a few years of their academic discovery (SNARKS in Zcash, Bulletproof in Mimblewimble and Monero and STARK in Starknet).

[27] DeFi also presents challenges that require appropriate regulatory strategies. See Roukny (2022) and below.

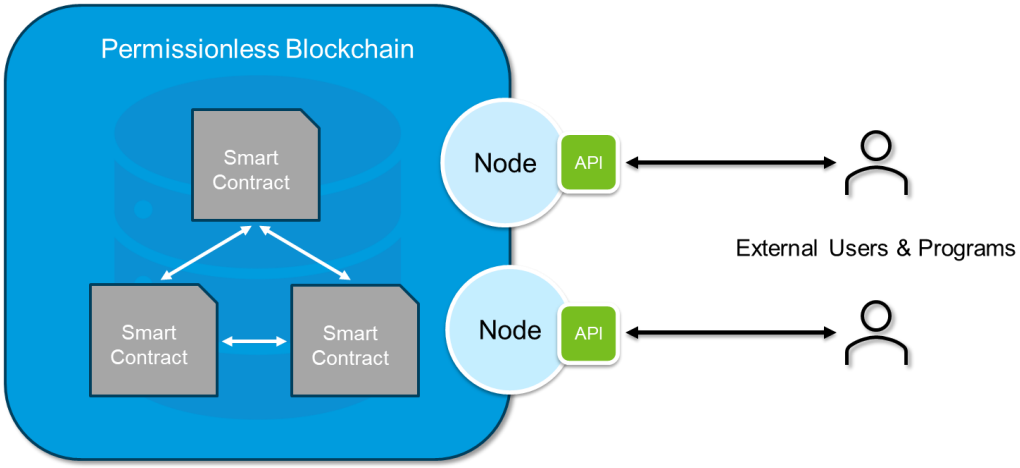
[28] Blockchain client software include (1) Web clients and applications, which interact with blockchain networks via third-party nodes; (2) Lightweight clients that verify transactions directly but do not download the entire blockchain, for less resource intensive hardware such as mobile devices; (3) Nodes, which download a complete copy of the blockchain and validate blocks and transactions against the blockchain's consensus rules.

The multiparty governance and decentralization guarantee external access and the correctness of execution. As no single entity controls access and validation, no single party can prevent access to users. The mutualized ledger reduces the need for reconciliation and guarantees data integrity under certain conditions.[29] Internal services themselves benefit from the same properties. Their execution is guaranteed when predefined conditions are met, ensuring consistency with the state of other entries in the ledger.

Programming abstractions provide a unified environment for software development by offering shared and standardized high-level concepts. These abstractions, such as "classes of objects," enable rapid development and collaboration. Developers do not need to understand the underlying details of pre-existing functions or objects that they can reuse. Smart contracts (Box 1), for instance, allow easy deployment of new internal services without requiring a developer to know everything about the underlying blockchain or other smart contracts.[30] The underlying blockchain is abstracted with notions such as virtual machines[31] and programs can interact with existing smart contracts via standardized functions or "primitives".[32]

Figure 4. External and Internal Communication in Permissionless Blockchains

Each node of the blockchain provides the same API (in green) to external users. Internally, smart contracts interact seamlessly as parts of a unique execution environment.



[29] Nakamoto (2009) describes how economic incentives and cryptographic proofs are used to ensure data integrity. However, each mechanism and set up requires a thorough analysis such as Amoussou-Guenou et al. (2021) and Buterin (2020).

[30] See Schär (2021) and section *Hybrid Systems*.

[31] The virtual machine is a runtime environment for smart contracts allowing them to be executed in a sandboxed environment. Validator nodes collectively run the virtual machine independently of any single node or its hardware.

[32] Decentralized Finance primitives are standardized and composable building blocks (e.g., token standards, automated market makers, or lending protocols) that enable the creation of complex financial applications. See box *Categories of Smart Contract*.

Jointly, these characteristics make permissionless blockchains "fully programmable".[33] This level of programmability is the most flexible and characterizes systems that provide external users with the greatest extent of control and customization. Programs are executed along with the system's internal ledger. As depicted in figure 4, users can operate a blockchain node to be part of the permissionless blockchain, granting them full access to validate network activities, initiate transactions, design, deploy, and manage smart contracts.

However, programmability on permissionless blockchains faces challenges in achieving effective integration with the broader economy. While Decentralized Finance proposes advanced programmability, practical aspects limit integration with the existing financial system.[34] For example, validators, in charge of verifying transactions and proposing new updates to the ledger, can be in the position of front-running users, a problem known as Maximal Extractable Value (MEV).[35] In addition, privacy and confidentiality, as well as scalability, are known challenges of public blockchains.[36] Current regulatory frameworks are also not perfectly suited for an ecosystem built around the principles of decentralization and trust minimization.[37]

[33] See Lavayssière (2024) and George et al (2023) for a description of different levels of programmability.

[34] See section *Trade-offs of Enhancing Programmability*.

[35] For an analysis of MEV, see Eskandari et al. (2019) and Daian et al. (2020).

[36] For example, the blockchain scalability trilemma is an early conjecture that scalability, security, and decentralized governance cannot be obtained all at the same time. However, its validity is debated. See Del Monte et al. (2020).

[37] For an analysis of the challenges of controlling digital assets in the context of capital flow management, sees He et al. (2022).

Box 1. Categories of Smart Contracts

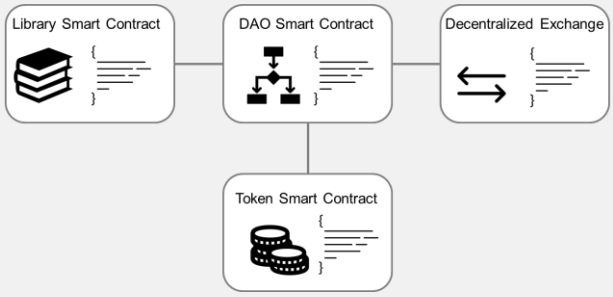
Smart contracts are computer programs that are executed in a manner that minimizes trust assumptions.[38] Compared to traditional programs, they operate in an environment that guarantees compliance with pre-determined rules. This "trust minimization" can be achieved by leveraging digital signatures, decentralized validation, and transparency.

Most smart contracts adhere to proven patterns and standards, which can be categorized as follows:

- **Token Smart Contracts:** These represent digital assets and enforce their foundational logic. Precise standards ensure interoperability with wallets and other smart contracts.[39]
- **Wallet Smart Contracts:** They provide functionalities to manage assets on-chain. For example, they can accommodate the use of multiple private keys or enable setting spending limits.[40]
- **Operation Smart Contracts:** These execute multi-pronged financial operations, such as using flash loans to exploit arbitrage opportunities and repay within the same transaction.[41]
- **Decentralized Autonomous Organizations (DAO):** DAOs support the governance of resources, upgrade mechanisms or treasury expenditure through token-based voting.
- **Decentralized Finance (DeFi):** These smart contracts provide trust-minimized financial infrastructures such as decentralized exchanges and lending platforms.

Most of these functionalities are provided by a combination of smart contracts. For example, a DeFi protocol can include tokens contracts, a governing DAO, and libraries[42] (Figure 5).

Figure 5. Example of a DeFi Protocol Combining Smart Contracts.



[38] The initial concept of smart contracts was to leverage cryptography to enforce agreements. However, smart contracts should not be conflated with the digitalization of contracts. See Szabo (1997).

[39] E.g., ERC-20 on Ethereum or FA1.2 on Tezos. They can also be part of the language or its main library (e.g., Aptos, Sui, and Solana).

[40] E.g., Gnosis Safe, Argent, and the "Account Abstraction" standard. For an empirical approach, see Benetti and Piazza (2024).

[41] For examples of adversarial usage of Operation Smart Contracts, see Qin et al. (2021).

[42] Library Smart Contracts provide functions that are used by other smart contracts. Other types of purely technical smart contracts can be used as upgrade mechanism or to generate other smart contracts.

From Advanced Programming Features to Composability

The advanced internal capabilities of these open systems allow programs to be used to define new financial assets, control transfer of ownership and manage accounts. While capacities of closed systems are defined internally, external participants can contribute to the evolution of the open systems. In the context of finance, Lavayssière (2024) proposes three main categories of programs available to external participants: assets, accounts and transfers of ownership.

Programming assets involves defining the core rules that safeguard the digital functioning and integrity of an asset. Existing financial assets are constrained by rules and characteristics defined by legal frameworks or contracts. In their digital version, code is used to define these constraints. For example, a program can define the logic for issuance and transfer.[43] The code serves as the asset's DNA and is executed at each operation to preserve the integrity of the asset.

With advanced internal capabilities, new assets can be programmed and added to a financial platform. For example, accounts receivable can be tokenized to facilitate factoring or other financing operations.[44] Additional rules can be embedded to ensure the integrity of the asset or constrain its usage. For instance, the program can ensure that some shares cannot be transmitted without the approval of other shareholders. Embedding a mandatory confirmation in the code provides guarantees to all shareholders. However, in the case of money, asset programmability is a more controversial topic.[45]

Programmability can also be used to manage accounts and set rules based on user preferences, business contracts, or regulations. Account management rules can include the ability to create, delete, or temporarily suspend accounts. Code can also be used to set constraints such as limits on stored value and transaction volumes, and to enforce security measures such as two-factor authentication checks. For example, users could set a weekly spending cap for budgeting purposes. Technologically, some of these features can be implemented directly within a payment instrument such as a card, or a smartphone based digital wallet.

Another advantage of programmable systems is their ability to execute transfers and enable automation, conditionality, and composability of financial transactions. Automation can accelerate processing by executing payments without manual intervention, such as in electronic toll collection systems. Conditionality introduces a requirement for settlement, i.e., requiring certain conditions to be satisfied to trigger a transaction.[46] For instance, a real estate transaction involves multiple entities, including lenders, land registries, real estate agents, the seller, and the buyer, each playing a different role and setting their own conditions for transferring ownership. With programmed conditional transfer, a payment could be automated

[43] See "token smart contracts" in box *Categories of Smart Contracts*.

[44] See BISIH and HKMA (2023) Project Dynamo. More advanced examples could include repo operations.

[45] See box *Challenges of Programing Money*.

[46] For an economic analysis, see Kahn and van Oordt (2022).

in theory from the buyer's to the seller's bank account. However, such possibility would require legal and technical changes beyond mere payment systems, such as compatible automated land registries.

Ultimately, the benefit of these internal programmatic capabilities is composability, which is the ability to combine multiple services and components seamlessly. When information is standardized, reliable and secure,[47] programs can easily use the output of other programs as input.[48] Composability allows different services, possibly developed by different external participants to interact, enabling new solutions like secured financing. For example, new assets can be created to represent a basket of other assets. The original assets are locked programmatically and can then be used for any operation, such as collateral for lending services, without requiring the involvement of the initial issuers of the assets.

Technically, these advanced programmatic features are not unique to permissionless blockchains. Several technologies can offer similar capabilities, but they must be arranged in a way to provide strong guarantees.[49] For example, the code defining an asset, such a Central Bank Digital Currency, must be protected from any external or unwanted modification. As shown by the ECB (2024) for a wholesale CBDC, objective criteria can be established and measured such as efficiency, robustness of code execution, and finality. While specific technologies can facilitate such design,[50] the overall architecture and other factors such as governance and security measures must be carefully considered.

[47] See Brammertz (2017) proposing the smart financial contract standard.

[48] Programmability can also play an informational role during transfers. Certain conditions could result in transferring information without moving assets, such as in the cases of transfer failures, mandatory reporting or triggering operations in another system.

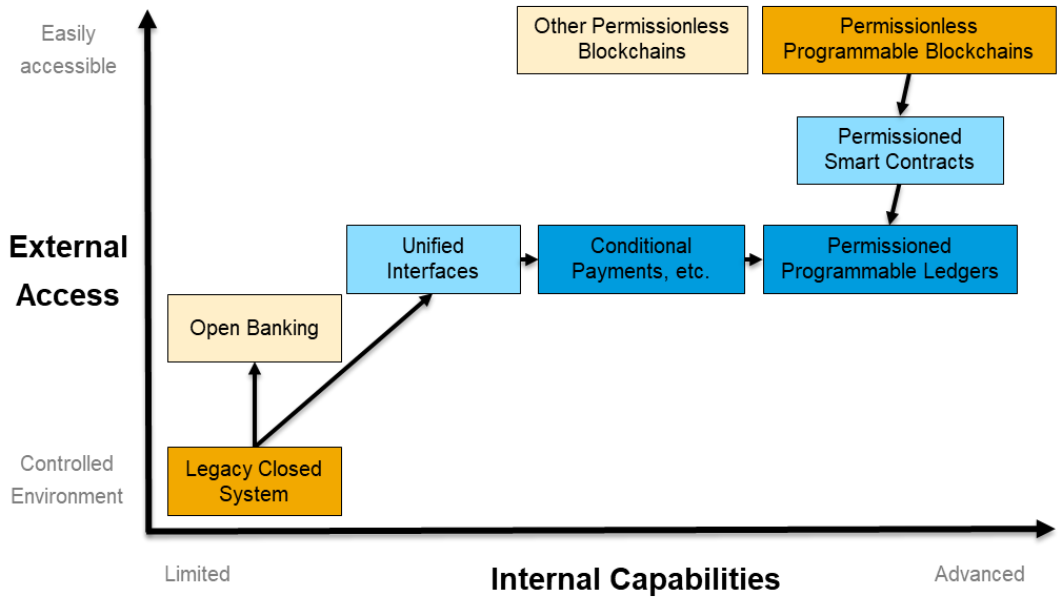
[49] For an early analysis of the requirements for a potential Canadian CBDC, see Veneris et al. (2021).

[50] Such as Zero-Knowledge Proofs, Multiparty Computation and Secure Elements. See Lavayssière (2024).

Strategies for Improving Programmability

There are many ways to design systems that fall between completely closed and completely open. The two extreme types of systems discussed above can be adjusted to satisfy the regulatory rigor required by banking and finance, while leveraging the potential for programmability to transform the sector. Many payment and settlement systems today use intermediary approaches, which we call "Hybrid Models." By using the framework proposed in this note, one can better understand how a range of options for programmable models can help meet specific objectives (Figure 6). More practical details on possible strategies are provided in the Appendix.

Figure 6. Hybrid Models can be Evaluated along the Two Dimensions of Programmability



Hybrid Systems

Hybrid systems aim to balance the characteristics of both closed and open systems. They make it easier for external users to access the system and enhance the system's internal capabilities in a controlled environment. While any system that balances the extreme models described above could be considered hybrid, we focus on modern payment and settlement systems that are created through coordinated efforts. Such systems tend to use standardized interfaces, provide transparent access, offer advanced features, and have shared governance.

The first step toward hybrid models is the introduction of shared interfaces and coordinated ecosystems. In hybrid models, interfaces are standardized and services that facilitate communication between participants

are introduced.[51] These models require tight coordination, often achieved through public-private partnerships. For example, the Unified Payment Interface (UPI) was launched by the Reserve Bank of India in cooperation with the private sector.[52] It provides a standardized interface and common services. Other modern Fast Payment Systems, such as PIX in Brazil and New Payment Payments Platform (NPP) in Australia, have achieved comparable results. In parallel, countries have harmonized nationally and internationally through simple payment communication protocols, such as QR codes.[53] Moreover, services relevant to all participants are shared and executed on a common digital infrastructure, such as liquidity saving mechanisms and identity.[54] Such public and private efforts foster harmonization and the rollout of new features within ecosystems while leveraging recent technologies and digital architectures.[55]

The second step involves adding programmable capabilities, such as vouchers and recurring payments.[56] Large financial exchanges often offer advanced operations within their respective ecosystems.[57] "Programmable", or "conditional", payments such as delayed, bundled, and recurring payments, can streamline business operations and enhance trust among parties.[58] Such features are considered in the design of modern payment and settlement systems.[59] For example, payment service providers in the European Union prepare dynamic future-dated payments via a premium API model.[60]

The third step is to establish appropriate governance to align the system's programmable capacities with specific goals. Several entities contribute to the operations and functioning of programmable payment and settlement systems. Therefore, governance bodies can align the coherence of standards and operations with the needs of the ecosystem. Permissionless blockchains have multipolar governance with sometimes contradictory objectives.[61] In contrast, the governance of hybrid models needs to balance legal requirements and innovation. Multilateral organizations and cooperatives such as SWIFT provide interesting models. Technologically, "permissioned blockchains" prevent undesired entities from interacting with the data and participating in the governance, while preserving the same internal programmatic capabilities as permissionless blockchains. The balance can be more subtle, such as in the case of "permissioned smart contracts". Instead of

[51] Such as an API Hub, a service that facilitates connections and provides documentation. See BIS project Icebreaker.

[52] Fast Payment System officially launched in 2016 by the National Payments Corporation of India (NPCI). See Cornelli et al. (2024).

[53] See World Bank Group (2021). Recent examples include cooperation between Mauritius and India.

[54] Such as "Aadhaar" identity service in India and BIS project Mandala.

[55] See Arcese et al. (2021) and Caricato et al. (2022) for an in-depth analysis of fast and resilient central bank infrastructures.

[56] For example, UPI 2.0 offers recurring payments with AutoPay and prepaid Vouchers. See Alonso & al (2023).

[57] FMIs can offer conditional payments, automated prefunding, defunding of accounts, or earmarked liquidity (Castelle et al., 2016).

[58] These hybrid models of programmability enable the second of the three levels of programmability defined in Lavayssière (2024), "configurable programmability", which introduces the ability to customize services by sharing them amongst several participants, and by letting each participant propose how their own needs can be programmed into that shared service.

[59] E.g., in the FPSs mentioned above. For a CBDC design including such features, see Bank of England (2023).

[60] SEPA Payment Account Access (SPAA) EPC (2022). See also New Payment Architecture (NPA) by Pay.uk.

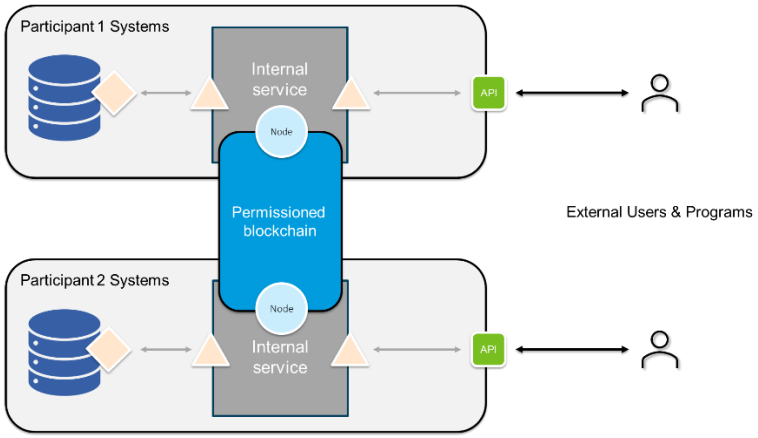
[61] For a primer on permissionless blockchain governance, see De Filippi et al. (2020).

using a different infrastructure, permissioned smart contracts control permissions programmatically on top of permissionless blockchains, by using mechanisms such as whitelisting.

Extensive research is underway to adapt the model of permissionless blockchains and their ecosystem into the context of traditional finance. Projects supporting programmable financial platforms include private initiatives (Onyx, SDX),[62] national CBDC platforms (DREX, Digital Rupiah),[63] and multilateral cross-border payments (Marianna, X-C platforms).[64] Other projects seeks to connect tokenized forms of money with payments (Monerium, Gnosis Pay)[65] and traditional financial instruments with DeFi infrastructures (SG-Forge, Backed, Spiko).[66] Efforts to create a global permissioned platform have started with private initiatives[67] to public-private partnerships such as project Agorá. While most of these projects are in early phases of exploration, these hybrid financial systems are more open, interconnected, and robust than most existing payment and settlement environments. As some early failures have shown, [68] it will take time to properly design platforms and tokenization projects with the right balance of interconnectivity with the broader financial system, balanced governance, regulatory compliance, and maintenance costs.

Figure 7. Example of Connecting Permissioned Blockchains with Legacy Infrastructures.

Participants to the payment and settlement system share a common platform to exchange information and assets.



While many systems solve regional or specific issues, they often operate in isolation, with limited standardization or coordination among them. Budau and Tourpe (2024) refer to these as "islands of

[62] Onyx is the blockchain division of J.P. Morgan. SDX is a subsidiary of SIX, provisioning a stock exchange and a CSD on a DLT.

[63] See Central Bank of Brazil (2023), Orestes and Townsend (2023) and Bank Indonesia (2022).

[64] See BISIH, Bank of France, Monetary Authority of Singapore, and Swiss National Bank (2023) and Adrian et al. (2023).

[65] Monerium is a stablecoin offering integration with SEPA payments. Gnosis Pay offers a self-custodial debit card.

[66] Forge, a Société Générale subsidiary, has issued bonds on-chain and used MakerDAO for refinancing. Backed issues tokenized treasury and corporate bonds. Spiko is a startup offering Money Market Funds as ERC20 tokens.

[67] E.g., Canton Network or Regulated Liabilities Network.

[68] E.g., Diem, which aimed at creating a private blockchain network, the initial Quorum project, intended as a fork of open-source clients, or the Australian Securities Exchange's project of creating a DLT infrastructure (DLT CHESS).

harmony within a sea of diversity of jurisdictions, technologies, and rules." The ASAP[69] model they propose aims to facilitate "cross-islands" collaboration at four distinct levels: the "Platform" layer focuses on the settlement and storage issues, which can for instance, be implemented with a permissioned blockchain with nodes owned and operated by clearly identified participants. They can deploy smart contracts on this layer, as shared services. Additional participants can expose either their own services, or re-composed services on the "Service" layer. Smart contracts also allow the programming of custom assets,[70] corresponding to the "Asset" layer. Finally, financial institutions and Fintech firms can provide an "Access" layer to external users via their own interfaces, such as QR codes or digital wallets. This layering of the technology stack is why Toh et al. (2024) estimate that permissioned bank operated blockchains blur the lines between "client-side programmability" and "bank-side programmability". This is because in hybrid models, more external actors can deploy programs that access and interact with the bank's financial ledger. By delineating external access and internal capabilities, our framework aims to clarify these lines. Defining hybrid systems through these four layers shows which different actors are working on which layers and where would interoperable connections happen. Figure 7 illustrates how interoperability can be established through the first layer, as proposed by the Global Layer One project.[71]

Trade-offs of Enhancing Programmability

Given the unique and critical nature of financial services, harnessing the benefits of programmability requires extra caution. Other industries have demonstrated that programmability can foster vibrant innovation. However, it can also introduce security, political and economic challenges.[72] Applying programmability to financial services demands a careful approach, as they are held to high expectations for security, resiliency, privacy, and fraud prevention.[73] Traditional ways of reducing risks, like heavily restricting access, would limit the benefits of programmability. Therefore, risks need to be understood and managed at the design stage and require novel approaches to cybersecurity.[74]

Programs do fail, be it due to software bugs, human errors, or the malicious intent of fraudulent actors. Even seemingly minor issues can escalate into complex and costly situations. While the possibility of failure is inherent in digital systems, their probability can be reduced, and their consequences minimized. This calls for anticipatory measures encompassing advanced and automated testing, audits, improved technological platforms, monitoring, recovery plans, as well as robust regulatory frameworks.[75]

[69] ASAP describes a technology stack for digital assets with layers: Access, Services, Assets, Platforms.

[70] As defined in the *programming asset* section of Lavayssière (2024).

[71] See MAS (2024).

[72] For a complete analysis of ecosystems and innovation, see Iansiti and Levien (2004).

[73] See CPSS-IOSCO (2012).

[74] A complete risk analysis would be the basis for a future note.

[75] See Zhou et al. (2022), Greene et al. (2018) and Lavayssière (2024).

When these technical failures occur, they can result in cascading effects throughout the financial system. A malfunction in one transaction might be replicated in multiple similar transactions or set off a chain reaction.[76] The breakdown of one platform can send ripples across the financial landscape. Interoperability and composability of financial services not only amplify the likelihood of mismatches between code and its intended function, but also magnify the resulting implications. Therefore, remedial strategies must be comprehensive, addressing the whole combination of services and platforms.

Moreover, financial services attract malicious actors looking to exploit vulnerabilities for financial gain. By combining services, and leveraging internal programmatic capabilities, these adversaries can orchestrate intricate fraud schemes, market manipulations, and crisis acceleration.[77] Complexity grows as the number of platforms and participants increases. In DeFi, attackers leverage the ability to create multi-pronged transactions to launch complex attacks, combining technical and economic elements. For perspective, the automation of trading in financial markets has resulted in several market and technical failures,[78] and the later emergence of High-Frequency Trading (HFT).

Fraudulent schemes and market manipulations could even originate from the platforms' operators. In conventional financial market infrastructures, platform operators may leverage their role to gain advantages.[79] There have been several instances of such frauds, amplified by the consolidation of exchanges in a handful of global corporations, as shown by Petry (2021). Permissionless blockchains are particularly prone to such risks, as decentralization and the censorship resistant design cannot easily prevent unwanted access, and manipulation.[80] Ensuring overall safety and market integrity of new platforms may necessitate specific supervision mechanisms.

Furthermore, the increased speed of automation and straight-through processing can elevate failure rates and facilitate fraud. Shorter settlement times shrink the window for compliance checks and surveillance. Faster transactions and instant settlements can enable multiple-hop frauds and money laundering schemes. With the modernization of infrastructures, oversight mechanisms must evolve at a commensurate pace.

Artificial Intelligence (AI) represents advanced forms of programs with transformative benefits, and new risks. AI is used in Finance for fraud detection, Know Your Customer (KYC), credit worthiness assessment or investment advisory tools. The development of programmability may facilitate the usage of AI by offering high-quality data and enabling more programmable actions such as trading[81] or payment agents.[82] However, the

[76] See Deloitte Germany (2021) on the consequences of the failures of Target.

[77] See Chakravorti (1999) and Rose (2023).

[78] E.g.: October 19, 1987 "Black Monday" in US stock markets partly due to trading bots with similar behaviors.

[79] Forms of market manipulation by platform operators include censoring, frontrunning, or order alterations.

[80] See section *A Standardized and Transparent Environment*.

[81] Autonomous trading agents can analyze market data, find investment opportunities, or even execute trades without human intervention.

[82] With "Autonomous Payments", payment agents could take financial decisions on behalf of users within predetermined parameters.

integration of AI in finance introduces concerns related to data privacy, algorithmic bias, explainability, and concentration of power. For example, an AI agent, or a combination of AI agents (an emerging trend know as Agentic AI), could create complex algorithmic operations in financial markets. The risks are not only related to faulty technology or bad actors, but also in functioning systems that transcend oversight.[83]

On the other hand, code can become a more intricate part of the security strategy for financial platforms. While current cybersecurity in the financial industry partly relies on processes and ring-fencing, the development of programmability can allow the use of code to automate various security-related tasks, from predefined contingencies to incident response.[84] AI technologies could assist developers and leverage data for monitoring. In DeFi, AI is already used to audit smart contracts and monitor economic outcomes.[85] Moreover, programmability can be used to provide the contingencies necessary for trust in a more complex environment.

Another category of concerns is the impact of deterministic properties of code in dynamic financial scenarios. Nuances of human situations contrast with the binary nature of digital systems. Coded implementations inherently simplify real-world situations. While humans adjust their interpretations based on context, including laws, social norms, and ethical consequences, code simply executes as written. An example is the difficulty of transcribing the legal notion of 'reasonable efforts' that expects a custodian to implement measures to safeguard funds when confronted with an issue. This rigidity can lead to unexpected costs that outweigh the initial benefits.

As a result, the technological implementation of laws and regulations by technological means is a delicate endeavor. Enforcing compliance rules *ex ante* represents a significant change from the traditional blend of private compliance and *ex post* law enforcement. System designers and operators would have a primordial role. If implemented broadly, coded rules can have a large effect, such as phasing out certain activities.[86] Additional geopolitical complexities could emerge when domestic platforms are utilized in international contexts.[87] The nature and extent of these challenges might differ based on the scope of policy implementations, ranging from AML/CFT to advanced environmental policies.

Interestingly, the inherent flexibility of programmability as a process could offer solutions to some of these challenges. Financial services can be tailored to fit various user needs, contexts, and activities. Financial infrastructures could encode integrity-related rules, while other topics are addressed in layers closer to the users. Targeted AI and privacy enhancing technologies could provide additional flexibility in specific contexts. On multilateral financial platforms, jurisdiction -specific rules, especially those concerning compliance or central bank operations, can be customized by the respective national bodies.

[83] See Aldasoro et al. (2024).

[84] Companies such as Palantir or Shift Technology provide AI tools to proactively flag any abnormal data and behavior.

[85] As offered by plethora of new startups such as 0x0.ai, chainGPT, and Bunzz Audit.

[86] According to Lessig (1999), code can become a form of *de facto* law, where anything not allowed initially becomes forbidden.

[87] E.g., BIS project Mandala explores such consequences between Singapore, Malaysia, Korea, and Australia.

Box 2. Challenges of Programming Money

While programming payment and settlement systems offers numerous benefits, the concept of "programmable money", the ability to attach logic to monetary units, is controversial.[88]

When money is in a digital form, code guarantees its core functions as means of payment, a store of value and a unit of account. With advanced internal programmatic capabilities, assets and payment instruments can embed innovative logic and dynamic rules, such as expiration dates, geofencing, or a list of acceptable recipients. However, introducing programmable controls at the retail level would raise policy questions as the public would need to trust in the fairness and integrity of these systems.

Historical lessons from Digital Rights Management (DRM) illustrate potential pitfalls.[89] Such systems offer precise control on the usage of digital content. For example, libraries can lend eBooks in a controlled manner. However, they also limit user rights, such as the ability to self-repair, to make a private copy or to use readers customized for certain disabilities. These controls also add legal and commercial liabilities to the entities involved. Additionally, with the possibility to track each use, data centralization strengthens dominant platforms at the expense of smaller entities.[90]

Transposed to the financial sector, similar dynamics could lead to disproportionate control by entities managing the enforcement mechanisms, potentially including the public sector in the case of CBDCs. Moreover, central banks face particular challenges in implementing programmable money due to their unique institutional position. Encoding rules and restrictions into the currency might be perceived as overstepping the traditional boundaries of central bank authority.[91]

One approach to mitigate these concerns is to issue new monetary instruments for specific purposes akin to vouchers. Government programs like Italy's "Bonus Cultura" have successfully moved from a paper-based solution to a digital system.[92] The concept of "Purpose-Bound Money" proposed by the Monetary Authority of Singapore (2020) is an exploration of extending programmable capabilities without affecting the fungibility of the initial assets. The Bank of Thailand (2024) experimented during its digital baht pilot how such features can contribute to financial education.

[88] E.g., "The digital euro therefore cannot be a programmable money." Eurogroup (2023).

[89] DRMs are software and hardware technologies used to control access usage of digital content. For instance, when a user plays a music file, music players can check if this usage is compliant with intellectual property rights.

[90] E.g., by paving the way predictive analysis, algorithmic recommendation, and targeted ads. See Lamdan & al. (2023).

[91] For a legal and institutional analysis of central banks' role, see Walker et al. (2020).

[92] "Bonus Cultura" offers five hundred euros to anyone reaching the age of majority to pay for cultural events, museums entrances, books, and music, as well as theater and language classes. Launched in 2016, it initially required merchants to register with the relevant ministry and involved the distribution and redemption of paper vouchers. It has been digitized into simple e-coupons using a web app linked with the national identity system (SPID).

Conclusion

While programmability is already used in digital payment systems, it is exploited below its potential. Programmability holds substantial promise to make financial services more innovative, open, interconnected, and resilient. However, realization of this potential is constrained by uncoordinated approaches and legacy systems. This note showed that programmability can be evaluated along two dimensions: external access and internal programmability capabilities which can be used to compare different payment and settlement systems. Any system or technology can be improved along these dimensions.

External programmatic access, **which represents the ability for exogenous actors and systems to leverage the services offered by the platform, can be improved with standardization and transparent access.** Common technology standards, documentation, and transparent terms of access can facilitate the use of functionalities offered by payment and settlement systems. Collaboration among stakeholders across private and public sectors, including international financial institutions and standard setters, can accelerate the establishment and adoption of relevant interfaces, in terms of quality and number of users.

Internal programmatic capabilities of payment and settlement systems can be expanded by improving the robustness of code execution and the programming features of the underlying infrastructures. These improvements may require more substantial change in technologies, either improving current infrastructures, or adapting new models while mitigating their drawbacks. Creating coordinated infrastructures, such as modern FPSs or identity systems, has proven to be an efficient modernization approach, providing fast standardization with adaptable functionalities.

While several benefits of enhancing programmability are apparent, the associated risks must be considered. Developing programmability might bring significant benefits in cost reduction and new opportunities. However, it also increases technical risks, from individual failures to more systemic ones. Moreover, long-term economic and political risks require further analysis.

This framework can help policymakers to better assess existing and new payment and settlement systems. In a complex and dynamic environment, strategies must look beyond the status quo and find the most effective arrangements. Realizing programmability's potential requires weighing trade-offs, enlisting diverse expertise, and coordinating efforts, based on specific country or cross-border circumstances. Therefore, the concepts provided in this paper can facilitate discussions, coordination, and the development of fruitful ecosystems. International collaborative efforts towards developing common concepts and frameworks represent the best approach to ensure informed decision-making.

Appendix: Maximizing Programmability Potential

External Programmatic Access

	Standardization	Transparent Access
Objective	Share common interfaces and facilitate integration.	Guarantee access to data and function.
Key Trade-off	Balance customization that accommodates diverse needs with broad compatibility and interoperability.	Balance broad access with reliability, privacy, and compliance.
Available Strategies	• Develop and adopt code libraries and standardized interfaces. • Harmonize communication protocols to ensure seamless data exchange. • Utilize virtual machines for consistent execution environments. • Consider the implementation of shared services or gateways to centralize common functionalities.	• Standardize contractual terms to clarify conditions of access, including costs and reliability expectations. • Optimize system latency, speed, and availability while ensuring compliance with security and privacy standards. • Create comprehensive documentation and Software Development Kits (SDKs). • Establish a coordinated governance.

Internal Programmatic Capabilities

	Robust Code Execution	Advanced Programming Features
Objective	Improve reliability and reduce the occurrence of failures.	Simplify development and enable innovation.
Key Trade-off	Balance robustness with development costs of programs and the platform.	Balance advanced features with the complexity of integrations and platform management.
Available Strategies	• Provide tooling and best practices to developers. • Ensure predictable and reliable execution and validation processes. • Monitor code execution. • Invest in advancing the maturity of underlying technologies to support robust operations.	• Implement conditional payments that can handle advanced transaction criteria. • Segregate code execution. • Provide sensible abstractions. • Enhance the internal coherence of data and execution of programs. • Potentially support Turing completeness or adopt a verification model.

References

Agur, Itai, German Villegas-Bauer, Tommaso Mancini-Griffoli, Maria Soledad Martinez Peria, and Brandon Tan (2024) Trading Tokens: Benefits and Risks of Tokenized Financial Markets.

Adrian, Tobias and Tommaso Mancini Griffoli (2023) The Rise of Payment and Contracting Platforms.

Alonso, Cristian, Tanuj Bhojwani, Emine Hanedar, Dinar Prihardini, Gerardo Uña, and Kateryna Zhabska (2023) Stacking up the Benefits: Lessons from India's Digital Journey.

Aldasoro, Iñaki, Leonardo Gambacorta, Anton Korinek, Vatsala Shreeti, and Merlin Stein (2024) Intelligent financial system: how AI is transforming finance.

Amoussou-Guenou, Yackolley, Bruno Biais, Maria Gradinariu Potop-Butucaru and Sara Tucci Piergiovanni (2021) Rationals vs Byzantines in Consensus-based Blockchains.

Arcese, Mauro, Domenico Di Giulio and Vitangelo Lasorella (2021) Real-Time Gross Settlement systems: breaking the wall of scalability and high availability.

Caricato, Gianluca, Marco Capotosto, Silvio Orsini, and Pietro Tiberi (2022) TIPS: a zero-down time platform powered by automation.

Castelle Michael, Yuval Millo, Daniel Beunza, and David C. Lubin (2016) Where do electronic markets come from? Regulation and the transformation of financial exchanges.

Bank of England (2023) The digital pound: Technology Working Paper.

Bank of Ghana (2022) Design paper of the digital Cedi (eCedi).

Bank for International Settlements Innovation Hub (BISIH) (2023) Nexus: A blueprint for instant cross-border payments.

BISIH and Bank of England (2023a) Project Rosalind - Building API prototypes for retail BCDC ecosystem innovation.

BISIH and Bank of England (2023b) Project Meridian - Simplifying transactions through innovation.

BISIH, Bank of France, and Swiss National Bank (2021) Project Jura - Cross-border settlement using wholesale CBDC.

BISIH, Bank of France, Monetary Authority of Singapore, and Swiss National Bank (2023) Project Mariana - Cross-border exchange of wholesale CBDCs using automated market-makers.

BISIH, Bank of Israel, and Hong Kong Monetary Authority (2023) Project Sela - An accessible and secure retail CBDC ecosystem.

BISIH and Hong Kong Monetary Authority (2023) Project Dynamo - Catalysing innovation for SME growth.

Bank Indonesia (2022) Project Garuda: Navigating the Architecture of Digital Rupiah.

Bank of Japan (2023) Privacy Enhancing Technologies: Payments and Financial Services in a Digital Society.

Bank of Thailand (2024) Pilot Program - Retail CBDC Conclusion Report.

Banque de France (2018) Payment and financial market infrastructures in the digital era.

Banque de France (2023) Wholesale Central Bank Digital Currency Experiments with the Banque de France.

Bechara, Marianne, Wouter Bossu, Amira Rasekh, Chia Yi Tan, and Akihiro Yoshinaga (2023) Private Law Aspects of Token-Based CBDC.

Benetti, Zeno and Frederico Piazza (2024) Decentralised Finance: A categorization of smart contracts.

Berk, Ivo M. van den, Sligner Jansen, and Lutzen Luinenburg, (2010) Software Ecosystems: A Software Ecosystem Strategy Assessment Model.

Blum, Manuel, Paul Feldman, and Silvio Micali (1988) Non-interactive zero-knowledge and its applications.

Budau, Victor and Herve Tourpe (2024) ASAP: A Conceptual Model for Digital Asset Platforms.

Buterin, Vitalik (2020) Credible Neutrality as a Guiding Principle.

Brammertz, William and Allan I. Mendelowitz (2017) From digital currencies to digital finance: the case for a smart financial contract standard.

Central Bank of Brazil (2023) Motivations for the Brazilian Digital Real pilot.

Chakravorti, Sujit (1999) Analysis of systemic risk in multilateral net settlements systems.

Committee on Payments and Market Infrastructures (CPMI-BIS) (2023) Facilitating increased adoption of payment versus payment (PvP).

Committee on Payment and Settlement Systems - International Organization of Securities Commissions (CPSS-IOSCO) (2012) Principles for Financial Market Infrastructures.

Cornelli, Giulio, Jon Frost, Leonardo Gambacorta, Sonalika Sinha and Robert M. Townsend (2024) The organisation of digital payments in India – lessons from the Unified Payments Interface

Daian, Philip, Steven Goldfeder, Tyler Kell, Yunqi Li, Xueyuan Zhao, Iddo Bentov, Lorenz Breidenbach and Ari Juels (2020) Flash Boys 2.0: Frontrunning in Decentralized Exchanges, Miner Extractable Value, and Consensus Instability.

Danmarks Nationalbank (2023) Oversight of the financial infrastructure.

Deutsche Bundesbank (2020) Money in programmable applications.

Deloitte Germany (2021) Report on the external review of TARGET Services in the context of the incidents in March, May, August, October, and November 2020.

De Filippi, Primavera and Xavier Lavayssière (2020) Blockchain Technology: Toward a Decentralized Governance of Digital Platforms?

Del Monte, Gianmaria, Diego Pennino, and Maurizio Pizzonia (2020) Scaling blockchains without giving up decentralization and security: a solution to the blockchain scalability trilemma.

Dinçkol, Size, Pinar Ozcan, Markos Zachariadis (2023) Regulatory standards and consequences for industry architecture: The case of UK Open Banking.

Drummond, Helga (1996) Escalation in Decision-Making: The Tragedy of Taurus.

Eskandari, Shayan, Seyedehmahsa Moosavi and Jeremy Clark (2019) SoK: Transparent Dishonesty: Front-Running Attacks on Blockchain.

European Commission (EC) (2023) Proposal for a regulation on the establishment of the digital euro.

European Payment Council (EPC) (2022) SEPA Payment Account Access (SPAA) Scheme Rulebook.

European Central Bank (ECB) (2023) Progress on the investigation phase of a digital euro – third report.

European Central Bank (ECB) (2024) Proposed learning objectives for the Eurosystem exploratory work and Key Performance Indicators for the assessment of the findings.

George, Nikhil, Tadge Dryja, and Neha Narula (2023) A Framework for Programmability in Digital Currency.

Greene, Richard, and Michael N. Johnstone (2018) An investigation into a denial-of-service attack on an Ethereum network

Grossman, Sanford J., and Oliver D. Hart. (1986) The costs and benefits of ownership: A theory of vertical and lateral integration.

Iansiti, Marco and Roy Levien (2004). The Keystone Advantage: What the New Dynamics of Business Ecosystems Mean for Strategy, Innovation, and Sustainability.

He, Dong, Annamaria Kokenyne, Xavier Lavayssière, Inutu Lukonga, Nadine Schwarz, Nobuyasu Sugimoto and Jeanne Verrier (2022) Capital Flow Management Measures in the Digital Age: Challenges of Crypto Assets.

Hojo, Masashi and Junichiro Hatogai (2022) Realizing Programmability in Payment and Settlement Systems.

Kahn, Charles, and Maarten R.C. van Oordt (2022) The Demand for Programmable Payments.

Lamdan, Sarah, Jason M. Schultz, Michael Weinberg, and Claire Woodcock (2023) The Anti-Ownership eBook Economy.

Lavayssière, Xavier (2023) Tokenization of Financial Assets.

Lavayssière, Xavier (2024) The Design of Programmable Ledgers.

Lee, Alexander (2021) What is programmable money?

Lessig, Lawrence (1999) Code, and Other Laws of Cyberspace.

Lovejoy, James, Cory Fields, Kevin Karwaski, Madars Virza, Tyler Frederick, David Urness, Anders Brownworth and Neha Narula (2022) A High-Performance Payment Processing System Designed for Central Bank Digital Currencies.

Monetary Authority of Singapore (2022) Project Orchid Whitepaper.

Monetary Authority of Singapore (2024) Global Layer One – Foundation Layer for Financial Networks.

Orestes, Victor and Robert M. Townsend (2023) Brazil Central Bank Digital Currency: Improving Financial Infrastructure with programmability.

Petry, Johannes (2021) From National Marketplaces to Global Providers of Financial Infrastructures: Exchanges, Infrastructures, and structural power in global finance.

Qin, Kaihua, Liyi Zhou, Benjamin Livshits, and Arthur Gervais (2021) Attacking the DeFi ecosystem with flash loans for fun and profit.

Roukny, Tarik (2022) Decentralized Finance: information frictions and public policies.

Schär, Fabian (2021) Decentralized Finance: On Blockchain and Smart Contract-Based Financial Markets.

Szabo, Nick (1997) Formalizing and security on Public Networks.

Veneris, Andreas, Andreas Park, Fan Long, and Poonam Puri (2021) Central Bank Digital Loonie: Canadian Cash for a New Global Economy.

Walker, Martin C. W. (2020) Do we need programmable money?

World Bank Group (2021) The use of Quick-Response codes in payments.

Zhou, Liyi, Xihan Xiong, Jens Ernstberger, Stefanos Chaliasos, Zhipeng Wang, Ye Wang, Kaihua Qin, Roger Wattenhofer, Dawn Song, and Arthur Gervais (2022) SoK: Decentralized finance (defi) incidents.

PUBLICATIONS

Programmability in Payment and Settlement – Concepts and Implications

Working Paper No. WP/2024/177