BEN MILNE

The *Value* Layer

A field guide for building software that moves money

FOUNDATIONS & DERIVATIONS
EDITION ONE · 2026

The *Value* Layer

BEN MILNE

The *Value* Layer

A field guide for building software that moves money

AN INDEPENDENT EDITION

The Value Layer

A field guide for building software
that moves money.

First edition, 2026.
An independent edition.

Drawn from essays published at BENMILNE.COM
between 2021 and 2026, edited and expanded for this edition.

Set in Iowan Old Style and Hoefler Text
with metadata in Menlo. Diagrams set in vector by the author.

EDITION ONE · BUILD V1.7.0 · 2026-04-27

"In the same way HTTP let anyone publish a webpage, stablecoins and protocols are letting anyone publish to the value layer, global by default, always on by default, infinitely customizable for virtually free."

Contents

FOREWORD

How to read this *book*

This is a short field guide for people who write software that moves money. It is not a survey, and it is not a manifesto. It is a mental model, a small, deliberate set of primitives that I have used for the better part of a decade to organize the messy world of payments, settlement, and stablecoins into something legible.

The model is simple by design. Three primitives, *ValueType*, *TransferType*, and *Exchange* are enough to describe nearly every system I have shipped, integrated with, or read about. The Issuer is inherent in the ValueType (USD names the U.S. government; USDC names Circle), so it does not need a separate place in the grammar. The first part of the book defines the three. The second part shows how new things arrive as *derivations* of older things. The third part is about building: the physics that constrain what software can do with money, and the conventions and rules I have found useful when designing on top of the layer.

The appendices are a working reference. Appendix A is a starting atlas of ValueTypes and their TransferTypes. Appendix B is a glossary. Appendix C maps the primitives onto the live Brale API.

Four frameworks shaped how this is presented. Chiara Marletto's physics of *what is possible* is why I describe primitives as a specification of transformations. Albert Wenger's argument that attention has replaced capital as the binding constraint is why the Twelve Rules favor open standards over moats. Carlota Perez's account of how revolutions *install* before they *deploy* is the lens through which I read this moment as a turning point. Hobart and Huber's *Boom* explains why such phases happen, and why what is left behind defines the next era. All four are in Appendix D.

If you are in a hurry, read Part I and Appendix C. If you have an afternoon, read it in order. Either way, the goal is the same: a smaller grammar in your head, so you can build faster.

PART ONE

The *Primitives*

Three abstractions that, taken together, describe nearly every system that moves value across the internet. Learn them, and most of payments collapses into a small grammar.

VALUETYPE	*what moves*
TRANSFERTYPE	*how it moves*
EXCHANGE	*converting between*

CHAPTER ONE

What the Value Layer *Is*

When I say *the value layer of the internet*, I mean the immensely complex series of systems that support value exchange across the internet. After many years of trying, I have come to think of this layer as comprising just three primitives, three abstractions that, taken together, specify the set of transformations the system can perform.

The list is new in the way that distinguishing between RAM and a hard drive is new, once you have the distinction, you cannot stop seeing it, and most of the cost of the system collapses out.

The three primitives.

> VALUETYPE
>
> *What is moving, a unit of value with an identity. USD, EUR, USDC, SBC, BTC, an NFT, a fractional share. Each ValueType inherently names its Issuer.*

> TRANSFERTYPE
>
> *How it moves, a network or rail capable of transferring a ValueType. Wire, ACH, SEPA, Ethereum, Solana, Lightning.*

> EXCHANGE
>
> *The conversion from one (ValueType, TransferType) pair to another. Where economics enters the model.*

A note before Part I begins: the Issuer of a ValueType (Circle for USDC, Brale for SBC, the U.S. government for USD) is inherent in the unit itself, so it does not appear separately in the grammar. Everything downstream of an Issuer's decisions is inherited by every instance of that ValueType.

CHAPTER TWO

ValueType

A *ValueType* is what is moving. It is a unit of value with an identity, something a system can pick up in one place and put down in another, and still call by the same name on the other side.

The natural first instinct is to think in currencies. USD, EUR, GBP, JPY, MXN. That is correct, but incomplete. A stablecoin is also a ValueType: USDC, USDT, SBC. So is BTC or ETH. So is a tokenized money-market fund share, or an NFT, or a fractional interest in an Apple I. The model does not care whether the unit is fungible or unique, government-issued or protocol-issued, on a balance sheet or on a chain. It only cares that the unit has an identity that survives a transfer.

I lead with ValueType, not geography, because *geographies are a people thing rather than an internet thing*. The internet is ruled by protocols; the world is ruled by laws. One does not exist without the other, but a model anchored on geography forces you to redo the abstraction every time the user crosses a border. ValueType-first holds steady.

A few things to notice.

- USD on Cash App and USD in a wire are the same ValueType.
- USDC on Ethereum and USDC on Solana are the same ValueType, on different TransferTypes.
- USD and USDC are different ValueTypes, even though both are denominated in dollars.
- An NFT is a ValueType with a quantity of one.

Issuers are inherent.

Every ValueType inherently names its Issuer. USD names the U.S. government. USDC names Circle. USDT names Tether. SBC names Brale Inc. The Issuer does not need to be named alongside the ValueType in an instruction, the unit carries its Issuer's reserves, redemption guarantees, and compliance posture by virtue of being that unit. Choose your ValueTypes carefully; everything downstream is inherited from the Issuer's decisions.

CHAPTER THREE

TransferType

A *TransferType* is how value moves. It is a network, a rail, that knows how to debit one place and credit another, and produces some kind of receipt.

The bank rails are the obvious entries: ACH, wires (Fedwire, Swift, Target 2), SEPA and SEPA INSTANT, FEDNOW, RTP, FASTER PAYMENTS, PIX, UPI. Card networks, Visa, Mastercard, Discover, are TransferTypes too, with their own peculiar settlement geometry. So are blockchains: Bitcoin mainnet, Ethereum, Solana, Base, Arbitrum, Stellar, Tron. So are interop protocols: CCTP, LayerZero, Axelar, Chainlink CCIP. So are closed loops: PayPal, Cash App, Venmo, WeChat Pay.

What makes them all the same kind of thing is not their technology. It is their *job*: route value, clear it, settle it, and confirm it. They differ on cost, latency, bandwidth, hours, finality, geography, and trust assumptions, and those differences are the entire interesting surface area of the layer.

Anatomy of a TransferType.

- Routing, how the destination is identified.
- Clearing, when balances tentatively reflect the move.
- Settlement, when finality is established.
- Reversibility, what can claw the move back, and for how long.
- Hours, when the rail is open.
- Cost, fixed and variable, including foreign-exchange spread.
- Bandwidth, transactions per second the rail can absorb.
- Trust, who you must believe in for the move to be safe.

A protocol like Solana looks nothing like ACH on the wire, but the questions you ask of it are the same ones the Federal Reserve asks of Fedwire. Once you have the questions, the answer space is small enough to fit in your head.

CHAPTER FOUR

Exchange

An *Exchange* is the conversion from one (ValueType, TransferType) pair to another. It is the only primitive in the model that is not a noun in the system; it is a verb, with a price.

A wire of USD from a U.S. bank, converted into USDC on Solana and sent to a wallet, is an Exchange. So is a Visa authorization that ends as an ACH credit to a merchant's bank, or a Forex desk that prints EUR/USD, or an automated market maker swapping USDC for ETH in a pool. The mechanics differ; the role does not. Each Exchange takes ($ValueType_1$, $TransferType_1$) on the input side and emits ($ValueType_2$, $TransferType_2$) on the output side, with a cost.

The cost is where economics enters the layer. Spread, fee, slippage, float, opportunity cost, capital requirement, these are the tariffs of the value layer. Most of the historical revenue in payments lives inside Exchanges, often invisibly bundled into a TransferType's headline price.

Why call it out as a primitive.

Once Exchange has a name, two useful things happen. First, every funds flow becomes legible: instead of an opaque "the bank handles it," you can point to the specific Exchanges in the path. Second, design choices that used to feel like magic, does this product hold inventory, route to a market, prefund, lazy-settle?, become explicit, and therefore reviewable.

Anatomy of an Exchange.

- Input, ValueType, TransferType, amount.
- Output, ValueType, TransferType, amount.
- Rate, the price applied.
- Counterparty, who the system depends on for liquidity.
- Latency, when the output side is available.
- Cost, fee, spread, slippage, capital.

CHAPTER FIVE

How they *compose*

The three primitives compose into a small grammar. Every leg of every funds flow can be written as a tuple, and that is the only sentence the model speaks.

FIG. 1 · **A LEG OF A FUNDS FLOW.** VALUETYPE, TRANSFERTYPE, AMOUNT. THE ISSUER IS IMPLICIT IN THE VALUETYPE.

A funds flow is a sequence of legs joined by Exchanges. *Send usd by wire to an Issuer, who exchanges it to usdc on Solana, and forwards usdc on Solana to a recipient wallet* is a three-leg flow with one Exchange. Drawn at any level of detail, every payment system in production today reduces to legs and Exchanges.

Two consequences are worth noticing. First, design choices that look like architecture are usually choices about which legs are *your* legs and which are someone else's. A fintech that issues no ValueType and runs no TransferType is chaining other people's legs. A bank with its own core is one or two of its own legs deeper into the stack. There is no judgement in either; the model just makes the choice visible.

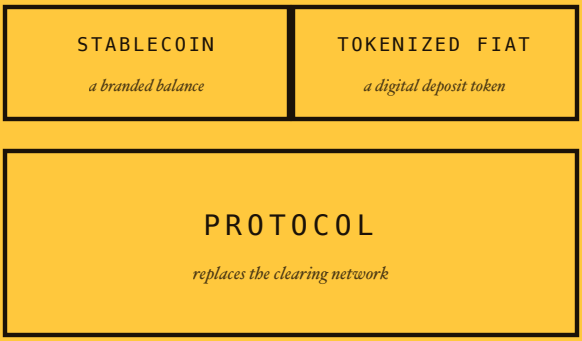
Second, a clean grammar makes systems composable. If your internal ledger speaks *(ValueType, TransferType, Amount)*, every new ValueType you support is one row of configuration, not a rewrite. That is the entire premise behind the Brale API, and behind every other API in this layer worth using.

PART TWO

Derivations

II

A derivation isn't a new invention. It's an evolution, a familiar primitive re-emerging in a new technical context, reshaped by what is now possible. The interesting motion in the value layer is almost always derivative.



CHAPTER SIX

A derivation is not an *invention*

A derivation is not a new invention. It is an evolution, a familiar idea re-emerging in a new technical or economic context, reshaped by what is now possible. The financial system is full of them, and most of the things people call disruption turn out, on inspection, to be derivations.

ACH was introduced in 1974. The first paper on Merkle trees was not published until 1979. Their use as cryptographic structures, and their role in distributed networks, was not even imagined. Fast-forward forty years and Merkle trees are the backbone of every blockchain, which are, in many ways, derivations of traditional settlement networks, just with different constraints, bandwidth, and trust assumptions. Different compute, same job.

Most of today's rails extend from the last generation's. Each technological paradigm of the last two centuries arrived through derivations of the last; the new paradigm does not replace the old one, it composes with it. Where the model gets useful is in distinguishing a real *grand* derivation, one that changes the surface area of the layer, from a thin coat of paint over the old thing. A grand derivation flips an axis: cost, latency, bandwidth, or trust. A thin derivation just renames it.

Three derivations, one substrate.

· **Protocols** are a derivation of the clearing network, a new TransferType, the substrate of the others.
· **Stablecoins** are branded balances on the protocol, a new ValueType.
· **Tokenized fiat** is a digital deposit token, kindred to the stablecoin.

The next three chapters take each in turn.

"Most of what looks like disruption, on inspection, is interpolation."

CHAPTER SEVEN

Stablecoins as *liquidity* derivations

Stablecoins are the new cash equivalents, accessible, digital, programmable. They are a derivation of an old role: the ValueType that sits next to operating cash, ready to move on demand, with the smallest possible cost of motion.

In the previous iteration of this role, ACH and same-day wire played the part. A treasury team kept a buffer at a bank, and ACH or wire was the off-ramp from one party to another. Stablecoins now serve the same role, but with three differences that matter at scale: they do not stop at the U.S. border, they do not respect banking hours, and the cost of motion is closer to the cost of a database write than the cost of a payment.

The result is liquidity that simply did not exist before. USDC and USDT are ValueTypes that function globally, instantly, in volumes the legacy cash-equivalent stack cannot match. The deepest permissionless liquidity pool on the planet, Bitcoin, is now programmatically reachable from a stablecoin contract on the same TransferType, which is a thing that has never been true before about any cash equivalent.

What changes for builders.

Float is no longer the engine. In the old world, the friction of moving money funded the system; in the new world, the friction is closer to zero, and the engine is volume and composition. Designs that depended on captive float, overnight balances, settlement delays, FX spread inside a flow, get repriced. Designs that compose stablecoin liquidity at the edges of an existing product get cheaper.

The honest summary: stablecoins did not invent the cash equivalent. They derived a new one whose physics, global, instant, always-on, near-free, make a class of products possible that previously couldn't pay for themselves.

CHAPTER EIGHT

Protocols as *transfer* derivations

Blockchains and the interop protocols that connect them, LayerZero, Axelar, CCTP, Chainlink CCIP, Solana itself, are not just payment rails. They are TransferTypes, in the strict sense of the model, but they are derivations of an unusually general kind: routers that can carry assets, states, or instructions for any purpose, not just value.

What they borrow from the legacy is the question set: routing, clearing, finality, reversibility, hours, cost. What they change is the answers. Routing is by address, not by routing number. Clearing and settlement collapse into the same event. Finality is probabilistic for the first few blocks and economic thereafter. Hours are 24/7. Reversibility is contract-defined, not policy-defined. The cost is per-byte and per-compute, not per-payment.

The distinction worth holding onto is that a protocol is not a single TransferType, it is a *factory* for them. Ethereum is one TransferType. So is Base. So is Arbitrum. So is the bridge between them. The fact that they share a virtual machine is a fact about engineering, not a fact about the model. From the layer's point of view, every distinct settlement domain is a distinct rail.

Why this matters at design time.

Treating each protocol as its own TransferType makes routing decisions explicit. *Should this transfer settle on Ethereum, on Base, or on Solana?* is not a religious question; it is a routing question with answers about cost, latency, liquidity, and counterparty trust. The same way an old payments engineer chose between ACH and wire by hour-of-day and amount, a new payments engineer chooses between L1s and L2s by liquidity and finality. Different compute, same job.

CHAPTER NINE

Legacy rails going *onchain*

Every fiat currency and bank rail already fits the model. The third grand derivation is not the arrival of a new ValueType or a new TransferType, it is the migration of the old ones onto the new TransferTypes. Tokenized fiat. Tokenized treasuries. Tokenized money-market funds. The legacy stack putting itself on protocols.

This is not a bridge in the metaphorical sense. The marketing language has always leaned on words like *bridge* and *on-ramp*, but those are metaphors of distance. What is happening here is closer than that. It is simply a move. When you move a file between folders on your computer, you do not "bridge" the file; you move it. There is a log somewhere that records the move, and that is how the computer knows where to show it. A tokenized dollar works the same way when it is well-designed: a value packet is picked up in one TransferType and placed in another, with the metadata that makes it legible to both sides of the move.

What you get when fiat goes onchain.

- Infinite bandwidth, in the sense that a chain's block space scales with software, not with banking hours.
- Instant settlement, with finality measured in seconds.
- Global reach without a correspondent bank in every jurisdiction.
- Programmability, every transfer can carry instructions.
- Composability with stablecoin liquidity already on the same rail.

None of this turns the legacy stack off. A tokenized dollar still sits on top of an FIS or Jack Henry core somewhere, and a stablecoin wallet can settle back into a bank account on those cores without any drama. *That is not disruption. That is interpolation.* Cores do not get turned off as the bandwidth of money goes up, only the ceiling moves.

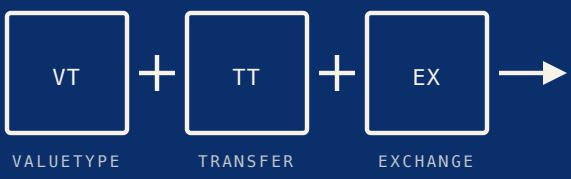
Derivations connect; they do not replace. The order is always the same: new ValueType, new TransferType, then legacy onto the rail.

PART THREE

Building

III

The model exists to make building easier. This part is about the operating constraints of money on the internet, the conventions that have proven useful, and a small set of rules for designing software that moves value.



a small grammar, a workbench

CHAPTER TEN

The *physics* of money

Money has physics, cost, latency, and bandwidth. Those three constraints define a *possibility space*: the set of tasks money can be made to perform. Change a constraint by enough orders of magnitude and the space changes shape.

In the last six months, those constraints effectively dissolved. The cost to launch a stablecoin has hit zero. The cost of an onchain transaction is now lower than the Federal Reserve master-account fee. Protocols have eliminated bank hours. Everyone, everywhere, can build on demand. None of these were true a year ago.

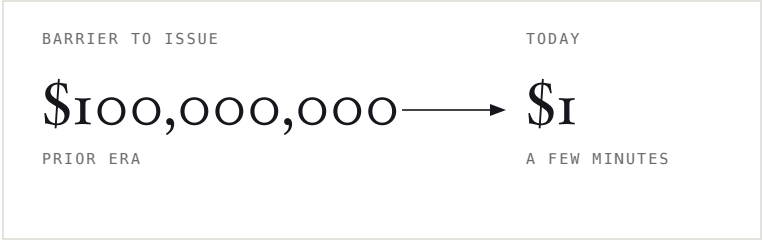


FIG. 2 · **A NINE-ORDER-OF-MAGNITUDE COLLAPSE** IN THE COST TO ISSUE A STABLECOIN.

$100m to $1.

When the barrier to creating a stablecoin was a hundred million dollars, it was simply too high for most companies. Now the barrier is roughly a dollar and a few minutes, and what used to be the privilege of the largest institutions, the ability to denominate, issue, and move value, is now an API call.

Most of the rest of this book is downstream of that single fact. When a primitive collapses by nine orders of magnitude, the right response is to map the new possibility space and build the things only that space allows.

> *The cost collapse is a counterfactual expansion (Marletto), paid for by a bubble that subsidized blockspace below Fed-rail cost. What was left behind is the rail.*
> SEE APPENDIX D · I & IV

CHAPTER ELEVEN

From verticals to *ecosystems*

Matt Harris once gave a talk about the verticalization of fintech. The argument: a great deal of high-value technology will be built and scaled across customer verticals, the same product tailored, vertical by vertical. Horizontal technology working everywhere it's pointed.

In the world of protocols the parallel almost fits, but a worthwhile difference shows up at the start of every conversation. The opening question is no longer "which vertical?" It is "which ecosystem?" Each L1, Ethereum, Solana, Base, Stellar, Canton, has a center of gravity that draws specific kinds of applications, liquidity, and developers. The vertical follows the ecosystem, rather than the other way around.

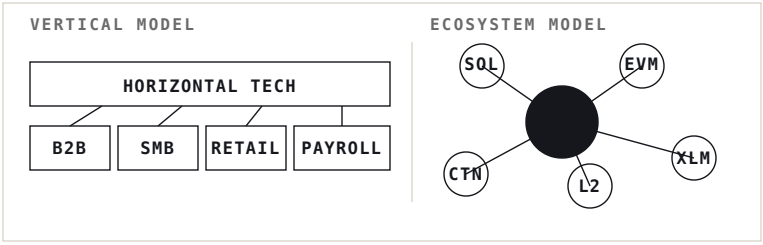


FIG. 3 · **VERTICALS FAN DOWNWARD; ECOSYSTEMS RADIATE OUTWARD.**

The most interesting opportunities are now cross-ecosystem and multi-chain. Breaking into a vertical can take a decade. In a healthy ecosystem, an integration that would have taken a year of business development can be running in a few hours. Welcoming-by-design is the new moat.

> *Wenger calls this dynamic a* knowledge loop*: the cost of an open standard approaches zero, so the value of building one compounds. Welcoming ecosystems are knowledge loops in financial form.*
>
> SEE APPENDIX D · II, WENGER

CHAPTER TWELVE

A *common* format for funds flows

Funds flows are a pain. Everyone draws them, everyone draws them differently, and the format keeps changing. As stablecoins are embedded into more products, I find myself drawing a funds flow once or twice a day. So we built a small standard at Brale called the *Commons Stablecoin Format* a set of rules to hand to a language model so it does not have to choose a format.

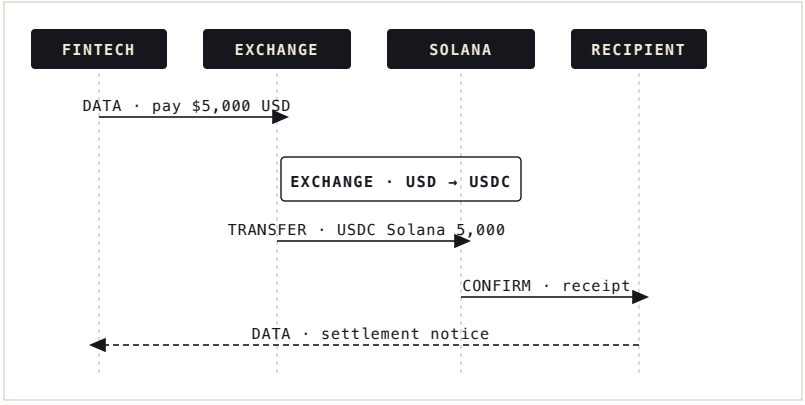


FIG. 4 · **CSF V1.4.5 (MEDIUM)** · USD WIRE → USDC ON SOLANA.

The grammar is small: *ValueType*, *TransferType*, *Amount*. SBC Base 100. USDC Ethereum 100. USD Wire 100. Like a circuit schematic, the diagram becomes legible the moment its labels are consistent. The standard is open source and has three depth levels, Light for sketches, Medium for compliance walk-throughs, Heavy for error and edge-case handling.

CHAPTER THIRTEEN

Twelve rules for *builders*

The rules below are the ones I have found most useful, collected from a decade of shipping, integrating, and debugging on the value layer. They are short on purpose.

1. **Lead with ValueType.** Geography belongs at the edge, on providers, not on the unit of value. Anchoring on country forces a rewrite at every border.
2. **Choose ValueTypes carefully.** The Issuer is inherent, USD names the U.S. government, USDC names Circle, SBC names Brale. Everything downstream is inherited. Pick units whose Issuer you trust.
3. **Treat each protocol as its own TransferType.** Ethereum, Base, the bridge between them, three rails. Pretending otherwise hides routing decisions.
4. **Make Exchanges explicit.** Every conversion has a price. If your funds flow does not show the Exchange, you do not know who is paying for it.
5. **Write flows as tuples.** *(ValueType, TransferType, Amount)* per leg, joined by Exchanges. Diagrams that match the grammar review themselves.
6. **Design for instant by default.** Build as if every TransferType is 24/7 with seconds-to-finality. The legacy ones will catch up.
7. **Compose with boring stablecoins.** Innovation belongs above the Issuer, not inside it. The boring ones do not break.
8. **Prefer welcoming ecosystems.** They are knowledge loops in financial form. A chain that ships docs, sample apps, and a chat where someone replies in an hour will compound on you.
9. **Plan for cores that don't turn off.** A stablecoin that cannot settle back into an FIS or Jack Henry core is not a strategy. Interpolation, not disruption.
10. **Use a standard funds-flow format.** A small grammar, CSF, or your own, pays for itself the first time you onboard a partner under a deadline.
11. **Treat regulation as part of the model.** The framework works well for fiat-backed, fully-reserved Issuers. Build inside it; don't pretend it is optional.
12. **Keep the grammar smaller than the catalog.** ValueTypes will grow without bound. Primitives should not.

CODA

The coming *decades*

If history is a guide, the next thirty years will do to money what the last thirty did to publishing and commerce. The pattern is already legible: Fedwire in 1918, the consumer internet in the 1990s, digital bank cores in the 1980s and 2000s, mobile in the 2000s, Bitcoin in 2009, stablecoins in 2014.

We are at, or just past, the turning point. The frenzy installed the rails, the chains, the wallets, the issuers, the protocols, the regulatory arguments. Most of that capital will not survive. What survives is the small grammar in Part I and the small set of boring institutions in Part II. The deployment phase, where productive infrastructure compounds for fifteen, thirty, forty years, is ahead. More likely than not, it will be many projects, many companies, many generational winners. Not one.

"In the same way HTTP let anyone publish a webpage, stablecoins and protocols are letting anyone publish to the value layer."

Global by default. Always on by default. Infinitely customizable for virtually free. Derivations matter less individually than in constellation. Together, they compound. That is why the future is not coming. It is already here. Our job is simply to build on it.

The value layer is no longer a metaphor. It is a workbench, with three primitives, a small grammar, and an open invitation. The atlas in Appendix A is the part of the catalog I have personally walked. The glossary in Appendix B is what each term means in this book. The API map in Appendix C is one working implementation of the model, ready to call.

The rest is up to you.

APPENDIX A

An atlas of *value*

A starting catalog. Each row is a ValueType, with its Issuer and common TransferTypes. The list is not exhaustive, it is a working sample, written to help readers extend the model into their own domain.

FIAT VALUETYPES

VALUETYPE	ISSUER	TRANSFERTYPES
USD	U.S. Treasury / Federal Reserve	ACH · Wire · FedNow · RTP · Zelle
EUR	European Central Bank	SEPA · SEPA Instant · TARGET2
GBP	Bank of England	Faster Payments · BACS · CHAPS
BRL	Banco Central do Brasil	PIX · TED
INR	Reserve Bank of India	UPI · IMPS · NEFT · RTGS
MXN	Banco de México	SPEI · CoDi
RON	National Bank of Romania	ReGIS · Plăți Instant · TransFonD SENT

STABLECOIN VALUETYPES

VALUETYPE	ISSUER	TRANSFERTYPES
USDC	Circle	Ethereum · Solana · Base · Arbitrum · Stellar · Polygon
USDT	Tether Limited	Ethereum · Tron · Solana · Avalanche
USDP	Paxos	Ethereum · Solana
PYUSD	Paxos (for PayPal)	Ethereum · Solana
SBC	Stable Coin Inc.	Base · Solana · Ethereum

APPENDIX A · CONTINUED

PROTOCOL-ISSUED VALUETYPES

VALUETYPE	ISSUER	TRANSFERTYPES
BTC	Bitcoin protocol	Bitcoin · Lightning · Wrapped (e.g. WBTC on EVM)
ETH	Ethereum protocol	Ethereum · Base · Arbitrum · Optimism
SOL	Solana protocol	Solana
XLM	Stellar Development Foundation	Stellar

TOKENIZED REAL-WORLD VALUETYPES

VALUETYPE	ISSUER	TRANSFERTYPES
BUIDL	BlackRock / Securitize	Ethereum · Base · Solana · Polygon
BENJI	Franklin Templeton	Stellar · Polygon
USYC	Hashnote	Ethereum · Canton

CLOSED-LOOP VALUETYPES

VALUETYPE	ISSUER	TRANSFERTYPES
Cash App $	Block, Inc.	Cash App internal ledger
PayPal $	PayPal	PayPal internal ledger
WeChat ¥	Tencent	WeChat Pay internal ledger

A few notes about the atlas. **Issuer** for protocol-issued ValueTypes is the protocol itself, there is no person, but the rules are explicit, and that is enough. **TransferTypes** for stablecoins are the chains the Issuer has minted on; balances on different chains are the same ValueType on different rails. **Closed-loop ValueTypes** are their own TransferType, money inside Cash App moves only on Cash App. The model handles all of these uniformly.

APPENDIX A · CONTINUED

TransferTypes, *offchain*

Four worked examples. Each pairs a different base currency, and therefore a different Issuer, with the rail that natively settles it. The point is to show that the model speaks the same grammar across jurisdictions, even when the institutions look nothing alike.

VALUETYPE	TRANSFERTYPE	ISSUER	NOTES
USD	ACH	U.S. Federal Reserve / Treasury	M–F; T+1 batch. The default for U.S. payroll, B2B, and recurring debits. Low cost, high reach.
EUR	SEPA Instant	European Central Bank	24/7; instant. Pan-Eurozone retail and business transfers, settled in seconds.
GBP	Faster Payments	Bank of England	24/7; near-instant. UK domestic peer-to-peer, payroll, bill pay.
BRL	Pix	Banco Central do Brasil	24/7; instant. Brazil universal: free for individuals, used for everything from payroll to street commerce.

OTHER OFFCHAIN TRANSFERTYPES WORTH KNOWING

U.S.: Fedwire, FedNow, RTP, Zelle. **Europe:** SEPA, TARGET2, BACS, CHAPS. **Latin America:** SPEI (Mexico), TED (Brazil). **Asia:** UPI (India), PromptPay (Thailand). **Card networks:** Visa, Mastercard, Amex, Discover. **Closed loops:** Cash App, Venmo, PayPal, WeChat Pay. Each carries its own ValueType identity through the same primitive grammar.

APPENDIX A · CONTINUED

TransferTypes, *onchain*

Four worked examples. Each pairs a different stablecoin Issuer with the chain it natively settles on. The four chains and the four Issuers are deliberately distinct, to make the point that the model is neutral to which chain a unit lives on and to who issues it.

VALUETYPE	TRANSFERTYPE	ISSUER	NOTES
USDC	`Ethereum`	Circle	24/7; ~12 s blocks, ~12 min finality. The deepest stablecoin liquidity pool on the most widely supported EVM chain.
USDT	`Tron`	Tether Limited	24/7; ~3 s blocks. Globally the highest-volume stablecoin transfer pair, especially in cross-border remittance corridors.
PYUSD	`Solana`	Paxos (for PayPal)	24/7; sub-second finality. PayPal's stablecoin natively bridged to a high-TPS chain for consumer-facing payments.
SBC	`Base`	Brale Inc. (Stable Coin Inc.)	24/7; ~2 s blocks. A community-issued USD stablecoin on Coinbase's L2, designed for permissionless settlement at near-zero cost.

OTHER ONCHAIN TRANSFERTYPES WORTH KNOWING

L1: Bitcoin, Stellar, Avalanche. **L2 (Ethereum):** Arbitrum, Optimism, Polygon PoS. **Off-chain on Bitcoin:** Lightning. **Interop / cross-chain:** CCTP (Circle), LayerZero, Axelar, Chainlink CCIP. The same four-row pattern applies: a ValueType, the chain it lives on, the entity that issues it, and the operating notes that distinguish it from neighbors.

APPENDIX B

A glossary of *primitives*

VALUETYPE
A unit of value with an identity that survives transfer. May be a currency (USD), a stablecoin (USDC), a protocol asset (BTC), a tokenized share, or an NFT. The model is neutral to all of them. Each ValueType inherently names its *Issuer* the entity, government, or protocol whose obligation gives the unit its meaning. USD names the U.S. government; USDC names Circle; SBC names Brale Inc. The Issuer does not appear separately in the grammar.

TRANSFERTYPE
A network or rail that knows how to move a ValueType from one place to another and produce a receipt. ACH, wires, SEPA, FedNow, Ethereum, Solana, Lightning, card networks, and closed loops are all TransferTypes.

EXCHANGE
A conversion from one (ValueType, TransferType) pair to another, with a price. The only verb among the primitives. Where economics, fee, spread, slippage, capital, enters the model.

FUNDS FLOW
A sequence of legs joined by Exchanges. Every payment system in production reduces to a funds flow, and the Commons Stablecoin Format is one open standard for writing them down.

DERIVATION
An evolution of a familiar primitive in a new technical or economic context. Stablecoins are a derivation of cash equivalents; protocols are a derivation of clearing networks; tokenized fiat is a derivation of bank rails.

INTERPOLATION
What this book calls the connection between legacy and new, neither bridge nor disruption. A stablecoin wallet settling back into a bank account on an FIS core is interpolation: the systems remain, the bandwidth between them grows.

BORING STABLECOIN
A fully reserved, regulated, fiat-backed stablecoin whose Issuer is dull on purpose. Belongs underneath the interesting things; does not break.

APPENDIX C

The Brale API, mapped to the *model*

The model in this book is not theoretical. The Brale API is one working implementation, designed around the same three primitives. The table below maps each primitive onto the endpoints you will actually call. Full documentation is at DOCS.BRALE.XYZ.

PRIMITIVE	ENDPOINT	WHAT IT DOES
(Issuer config)	`/accounts`	Identity and configuration of the regulated entity issuing the ValueType. Compliance, reserves, and KYB live here. Not a primitive in the grammar, the Issuer is inherent in the ValueType the account provisions.
	`/addresses`	The custodial and external addresses through which the account can send and receive value. *internal* addresses are custodial; *external* are user-controlled.
ValueType	`/values` `/balances`	Enumerates the ValueTypes the account is provisioned for, and returns balances per (ValueType, TransferType) pair.
TransferType	`/transfer-types`	The set of rails the account can route over, including bank rails (ACH, wire) and chains (Ethereum, Solana, Base, Stellar, others).
Exchange	`/transfers`	The verb. Submit a transfer with input (ValueType, TransferType, Amount) and output (ValueType, TransferType). Conversion is performed by the platform; receipts are returned.
	`/data/prices`	Pricing reference, the rates applied to Exchanges, surfaced for inspection and reconciliation.
	`auth.brale.xyz`	OAuth client_id / client_secret authentication. Bearer tokens are scoped per account.

A useful exercise: walk any CSF funds flow through this table. Every leg is a /TRANSFERS call; every pre-flight is /BALANCES or /DATA/PRICES. Model and implementation speak the same grammar.

APPENDIX D · FOUR READINGS

Reading the value layer through *Marletto*

I · CONSTRUCTOR THEORY & COUNTERFACTUALS

A note on Chiara Marletto, The Science of Can and Can't: A Physicist's Journey through the Land of Counterfactuals (Allen Lane, 2021).

Marletto's book reformulates physics in terms of transformations, what is possible, what is impossible, and what substrate-independent rules govern the difference. The deepest claim of her program is that physics is incomplete unless it accounts for *counterfactuals*: not only what happens, but what could have happened.

The value layer fits this frame almost perfectly. Each of the three primitives is a constraint on what transformations are possible. A ValueType is a unit that can be picked up and put down somewhere else, a possibility claim, with the obligations of its Issuer baked into its identity. A TransferType defines the set of physical and informational transformations a network can perform on that unit. An Exchange is itself a transformation between (ValueType, TransferType) pairs.

Read this way, the chapter on the *physics of money* is doing what Marletto's program calls for. The collapse from $100M to $1 is not a price change. It is a change in the set of possible tasks, a counterfactual expansion. Tasks that were impossible (issue a stablecoin in 2018 with $1M of capital) became possible (issue one in 2026 with $1). The constructor specification of money widened.

The implication is stronger than "things are cheaper." It is the claim that the space of value-bearing computations a system can perform is now strictly larger than it was. The three primitives are not modeling conveniences. They are the substrate-independent description of value as a thing that can be transformed. A working API for the value layer is, on Marletto's account, a working physics of money in miniature.

APPENDIX D · FOUR READINGS

Reading the value layer through *Wenger*

II · THE WORLD AFTER CAPITAL

A note on Albert Wenger, The World After Capital (worldaftercapital.org, 2017–ongoing).

In *The World After Capital*, Wenger argues that the binding constraint on human activity has shifted. In the Industrial Age it was capital. In the Knowledge Age it is attention. The reason is that digital technology is universal and zero marginal cost, software can do anything other software can do, copied for free.

Money is one of the last categories to fully cross this threshold. For most of the last century the cost of moving and issuing money was nontrivial; that cost shaped institutions and behaviors. The argument of this book is that the cost has collapsed and that the value layer is now Turing-complete in the relevant sense, programmable, universal, near-zero-marginal.

Wenger's framework predicts three corollaries, and all three are visible in the chapters above. First, the binding constraint moves from the cost of moving money to the cost of *deciding* what to do with it, an attention problem more than a capital problem. Second, knowledge loops emerge: the cost of an open standard like the Commons Stablecoin Format approaches zero, so the value of producing one compounds. Third, the right strategic response is to design for informational and psychological freedom, not new capital moats.

The Twelve Rules read like an applied tract of the Knowledge Age. *Compose with boring stablecoins* is a knowledge-loop argument: build on the parts that are open and stable so others can build on you. *Prefer welcoming ecosystems* is the same: a chain whose docs and chat exist is a chain that compounds, because compounding is the law of the Knowledge Age. The value layer is the financial half of the world after capital.

APPENDIX D · FOUR READINGS

Reading the value layer through *Perez*

III · TECHNO-ECONOMIC PARADIGMS

A note on Carlota Perez, Technological Revolutions and Financial Capital: The Dynamics of Bubbles and Golden Ages (Edward Elgar, 2002).

Perez offers a remarkably stable framework for reading this kind of moment. Each of the five technological revolutions of the last 240 years follows the same arc: an *installation phase* (irruption, then frenzy), a turning point, and a *deployment phase* (synergy, then maturity). Bubbles do not destroy the paradigm; they install its infrastructure.

The value layer, as described in this book, is the infrastructural deposit of the current installation phase. The capital flowing through crypto since 2017 has done what frenzies do, funded the chains, the wallets, the issuers, the protocols, the regulatory arguments. Most of it will not survive. What survives is the small grammar described in Part I and the small set of boring institutions described in Part II.

Perez would predict, on this reading, that we are near or just past the turning point of the value layer. Bubbles are still happening; the ratio of speculative motion to productive motion is still very high. But the deployment-phase infrastructure is now visible, boring stablecoins, regulated Issuers, fully-reserved fiat-backed instruments, working APIs. Those are the pieces that compound through the golden age.

Two implications, in Perez's vocabulary. First, the right strategic posture today is to invest in the parts of the stack that survive the turning point. Not the parts that look most exciting in the frenzy. Second, the deployment phase is longer than installation by a factor of two or three. The opportunity to build inside the value layer extends, on Perez's clock, somewhere between fifteen and forty years. The book closes with "the rest is up to you." Perez would amend that to: *the rest is up to the deployment phase*.

APPENDIX D · FOUR READINGS

Reading the value layer through *Hobart & Huber*

IV · BUBBLES INSTALL INFRASTRUCTURE

A note on Byrne Hobart and Tobias Huber, Boom: Bubbles and the End of Stagnation (Stripe Press, 2024).

Hobart and Huber argue that bubbles are not always pathologies. The Manhattan Project, Apollo, Moore's law, and Bitcoin share a common shape: small groups with unified vision, vast funding, and surprisingly poor accountability. The combination is financially unstable, but precisely because of that, it produces infrastructure ordinary capital never would. "Optimism," they say, "can be a self-fulfilling prophecy."

The crypto bubble is exactly this shape. Tens of billions of dollars of VC and retail capital have been poured into chains, wallets, issuers, and protocols since 2017. The visible top of that funnel was a casino: Bored Apes, leveraged tokens, governance theater. Most of that capital will not see a productive return. But the casino was not the point. The point is what was *left behind*: the chains, the wallets, the issuers, the standards, the developer tooling. That is the productive residue of the bubble, and that is what accelerates the future.

The chapter on the *physics of money* is the consequence. The constraints that defined the possibility space of money for a century, cost, latency, bandwidth, collapsed because Hobart and Huber's machinery worked. The casino subsidized blockspace below the cost of a Fed master-account fee and below a relational-database row-write at scale. That is a bubble subsidizing the infrastructure of the next paradigm. The bandwidth of money is now effectively infinite, downstream of the bubble.

Complementary to Perez: where Perez describes the *shape* of installation, Hobart and Huber describe the *fuel*. Builders today are using rails paid for by the last decade's froth. The honest move is to use them.

APPENDIX E

Bibliography

Companion readings

Hobart, Byrne, and Tobias Huber. *Boom: Bubbles and the End of Stagnation.* Stripe Press, 2024. STRIPE.PRESS/BOOM.

Marletto, Chiara. *The Science of Can and Can't: A Physicist's Journey through the Land of Counterfactuals.* Allen Lane / Viking, 2021.

Perez, Carlota. *Technological Revolutions and Financial Capital: The Dynamics of Bubbles and Golden Ages.* Edward Elgar, 2002.

Wenger, Albert. *The World After Capital.* Self-published, 2017–ongoing. WORLDAFTERCAPITAL.ORG.

Source essays (benmilne.com)

Milne, Ben. "The Value Layer." 31 July 2021.

Milne, Ben. "The Value Layer, Expanded." 9 December 2022.

Milne, Ben. "Verticals to Ecosystems." 5 September 2022.

Milne, Ben. "Payment Regulations and the Evolving Value Stack." 24 September 2022.

Milne, Ben. "Better Funds Flows." 13 April 2025.

Milne, Ben. "Value Derivations." 28 September 2025.

Milne, Ben. "The Physics of Money." 6 February 2026.

Milne, Ben. "$100m to $1." 8 February 2026.

Milne, Ben. "Rate of Change." 19 April 2026.

Working implementations

Brale. *Stablecoin API.* DOCS.BRALE.XYZ.

Commons Stablecoin Format. GITHUB.COM/SUPERDUPERDOT.

ACKNOWLEDGMENTS

Acknowledgments

The model in this book is not mine alone. It was sharpened, over many years, by people who challenged it, used it, broke it, and improved it.

Matt Harris's talk on the verticalization of fintech is one of the few pieces of analysis I keep returning to a decade after first hearing it. Robert Bench made me think differently about throughput because of his work with OpenCBDC. Matt Homer pushed my thinking on the regulatory overlay. Alka Gupta and Eric Saraniecki gave generous feedback on the primitives at a stage when they were not yet ready to be primitives. Auren Hoffman is the reason I started writing the chicken-scratch notes that turned into the original Value Layer essay.

The ecosystems behind every blockchain referenced in these pages, Stellar, Solana, Base, Ethereum, and others, gave us room to experiment without knowing how it would play out. Stellar in particular supported the early work that became this model.

The Brale team built the API that proved the primitives compose. Jeff Milewski and the team behind Stable Coin Inc. carried forward the work that started as SBC inside Brale.

And to the people who kept asking the question, *what do you actually mean by the value layer?* thank you. This book is the answer I have been trying to give for a decade.

B.P.M.

The Value Layer

An independent edition

By Ben Milne
benmilne.com

EDITION ONE · 2026