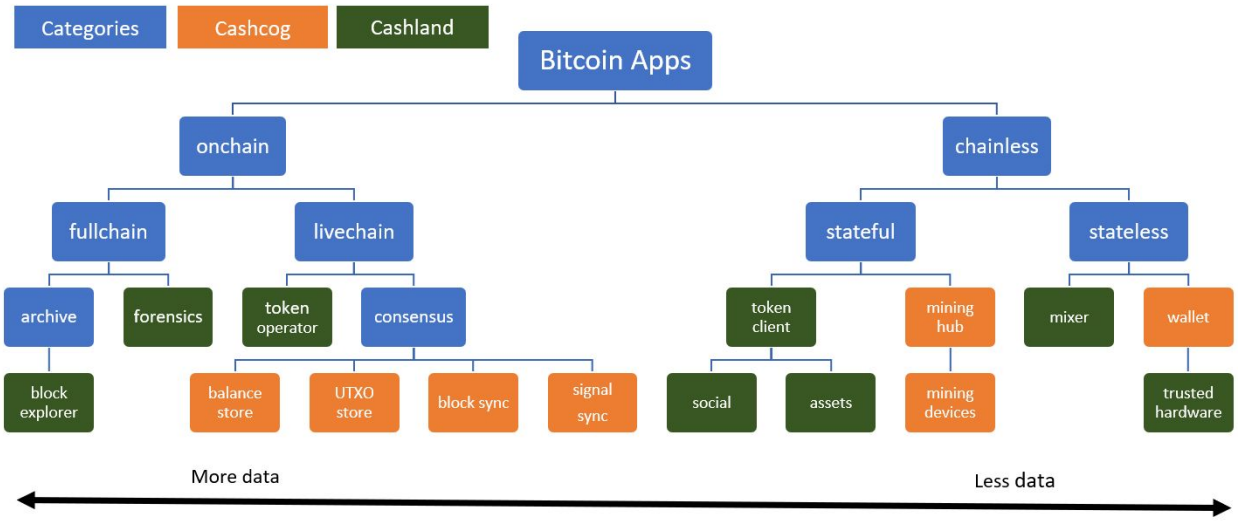
A taxonomy of the Bitcoin applicative landscape

By Joannes Vermorel, CEO of Lokad, May 7th, 2018

Bitcoin has been plagued for years by intermittent sterile discussions where most participants seem to be talking past each other. While bad faith is rampant among certain crypto-circles, partially because some participants have financial incentives not to listen to certain objections[1], the terminology around Bitcoin itself is not without problem either. The most abused word is probably the term *node*; with the term *miner* trailing not too far behind. Even between participants acting on good faith, it is difficult to properly articulate technical discussions about Bitcoin, especially when concerns like *censorship resistance* are expressed, as those concerns are themselves frequently loosely defined as well.

The taxonomy that is proposed in the following adopts a *functional* perspective - as in *functional specification* not functional programming - where parts of the Bitcoin landscape are isolated depending on the primary function they serve. This taxonomy is refined by a *scaling* perspective that emphasizes how much data is involved for every part. Indeed, in Bitcoin, elements of interest can be as small as a few tens of bytes - secrets - and as large as petabytes - blockchain at scale.



Higher resolution at http://media.lokad.com/bitcoin/taxonomy-schema-2018-05-07.pdf

[1] When your business delivers a solution to a specific problem, the survival of your business depends on the survival of the problem itself. Thus, any simple solution to such a problem is a danger to the business, which presses people in charge of said business to promptly act against said simple solution.

Reviewing the problematic terms

Before delving into the fine print of the proposed taxonomy, let's review some terms that are commonly used in discussions about Bitcoin, and clarify why those terms frequently prove themselves to be misleading rather than helpful.

Bitcoin node: The term *node* is generally understood from a *distributed computing* perspective, with a *node* being a computer running a specific piece of software while being connected to a network. However, in the Bitcoin context this term is misleading at multiple levels. First, as we will see in the following, Bitcoin is made of many parts, and thus a single Bitcoin participant might have many specialized "nodes" to support her own Bitcoin activity. Second, some parts of Bitcoin require horizontal scaling, and thus many machines, which turn out to be as many "nodes". Finally, from a functional perspective, the desired outcome associated to the ownership of a node heavily depends on the context, thus, the intent that should characterize the *node* remains vague at best.

Mining node: Mining is the randomized work that offers the possibility to emit *proofs* (of work) in order to claim block rewards. Mining is carried by specialized hardware devices and requires massive horizontal scaling to be of any interest, i.e. thousands of hardware units need to be involved to make any non-trivial contribution to the Bitcoin security. Yet, at the same time, those units require little networking resources. This property goes against the intuition typically associated with the terminology of *node* in distributed computing where a node is a busy participant in the state of affairs of the network. Also, the term *miner* is frequently used to refer to participants who construct and propagate new blocks, which is misleading as this work entails challenges that are orthogonal to the hashrate competition.

Mining pool: The primary economic function of the mining pool is to federate the collective hashrate of many market participants in order to lower the variance of the payouts associated to the emission of new blocks. However, this activity bundles two distinct challenges, fostering further confusion about the nature of *miners*. First, there is an IoT challenge (Internet of Things) which consists of federating a large number of mining devices[2]. Second, there is a block synchronization challenge, which requires strong networking ties to peers[3], as well as solving a couple of related challenges such as establishing a working transaction fee market, and securing microlatent transactions.

Bitcoin client (or Satoshi client): The *client* is a terminology originating from the client-server model, which is unfortunately in opposition to the macro design of Bitcoin, where *peers* operate.

[2] While this is speculative at the time of writing, mining pools may provide in the future the necessary instrumentation to turn on and off the mining devices depending on market conditions which include a spot price for kilowatts.

[3] Those peers would usually be referred as *miners* however, as previously pointed out, this terminology is vague. Thus, for now, let's keep it as *block synchronization peers*.

Moreover, the original implementation by Satoshi Nakamoto - Satoshi's codebase - was an heteroclite compilation of software capabilities bundled in a monolithic software implementation. This perspective is giving a false sense of unity to parts that should be kept distinct. This issue is even more confusing when computing requirements differ wildly from one part to the next.

Layer 1 (and Layer 2): As Satoshi's client was a monolithic implementation, a bizarre layer 1 vs layer 2 terminology emerged to distinguish whether code would run from Satoshi's codebase, which would be deemed as Layer 1, or would run elsewhere, which would be deemed as Layer 2. This perspective wasn't overly sensical in the first place, as the Satoshi's codebase was too heteroclite to be the foundation of a Bitcoin taxonomy. Later on, the confusion increased as those two layers were increasingly understood as Bitcoin 1.0 (Layer 1) against Bitcoin 2.0 (Layer 2), where the second layer was used to solve a supposedly unsolvable problem within Bitcoin 1.0.

SPV wallet (simplified payment verification, or thin wallet): As Satoshi's codebase bundles heteroclite concerns, the terminology *SPV wallet* emerged to emphasize a wallet that did not happen to be entangled with Satoshi's codebase. However, this terminology was unfortunate because it gives the incorrect impression that there should be a "regular" wallet to be opposed to a SPV wallet - the later being a degraded version of the former. Yet, as detailed in the following, the opposite is true: all "proper" wallets are SPV wallets, leaving the user free to decide which *balance store* is to be trusted. Much of the sterile debate around the hardware requirements of a "proper" wallet emerged by conflating orthogonal concerns under a loosely defined notion of wallet.

The functional perspective

The taxonomy being presented in this document is *functional*: every part is distinct because it answers a different *why* question. Thus, the purpose is the defining trait of every part of Bitcoin. This approach is superior to the observational perspective in that the latter merely reflects the *what*, that is, observing Bitcoin as presently implemented. Indeed, if Bitcoin ever manages to scale, its underlying software will undergo profound transformations, precisely intended to allow the scaling to happen. Thus, the *what* is likely to be in constant flux, possibly for a decade or two. In contrast, the *why* has a decent chance to stay largely unchanged for a long period of time anchored in the ambition of Bitcoin itself of being the highest form of cash.

The data fragments of interest in Bitcoin range from 32 bytes ($\approx 10^2$), the size of a secret, to 1 exabytes (10^{18}), the size of a world-encompassing blockchain. Similarly, on the computing side, the computations of interest range from 10^5 BOPS - Basic Operations Per Second[4] - for a simple signing device, to probably about 10^{20} BOPS - for the global hashrate of a world-dominating Bitcoin. In the two cases, we have at least 15 orders of magnitude to account

[4] BOPS, Not FLOPS! A New Metric and Roofline Performance Model For Datacenter Computing, Lei Wang, Jianfeng Zhan, Wanling Gao, ZiHan Jiang, Rui Ren, Xiwen He, Chunjie Luo, Gang Lu, Jingwei Li, May 2018, https://arxiv.org/abs/1801.09212

for. In comparison, if 1 meter is supposed to be the distance travelled by a human at walk speed in 1 second, 10^{15} meters is roughly the distance travelled by light - at lightspeed - in a month. Within Bitcoin, the divergence between the *micro* and the *macro* is so massive that parts should be classified based on their scaling requirements.

Finally, as Bitcoin is both a computer network as well a social network, all those parts interact, and no part makes sense in isolation. Thus, any hierarchical classification is doomed in some sense, as it cannot capture those interactions. Yet, without prior intellectual preparation, tackling Bitcoin from the perspective of a complex interaction graph is unlikely to yield any insight, and more likely to yield a lot of confusion. Once the purpose of the parts is clearly understood, most of the interactions, which are left implicit, can be inferred.

Scale-dependent categories

The taxonomy attempts at classifying *apps*[5] that supports Bitcoin. The categories are defined on purpose around the *data scalability* burden associated with every single app. Indeed, the amount of data involved to deliver a specific feature has profound implication on the design of the software but also on the shape of the Bitcoin ecosystem that can be expected to emerge.

Onchain apps: The app has to process a sizeable fraction of the blockchain, independently of any specific usage of the app itself. Thus, those apps are fundamentally big data apps, carefully engineered for unlimited horizontal scaling. The scalability challenges are centered around preserving the locality of reference - to make the most of non-uniform memory accesses - and mitigating long term data storage costs.

Fullchain apps: A type of *onchain app* that preserves all the blockchain, including all the parts that are fully pruneable from both the cash and token perspective. As neither cash nor token require all this data to be persisted, actors that invest in the long term preservation of such a massive amount of data are able to generate revenues from second-order use cases such as forensics or similar attempts at de-anonymizing Bitcoin addresses.

Livechain apps: A type of *onchain app* that only has to deal with the UTXO set, or possibly the UTX set[6]. Those apps deal with the non-pruned fraction of the blockchain, which is expected to be two or three orders of magnitude smaller than the full blockchain. The mechanics of Bitcoin as cash, or as a token infrastructure, are funding the costs associated with those apps.

Consensus apps: A type of *livechain app* that ensures that the Nakamoto consensus is executed correctly, and that the resulting fine-grained data that emerge from the consensus are

[5] The term *service* as in *Service Oriented Architecture* (SOA) could be used here instead of *app*. However, the term *service* is so generally overloaded with varying semantics than I find this alternative even more troublesome than the term *app*.
[6] The UTX set is a superset of the UTXO set which is of interest for tokens. See *Tokeda, Viable token-driven metadata within Bitcoin*, Joannes Vermorel, May 2018

exploitable at large. This category can be understood as the essence of the peer-to-peer infrastructure backing Bitcoin. However, as we will see in more details, this category is far from being a monolith.

Chainless apps: Those apps do not have to process and persist any more than fragments of the blockchain. Those fragments can be small or large depending on the usage made of those apps by the end-users themselves. The scalability requirements are fundamentally driven by the usage of the app itself, not by the growth of the blockchain. Big data requirements arise naturally for multi-tenant apps that happen to have a lot of users (tenants) to support.

Stateful apps: A type of *chainless app* that maintains a non-trivial state outside of the blockchain. This state can be maintained outside of the blockchain for many reasons such as costs - blockchain storage is expensive - or requirements - blockchain storage cannot be made mutable - or simply out of practicality, as relational databases are more appropriate than the blockchain for most use cases.

Stateless apps: A type of *chainless app* that maintains only a trivial state, possibly immutable after its initialization such as a 32-byte secret. Entire classes of attacks are avoided by eliminating state transitions from the app. Thus, stateless apps are of prime interest whenever secure computing is sought to be delivered to the end-user in the Bitcoin ecosystem.

Cashcog apps

The prime differentiator between apps within the Bitcoin ecosystem boils down to whether they are needed to *keep Bitcoin working as cash*. In the taxonomy, a series of apps is identified as *cashcog* because removing any one of those apps from the applicative landscape denatures Bitcoin to the point it's not Bitcoin anymore. Conversely, apps that are delivering capabilities powered by *Bitcoin working as cash* are referred to as being part of *cashland*. While those apps can be highly desirable, they are not a strict requirement to keep the cash flowing.

Within *cashcog*, the community should expect that permissionless options exist for all apps involved. In practice, permissionless means that the source code is open source (although not exclusively, see below), and that the hardware involved to run the software is widely accessible to the point where it can be treated like a commodity.

Being permissionless is the key to achieving *decentralization* which should not be understood as *all market participants are equal* but understood as *no market participant can get a definitive upper hand*. It is unreasonable to expect a market - Bitcoin or otherwise - to conjure an equilibrium that does not involve hierarchies between actors, some being more competent than others, and rewarded as such by the market. As a practical consequence for Bitcoin, it is an unreasonable burden to put on *cashcog* apps to expect all participants to be undifferentiated.

Then, having access to an open (source) option does not preclude having access to proprietary options as well, and possibly a mix of both. The purpose of the open option is to maintain a *permissionless* system. To deliver this, the open option does not need to be the most cost-efficient option offered by the market, it only needs to be a reasonable one. Ideally, *open hardware* will gradually be developed as well to consolidate this angle further. However, as the implications ramify well beyond Bitcoin, those options are likely to emerge quite independently from Bitcoin.

Finally, the economic resources involved to support *cashcog* apps should not be expected to be bounded by anything but the market forces themselves. This proposition goes in both directions: the cost to run an app has no upper or lower bound outside the market context. For example, there is no absolute upper bound to the amount of resources to be invested in *mining devices* - which deliver the hashrate. Only the market can tell when the hashrate is high enough. Similarly, there is no absolute upper bound for the size of the blockchain. The market itself[7] will tell when the blockchain is *large enough*. Conversely, for the wallet app, while low-cost options are already available - a $20 second hand smartphone is enough to support a wallet - the market will decide whether there is a need for disposable $0.1 wallets delivered as RFID chips.

As a corollary of the *cashcog* perspective, let's point out two capabilities that cannot be served by anything but cashcog apps:

- **Scalability**: if an app happens to be required for Bitcoin to operate at scale, per original design intent of Bitcoin being the *cash of the world,* then this app is cashcog.
- **Microlatent security**: similarly, if securing a Bitcoin transaction within milliseconds requires a specific app, then this app is cashcog as well.

As a consequence of the discussion above, at least one option should exist to make those capabilities accessible *without permission*. This does not preclude proprietary solutions to supplement them, however the short history of Bitcoin has already demonstrated several times that *hybridization*[8] was an angle to attempt introducing permissions in disguise[9].

Cashland apps

The cashland apps supplement Bitcoin and provide ties with the non-Bitcoin applicative ecosystem at large. The primary benefit of having an app part of cashland rather than cashcog

[7] Bitcoin is still facing a short series of well-identified issues that prevent a working fee market to emerge for the size of the blockchain. Solutions are known but not immediately accessible. See *Midas, united non-colluding transaction fees for Bitcoin,* Joannes Vermorel, April 2018

[8] Hybridization refers here to a technical solution intended for Bitcoin which introduces an entire applicative landscape of its own, equivalent to one of Bitcoin itself. The prime danger with this approach is that, while Bitcoin is relatively known and well-understood, the situation is a lot muddier for the hybrid counterpart.

[9] Merely releasing software as open source, albeit being a positive contribution to the ecosystem at large, is nowhere enough to ensure that options remain *permissionless* in the context of Bitcoin. A Bizantine actor can even use open source to catalyze its capture of an otherwise permissionless market.

is practicality: within cashland, apps are regular software, and remain unburdened by the rather specific requirements of cashcog. Depending on the situation, a cashland app may require specific efforts on security or scalability, however those efforts are context-dependent and not some existential requirements.

Let's point out that the boundary between cashcog and cashland is not, at the time of writing, fully finalized, although there is little doubt about the outcome of this process when looking 5 years ahead. Below, a few apps that represent *fringe* cases within cashland which, depending on the philosophical interpretation of Bitcoin, be interpreted as cashcog.

Blockchain archive: Bitcoin has been designed to allow its blockchain to be pruned as only the UTXO set is needed to keep the cash flowing. Thus, UTXO commitments will be introduced in one form or another, removing the need to persist the blockchain in full. As a result, serving all blocks from the genesis block on should be considered as cashland. While non-paying options might remain available in the future for Bitcoin hobbyists, the lasting availability of those options is not a requirement for the survival of the cash use case.

Token operator (for assets): There is a whole range of uses[10] for *tokens* as enabled by Bitcoin. From the cashcog perspective, no matter how valuable are the *assets* operated through tokens on the blockchain, those tokens are not required to get the cash working. Furthermore, as the monetary value of Bitcoin itself is nothing but trust-but-verify recursively applied upon itself, there is little economic gain to justify having tokens part of cashcog, especially considering the sheer diversity of the use cases, which reflect the diversity of the world outside of Bitcoin.

Mixer (or coin shuffling): The flow of *bitcoins* in not opaque which makes it possible to external observers of the blockchain to deanonymize Bitcoin addresses, and the UTXO set in general. A mixer delivers opacity by constructing complex transactions where it's not possible to identify which input matches which output. As mixers are not a requirement to get the *cash* flowing[11], while being a highly desirable option, those apps are cashland.

A review of cashog apps

As cashcog is defined as the set of parts of Bitcoin which are required to keep the cash flowing, it only involves a rather limited set of apps, which is rather convenient if one intends to build a taxonomy, which is precisely the point of the present document. Let's review the apps part of cashcog.

[10] Bitcoin offers practical ways to propagate a publicly provable trust-but-verify security model outside of its own monetary system. See *Tokeda, Viable token-driven metadata within Bitcoin, Joannes Vermorel, April 2018*

[11] At the time of writing, a large majority Bitcoin users have most likely never used a mixer.

We have 4 apps whose collective purpose is to establish the fine print of *what* is being agreed upon as an outcome of the Nakamoto consensus. Thus, those apps are referred to as the *consensus* apps.

Balance store: This app is probably best understood as the *backend* of a wallet. Considering that the purpose of a wallet is to propagate proofs of ownership of a secret without disclosing the secret, the balance store can be queried to (1) establish how much funds are under the control of the end-user and (2) retrieve the proper list of UTXO entries needed to construct a Bitcoin transaction, both being required to move *bitcoins* around. Balance stores compete to win market shares among *wallets* (see below). The key technical challenge is to lower the latency for better user experience, at a low cost, while serving many end-users.

UTXO store: This app, in its most basic form, is dedicated to the validation of blocks. A Bitcoin participant can use the UTXO store to perform the economic validation of transaction, i.e. checking that the total number of *bitcoins* in the inputs of the transactions is equal or greater than the total number of *bitcoins* in the outputs. This store provides the capability to check the status of any entry in the Bitcoin ledger. UTXO stores compete to win market shares while collaborating with *mining hubs*. The key technical challenge is to provide a high reliability minimizing data storage costs as well as minimizing the latency involved with the synchronization of a new block.

Block sync: This app is dedicated to the synchronization with the longest chain of blocks which gets extended every 10 minutes on average. Blocks are emitted infrequently, but their content is almost entirely known in advance, as the next block is mostly contain the transactions that have been successfully propagated among the participants who have the capability to emit a new block. Block sync apps compete to win market shares among block emitters. The key technical challenge is to keep the networking requirements as low as possible at block-propagation time, and also to make the most of a non-uniform network, where peers are proportionally favored based on their capacity to propagate new blocks.

Signal sync: This app refers to the pre-consensus signaling[12] mechanism that establishes the status of a transaction before it gets included in a block. Pre-consensus signaling answers two questions: (1) what is the appropriate market-driven fee for a given transaction (2) what is the degree of trust to be given to a transaction not yet included in a block. Signal sync apps compete to win market shares among payment processors or token operators who seek to operate with arbitrarily low latencies. The key technical challenge is to geographically distribute the app itself in fashion similar to what is already being done for CDNs (content delivery networks) to deliver market-driven microlatent transactions.

[12] The concept of pre-consensus signaling is taken from *Ansible, practical faster-than-light secure 0-conf transactions for Bitcoin,* Joannes Vermorel, April 2018.

Mining hub: This app federates a large number of mining devices (see below). The purpose is to keep devices working on the longest chain of blocks almost all the time. As there is little gain in having any capability - e.g. smart power management[13] - backed into the mining devices themselves, those responsibilities should naturally move to the mining hub itself. Cloud computing platforms which support IoT hubs are already illustrating this pattern: everything that does not have to be done in the IoT device is preferably done in the cloud, as the cloud option requires less engineering and is easier to maintain. Mining hubs compete to win market shares among operators of mining devices. The key technical challenge is to maximize the economic value of the mining devices, while minimizing the operating costs.

Mining devices: Unlike all other entries in this document, it's a device instead of an app. Nonetheless, specialized mining devices are required to keep Bitcoin secure. The phrasing *one-CPU-one-vote* of the original Bitcoin paper (Satoshi Nakamoto, 2008) should not be understood as Bitcoin needing actual generic CPUs doing the proof-of-work. On the contrary, since 2008, the evolution of the internet at large[14] has given ample proof that specialized mining devices which vastly outperform generic CPUs are *required* to keep Bitcoin safe against botnets. Mining devices compete for the market of long term Bitcoin investors who want to maximize their returns in *bitcoins* while minimizing the TCO (total cost of ownership) of the devices. The key technical challenge is too maximize the hashrate that can be sustainably produced for a given power consumption.

Wallet: This app is intended to propagate various proofs of ownership of a secret while keeping the secret itself private forever. As complexity is the first enemy of software security, the inner design of a wallet is intentionally kept as bare as possible. In particular, the state of the wallet, which necessarily includes the secret itself, is minimized as much as possible, as every moving part within the state of the wallet is an extra opportunity to leak the secret. This implies that a wallet, which delivers security, *should not* include a balance store, a big data piece of software that is several orders of magnitude more complex - hence massively less secure - than the wallet itself. The wallet competes within the market of Bitcoin users to deliver a maximal amount of security for their holdings (bitcoins or tokens) with the minimal amount of friction for a given level of security. The key technical challenge of a wallet is to increase the security while improving the user experience in the same time.

---

[13] Most modern computing devices can tune their internal clock speed to balance their power consumption vs. their raw processing throughput. The raw processing throughput is typically sublinear with respect to the power consumption. Thus, when the kilowatt spot-price increases, it makes sense to start slowing down the mining devices before shutting them down altogether if the spot-price has become too high.

[14] During the last decade, botnets - i.e. a large network of compromised machines - have been causing increasingly large problems to the internet at large. Rogue IP cameras can be turned into devastating DDOS vectors. China is planning to have over half a billion IP cameras active by 2020 which does not augure anything good as far botnets are concerned https://www.qdaily.com/articles/47431.html Fortunately, the Chinese authorities put the *Great Firewall* in place years in advance to protect the rest of the world from those botnets.

Conclusions

The taxonomy presented above does not reflect Bitcoin's codebases as they stand today. Indeed, a lot of work remains to be done to achieve such a separation of concerns within the Bitcoin applicative landscape. This separation of concerns will ultimately be a co-evolutionary process associated with the emergence of more capable companies, by virtue of market specialization.

Language cannot be legislated, it's an ever evolving social construct. However, not all terminologies should be considered equal. As illustrated above, strategic terms frequently used in Bitcoin circles are misleading, leading to sterile discussions and to incorrect assessments of the desirability of certain evolutions of Bitcoin. The taxonomy proposed in this document attempts to address some of the most obvious flaws associated with the terminology in present use. It is up to the Bitcoin community at large to decide whether this taxonomy deserves to be used or not.